ST62 LCD DRIVER
ST624x/ST628x

DATABOOK

1st EDITION

MAY 1993

USE IN LIFE SUPPORT DEVICES OR SYSTEMS MUST BE EXPRESSLY AUTHORIZED

SGS-THOMSON PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPOR DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF SGS-THOMSON Microelectronics. A used herein:

1. Life support devices or systems are those which (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided with the product, can be reasonably expected to result in significant injury to the user.

2. A critical component is any component of a life suppo device or system whose failure to perform can reaso ably be expected to cause the failure of the life suppo device or system, or to affect its safety or effectiveness

TABLE OF CONTENTS

INTRODUCTION

SGS-THOMSON Microelectronics provides a wide range of microcontroller products to suit all major application environments. From the high level control of systems (INMOS Transputer, ST9 and ST10 families), through a range of intermediate level products (Second-sourced 6801, 6805, Z8) to controllers offering the economical solutions to the control of small systems. SGS-THOMSON has introduced the ST6 family, enhanced by the multi-purpose CMOS technology integrating non-volatile EPROM and EEPROM memories, to continue the level of economy initially offered by the COPs family.

The ST6 family has been developped to suit fully flexible control systems, by maximising the features integrated onto the silicon and accordingly minimising the number of external devices required. This brings the benefit of reducing the total system cost, very attractive for high volume control equipment among consumer, industrial and automotive applications.

ST62 OVERVIEW

Devices in the ST62 family are designed to exhibit key benefits for noisy environments, with their excellent EMI characteristics. Built-in hardware features minimise the noise generated by MCU operation and provide a high immunity to external noise.

Generation of noise is kept low by the careful control of the output buffer switching speeds and by employing an internal serial databus that minimises the number of transistors that switch simultaneously. The sensitivity to external noise is minimised by several features including the wide 3 to 6 volt supply voltage range and built-in protection diodes in the I/O ports.

Other major benefits include a full choice of EPROM, OTP and ROM versions to cover all needs from prototyping to high volume production; direct interfacing to analogue sensors, LEDs and power loads such as triacs and relays with the integral A/D converter and the 20mA output buffers. For user input and feedback, ST62 family members also provide efficient keyboard scanning configurations and direct LCD display drive. Several family members offer high reliability EEPROM for parameter storage.

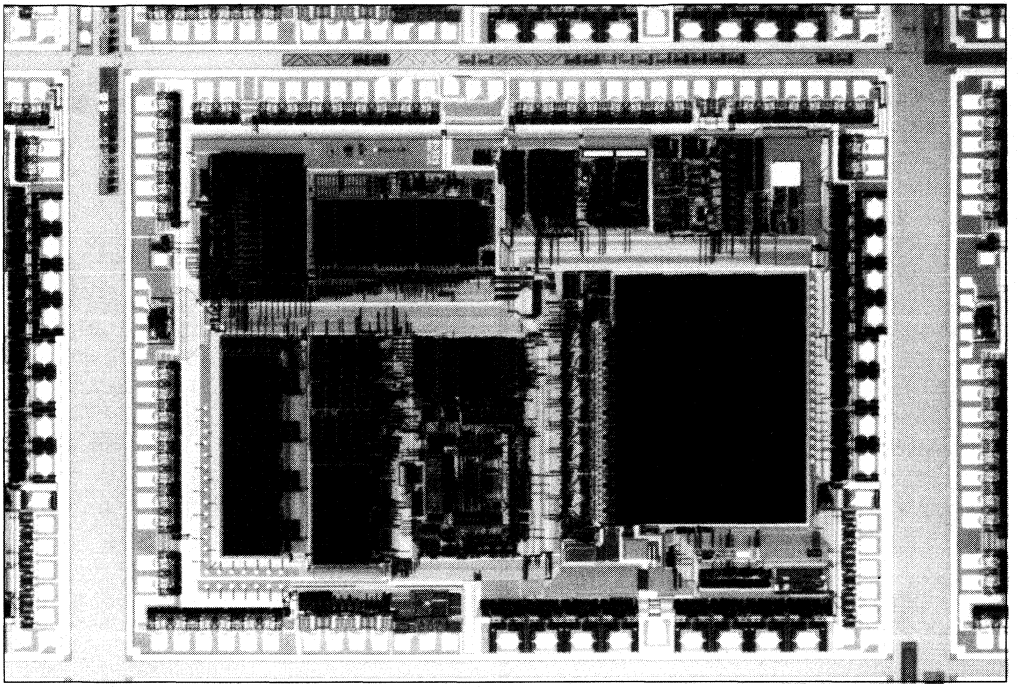
In addition to the EPROM and OTP ROM equivalent parts (for quick preproduction evaluation and test), shortening the critical development phase, reducing the Time to Market is also aided by full-feature development support tools: Assembler and Linker, Software simulator, Real-time Hardware Emulators and production EPROM programmers.

Further ST6 family members are dedicated to TV and Satellite tuning control applications (please refer to the SGS-THOMSON Video Products Databook, Volume 1 Signal Processing, for further information on the ST63 dedicated products).

ST6240 OVERVIEW

The new members of the ST62 family of microcontrollers, the ST6240, ST6242 and ST6245 include integrated LCD drivers making them ideal for shelf labelling, car radios and dashboards, thermostats, clocks, multimeters and many other applications requiring multidigit LCD display segments and/or non-volatile memory storage.

The ST6240 is an 8K ROM device with 192 bytes of RAM, 128 bytes of EEPROM, two timers and a 1 to 4 x 45 segment LCD driver, all housed in a Plastic QFP80 package. For applications not requiring on-chip non-volatile memory, the ST6242 contains 8K ROM, 128 bytes of RAM, one timer and a 1 to 4 x 40 segment LCD drive in a Plastic QFP64 package, while for minimal display requirements, but with non-volatile storage, the ST6245 includes 4K ROM, with 128 bytes of RAM, 64 bytes of EEPROM, two timers and a 1 to 4 x 24 segment LCD drive in a Plastic QFP52 package.



The ST62E40 EPROM version of the SGS-THOMSON ST62xx CMOS single chip microcontroller family.

ST62	ROM K	RAM	LCD RAM	EEPROM	I/O	A/D	LED	LCD	8bit TIMER	AR TIMER	SPI	PACKAGE
ST6210	2	64			12	8	4		1			DIP20/SO20
ST6215	2	64			20	16	4		1			DIP28/SO28
ST6220	4	64			12	8	4		1			DIP20/SO20
ST6225	4	64			20	16	4		1			DIP28/SO28
ST6240	8	192	24	128	16	12	4	4x45	2		1	QFP80
ST6242	8	128	24		10	6	4	4x40	1		1	QFP64
ST6245	4	128	12	64	11	7	4	4x24	2		1	QFP52
ST6260	4	128		128	13	7	6		1	1	1	DIP20/SO20
ST6265	4	128		128	21	13	8		1	1	1	DIP28/SO28
ST6280*	8	192	128	128	22	12	10	16x48	1	1	1	QFP100
ST6285*	8	192	96	-	12	8	4	16x40	1		1	QFP80
ST6292**	4	128		48	21	-	6		1	1	-	DIP28/SO28
ST6294**	4	128		128	21	13	8		1	1	1	DIP28/SO28

LED = Led or TRIAC driving AR TIMER = Auto Reload Timer SPI = Serial Peripheral Interface

* Under Development

** Please contact your nearest sales office

INTRODUCTION

The ST624x devices are all compatible with existing ST62 development tools and are supported by a ST6240 Starter Kit. This Starter Kit is designed to minimise project design time, and includes an on-board LCD, keyboard, I/O connectors to the user application and a socket for direct or on-board programming. Other forthcoming ST62 family members include devices with Dot Matrix LCD drive capability.

TOOLS FOR ST62XX FAMILY

DEVICES		EMULATOR		
ROM	**EPROM/OTP**	**COMPLETE**	**DEDICATION BOARD**	**PROBE**
ST621X ST622X ST621XB ST622XB	ST62E1X/T1X ST62E2X/T2X ST62E1XB/T1XB ST62E2XB/T2XB	ST621X-EMU (5V) ST626X-EMU (3/6V) ST621X-EMU/UPG	ST621X-DBE ST626X-DBE	Included in EMU and DBE
ST624X	ST62E4X/T4X	ST6240-EMU (with probe) ST6242-EMU (with probe) ST6245-EMU (with probe)	ST624X-DBE (no probe)	ST6240-P/QFP ST6242-P/QFP ST6245-P/QFP
ST626X ST629X	ST62E6X/T6X ST62E9X/T9X	ST626X-EMU	ST626X-DBE	Included in EMU and DBE
ST628X	ST62E8X/T8X	ST6280-EMU (with probe) ST6285-EMU (with probe)	ST628X-DBE (no probe)	ST6280-P/QFP ST6285-P/QFP
Software tools		ST6-SW/PC		

DEVICES		EPROM PROGRAMMER			KIT
ROM	**EPROM/OTP**	**SINGLE EPROM**	**COMPLETE GANG**	**GANG ADAPTOR**	
ST621X ST622X ST621XB ST622XB	ST62E1X/T1X ST62E2X/T2X ST62E1XB/T1XB ST62E2XB/T2XB	ST62E1X-EPB/220 ST62E1X-EPB/110	ST62E10-GP/DIP ST62E10-GP/SO ST62E15-GP/DIP ST62E15-GP/SO	ST62E10-GPA/DIP ST62E10-GPA/SO ST62E15-GPA/DIP ST62E15-GPA/SO	ST6220-KIT/220 ST6220-KIT/110 ST6220-KIT/UK ST622X-PWRKIT/50 ST622X-PWRKIT/60 ST622x-CHARGKIT
ST624X	ST62E4X/T4X	ST62E4X-EPB-PC ST62E4X-EPB/220 ST62E4X-EPB/110	ST62E40-GP/QFP ST62E42-GP/QFP ST62E45-GP/QFP	ST62E40-GPA/QFP ST62E42-GPA/QFP ST62E45-GPA/QFP	ST6240-KIT/220 ST6240-KIT/110 ST6240-KIT/UK
ST626X ST629X	ST62E6X/T6X ST62E9X/T9X	ST62E6X-EPB/220 ST62E6X-EPB/110	ST62E60-GP/DIP ST62E60-GP/SO ST62E65-GP/DIP ST62E65-GP/SO ST62E94-GP/DIP ST62E94-GP/SO	ST62E60-GPA/DIP ST62E60-GPA/SO ST62E65-GPA/DIP ST62E65-GPA/SO ST62E94-GPA/DIP ST62E94-GPA/SO	
ST628X	ST62E8X/T8X	ST62E8X-EPB/220 ST62E8X-EPB/110	Under development	Under development	
Software tools					ST62-fuzzy/PC

GENERAL INDEX

APPLICATION NOTES

ST624x DATASHEETS

ST624x DATASHEETS INDEX

ST624x DATASHEETS INDEX

ST624x DATASHEETS INDEX

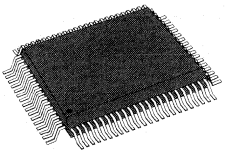
SGS-THOMSON MICROELECTRONICS

ST6240

8-BIT HCMOS MCU WITH LCD DRIVER, EEPROM AND A/D CONVERTER

PRELIMINARY DATA

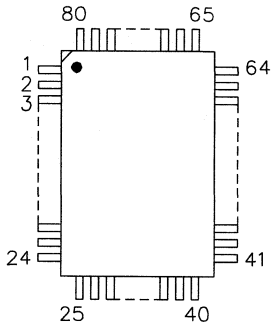
- 3 to 6V supply operating range
- 8.4MHz Maximum Clock Frequency
- -40 to +85°C Operating Temperature Range
- Run, Wait & Stop Modes
- 5 different interrupt vectors
- Look-up table capability in ROM
- User ROM: 7948 bytes
  Data RAM: 192 bytes
  LCD RAM: 24 bytes
  EEPROM: 128 bytes
- PQFP80 package
- 16 fully software programmable I/O as:
  - Input with/without pull-up resistor
  - Input with interrupt generation
  - Open-Drain or Push-pull outputs
  - Analog Inputs (12 pins)
- 4 I/O lines can sink up to 20mA for direct LED or TRIAC driving, and have SPI alternate functions
- Two 8-bit counters with 7-bit programmable prescalers (Timers 1 and 2)
- Software or hardware activated digital watchdog
- 8-bit A/D converter with up to 12 analog inputs
- 8-bit synchronous Serial Peripheral Interface (SPI)
- LCD driver with 45 segment outputs, 4 back-plane outputs and selectable duty cycle for up to 180 LCD segments direct driving
- 32kHz oscillator for stand-by LCD operation
- Power Supply Supervisor (PSS)
- One external not maskable interrupt
- 9 powerful addressing modes
- The accumulator, the X, Y, V & W registers, the port and peripherals data & control registers are addressed in the data space as RAM locations.
- The ST62E40 is the EPROM version, ST62T40 is the OTP version
- Development tool: ST6240-EMU connected via RS232 to an MS-DOS Personal Computer



PQFP80

(Ordering Information at the end of the datasheet)

Figure 1. 80 Pin Quad Flat Pack (QFP) Package Pinout



ST6240 Pin Description

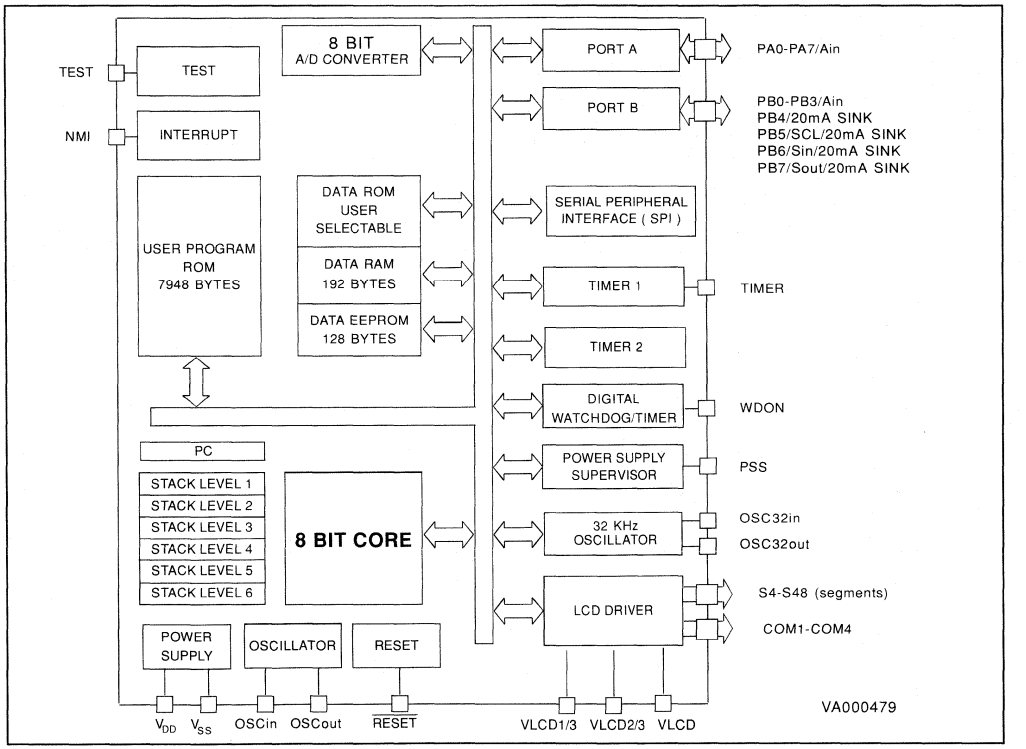
Pin number	Pin name	Pin number	Pin name	Pin number	Pin name	Pin number	Pin name
1	S43	25	$\overline{\text{RESET}}$	64	S26	65	S27
2	S44	26	OSCout	63	S25	66	S28
3	S45	27	OSCin	62	S24	67	S29
4	S46	28	WDON	61	S23	68	S30
5	S47	29	NMI	60	S22	69	S31
6	S48	30	TIMER	59	S21	70	S32
7	COM4	31	PB7/Sout (1)	58	S20	71	S33
8	COM3	32	PB6/Sin (1)	57	S19	72	S34
9	COM2	33	PB5/SCL (1)	56	S18	73	S35
10	COM1	34	PB4 (1)	55	S17	74	S36
11	VLCD1/3	35	PB3/Ain	54	S16	75	S37
12	VLCD2/3	36	PB2/Ain	53	S15	76	S38
13	VLCD	37	PB1/Ain	52	S14	77	S39
14	PA7/Ain	38	PB0/Ain	51	S13	78	S40
15	PA6/Ain	39	OSC32out	50	S12	79	S41
16	PA5/Ain	40	OSC32in	49	S11	80	S42
17	PA4/Ain			48	S10		
18	TEST			47	S9		
19	PA3/Ain			46	S8		
20	PA2/Ain			45	S7		
21	PA1/Ain			44	S6		
22	PA0/Ain			43	S5		
23	V_{DD}			42	S4		
24	V_{SS}			41	PSS		

Note 1: 20mA Sink

GENERAL DESCRIPTION

The ST6240 microcontroller is a member of the 8-bit HCMOS ST62xx family, a series of devices oriented to low-medium complexity applications. All ST62xx members are based on a building block approach: a common core is associated with a combination of on-chip peripherals (macrocells). The macrocells of the ST6240 are: a high performance LCD controller/driver with 45 segment outputs and 4 backplanes able to drive up to 180 segments, two Timer peripherals each including an 8-bit counter with a 7-bit software programmable prescaler (Timer), the digital watchdog (DWD), an 8-bit A/D Converter with up to 12 analog inputs, a Power Supply Supervisor and an 8-bit synchronous Serial Peripheral Interface (SPI). In addition these devices offer 128 bytes of EEPROM for storage of non volatile data. Thanks to these peripherals the ST6240 is well suited for general purpose, automotive, security, appliance and industrial applications. The ST62E40 EPROM version is available for prototypes and low-volume production, an OTP version is also available (see separate datasheet).

Figure 2. ST6240 Block Diagram



Note:
Ain= Analog Input

PIN DESCRIPTION

V_{DD} and V_{SS}. Power is supplied to the MCU using these two pins. V_{DD} is power and V_{SS} is the ground connection.

OSCin and OSCout. These pins are internally connected with the on-chip oscillator circuit. A quartz crystal or a ceramic resonator can be connected between these two pins in order to allow the correct operation of the MCU with various stability/cost trade-offs. OSCin is the input pin, OSCout is the output pin. An external clock signal can be applied to OSCin.

$\overline{\text{RESET}}$. The active low $\overline{\text{RESET}}$ pin is used to restart the microcontroller at the beginning of its program. The $\overline{\text{RESET}}$ pin is provided with an on-chip pull-up resistor and schmitt trigger input characteristics.

TEST. The TEST must be held at V_{SS} for normal operation (an internal pull-down resistor selects normal operating mode if TEST pin is not connected).

NMI. The NMI pin provides the capability for asynchronous applying an external top priority interrupt to the MCU.This pin is falling edge sensitive. The NMI pin is provided with an on-chip pull-up resistor and schmitt trigger input characteristics.

TIMER. This is the TIMER 1 I/O pin. In input mode it is connected to the prescaler and acts as external timer clock or as control gate for the internal timer clock. In the output mode the TIMER pin outputs the data bit when a time-out occurs.

WDON. This pin selects the watchdog enabling option (hardware or software). A low level selects the hardware activated option (the watchdog is always active), a high level selects the software activated option (the watchdog can be activated by software, deactivated only by reset, thus enabling STOP mode). An internal pull-up resistance selects the software watchdog option if the WDON pin is not connected.

PA0-PA7. These 8 lines are organized as one I/O port (A). Each line may be configured under software control as an input with or without pull-up resistor, interrupt generating input with pull-up resistor, open-drain or push-pull output or as analog input for the A/D converter. Port A has a 5mA drive capability in output mode.

PB0-PB3,PB4-PB7. These 8 lines are organized as one I/O port (B). Each line may be configured under software control as an input with or without internal pull-up resistor, interrupt generating input with pull-up resistor, open-drain or push-pull output. PB0-PB3 can be programmed as analog inputs for the A/D converter while PB4-PB7 can also sink 20mA for direct LED driving. PB5-PB7 can also be used as respectively Clock, Data in and Data out pins for the on-chip SPI to carry the synchronous serial I/O signals.

COM1-COM4. These four pins are the LCD peripheral common outputs. They are the outputs of the on-chip backplane voltage generator which is used for multiplexing the 45 LCD lines allowing up to 180 segments to be driven.

S4-S48. These pins are the 45 LCD peripheral driver outputs of ST6240. Segments S1-S3 are not connected to any pin.

VLCD. Display voltage supply. It determines the high voltage level on COM1-COM4 and S4-S48 pins.

VLCD1/3, VLCD2/3. Display supply voltage inputs for determining the display voltage levels on COM1-COM4 and S4-S48 pins during multiplex operation.

PSS. This is the Power Supply Supervisor sensing pin. When the voltage applied to this pin is falling below a software programmed value the highest priority (NMI) interrupt can be generated. This pin has to be connected to the voltage to be supervised.

OSC32in and OSC32out. These pins are internally connected with the on-chip 32kHz oscillator circuit. A 32.768kHz quartz crystal can be connected between these two pins if it is necessary to provide the LCD stand-by clock and real time interrupt. OSC32in is the input pin, OSC32out is the output pin.

ST62xx CORE

The core of the ST62xx Family is implemented independently from the I/O or memory configuration. Consequently, it can be treated as an independent central processor communicating with I/O and memory via internal addresses, data, and control busses. The in-core communication is arranged as shown in Figure 3; the controller being externally linked to both the reset and the oscillator, while the core is linked to the dedicated on-chip macrocells peripherals via the serial data bus and indirectly for interrupt purposes through the control registers.

Registers

The ST62xx Family core has six registers and three pairs of flags available to the programmer. They are shown in Figure 4 and are explained in the following paragraphs.

Accumulator (A). The accumulator is an 8-bit general purpose register used in all arithmetic calculations, logical operations, and data manipulations. The accumulator is addressed in the data space as RAM location at address FFh. Accordingly, the ST62xx instruction set can use the accumulator as any other register of the data space.

Figure 4. ST62xx Core Programming Model

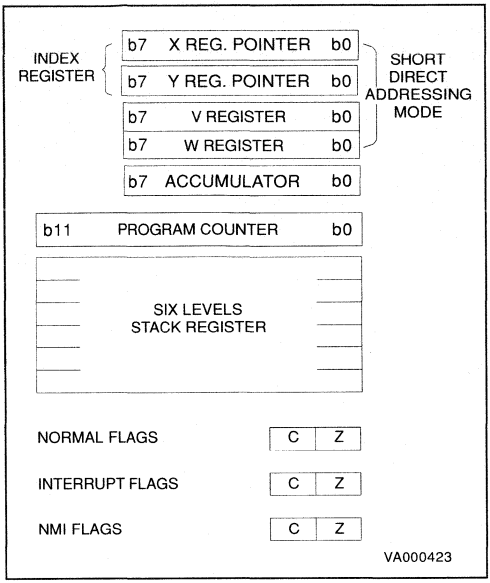
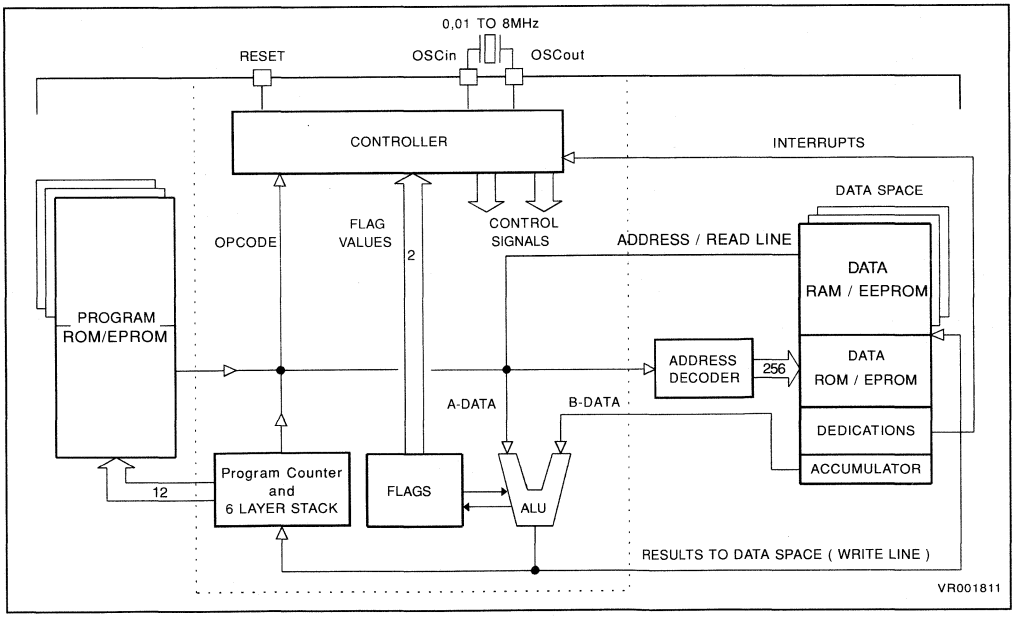


Figure 3. ST62xx Core Block Diagram



ST62xx CORE (Continued)

Indirect Registers (X, Y). These two indirect registers are used as pointers to memory locations in the data space. They are used in the register-indirect addressing mode. These registers can be addressed in the data space as RAM locations at addresses 80h (X) and 81h (Y). They can also be accessed with the direct, short direct, or bit direct addressing modes. Accordingly, the ST62xx instruction set can use the indirect registers as any other register of the data space.

Short Direct Registers (V, W). These two registers are used to save one byte in short direct addressing mode. These registers can be addressed in the data space as RAM locations at addresses 82h (V) and 83h (W). They can also be accessed with the direct and bit direct addressing modes. Accordingly, the ST62xx instruction set can use the short direct registers as any other register of the data space.

Program Counter (PC)

The program counter is a 12-bit register that contains the address of the next ROM location to be processed by the core. This ROM location may be an opcode, an operand, or an address of operand. The 12-bit length allows the direct addressing of 4096 bytes in the program space. Nevertheless, if the program space contains more than 4096 locations, as for the ST6240, the further program space can be addressed by using the Program ROM Page register.
The PC value is incremented after it is read from the address of the current instruction. To execute relative jumps the PC and the offset are shifted through the ALU, where they will be added, and the result is shifted back into the PC. The program counter can be changed in the following ways:

- JP (Jump) instruction . . . PC=Jump address
- CALL instruction . . . . PC=Call address
- Relative Branch instructions . . . . . PC=PC ± offset
- Interrupt . . . . . PC=Interrupt vector
- Reset . . . . . PC=Reset vector
- RET & RETI instructions . PC=Pop (stack)
- Normal instruction . . . . PC=PC+1

Flags (C, Z)

The ST62xx core includes three pairs of flags that correspond to 3 different modes: normal mode, interrupt mode and Non-Maskable Interrupt-Mode. Each pair consists of a CARRY flag and a ZERO flag. One pair (CN, ZN) is used during normal operation, one pair is used during the interrupt mode (CI, ZI) and one is used during the not-maskable interrupt mode (CNMI, ZNMI).

The ST62xx core uses the pair of flags that correspond to the actual mode: as soon as an interrupt (resp. a Non-Maskable-Interrupt) is generated, the ST62xx core uses the interrupt flags (resp. the NMI flags) instead of the normal flags. When the RETI instruction is executed, the normal flags (resp. the interrupt flags) are restored if the MCU was in the normal mode (resp. in the interrupt mode) before the interrupt. It should be observed that each flag set can only be addressed in its own mode (Not-maskable interrupt, normal interrupt or main mode). The flags are not cleared during the context switching and so remain in the state they were at the exit of the last mode switch.

The Carry flag is set when a carry or a borrow occurs during arithmetic operations, otherwise it is cleared. The Carry flag is also set to the value of the bit tested in a bit test instruction, and participates in the rotate left instruction.

The Zero flag is set if the result of the last arithmetic or logical operation was equal to zero, otherwise it is cleared.

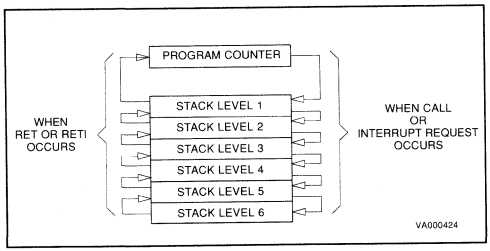
The switching between the three sets of Flags is automatically performed when an NMI, an interrupt or a RETI instruction occurs. As the NMI mode is automatically selected after the reset of the MCU, the ST62xx core uses at first the NMI flags.

ST62xx CORE (Continued)

Stack

The ST62xx core includes a true LIFO hardware stack that eliminates the need for a stack pointer. The stack consists of six separate 12-bit RAM locations that do not belong to the data space RAM area. When a subroutine call (or interrupt request) occurs, the contents of each level is shifted into the next level while the content of the PC is shifted into the first level (the value of the sixth level will be lost). When a subroutine or interrupt return occurs (RET or RETI instructions), the first level register is shifted back into the PC and the value of each level is popped back into the previous level. These two operating modes are described in Figure 5. Since the accumulator, as all other data space registers, is not stored in the stack, the handling of these registers should be performed inside the subroutine. The stack pointer will remain in its deepest position if more than 6 calls or interrupts are executed, so that the last return address will be lost. It will also remain in its highest position if the stack is empty and a RET or RETI is executed. In this case the next instruction will be executed.

Figure 5. Stack Operation



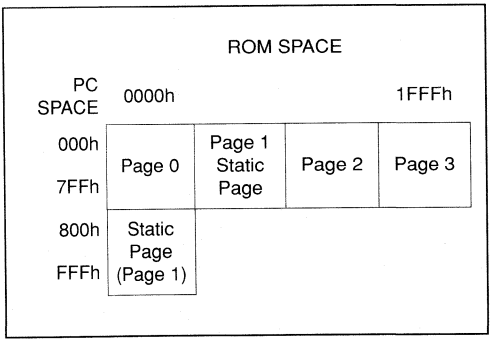
MEMORY SPACES

The MCU operates in three different memory spaces: program space, data space, and stack space. A description of these spaces is shown in the following figures.

Program Space

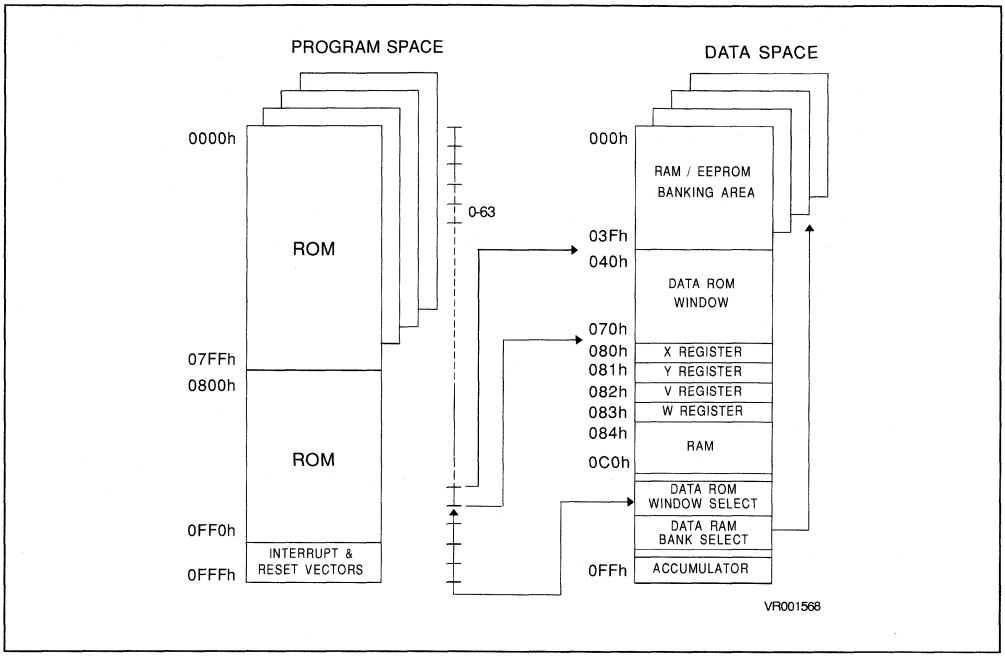
The program space is physically implemented in the ROM memory and includes all the instructions that are to be executed, as well as the data required for the immediate addressing mode instructions, the reserved test area and the user vectors. It is addressed by the 12-bit Program Counter register (PC register) and so the ST62xx core can directly address up to 4K bytes of Program Space. Nevertheless, the Program Space can be extended by the addition of 2Kbyte ROM banks as it is shown in the following figure in which the ST6240 8Kbyte memory is described.

Figure 6. ST6240 8Kbytes Program Space Addressing Description



MEMORY SPACES (Continued)

Figure 7. ST62xx Memory Addressing Description Diagram



These banks are addressed in the 000h-7FFh locations of the Program Space by the Program Counter and by writing the appropriate code in the Program ROM Page Register (PRPR register) located at address CAh of the Data Space. Because interrupts and common subroutines should be available all the time, only the lower 2K byte of the 4K program space are bank switched while the upper 2K byte can be seen as static page. Table 2 gives the different codes that allow the selection of the corresponding banks. Note that, from the memory point of view, Page 1 and the Static Page represent the same physical memory: it is only two different ways of addressing the same locations. On the ST6240 a total of 8192 bytes of ROM have been implemented; 7948 are available as user ROM while 244 are reserved for SGS-THOMSON test purposes.

Table 1. ST6240 Program ROM Memory Map

ROM Page	Device Address	Description
Page 0	0000h-007Fh 0080h-07FFh	Reserved User ROM
Page 1 "STATIC"	0800h-0F9Fh 0FA0h-0FEFh 0FF0h-0FF7h 0FF8h-0FFBh 0FFCh-0FFDh 0FFEh-0FFFh	User ROM Reserved Interrupt Vectors Reserved NMI Vector Reset Vector
Page 2	0000h-000Fh 0010h-07FFh	Reserved User ROM
Page 3	0000h-000Fh 0010h-07FFh	Reserved User ROM

MEMORY SPACES (Continued)

Data Space

The instruction set of the ST62xx core operates on a specific space, named Data Space, that contains all the data necessary for the processing of the program. The Data Space allows the addressing of RAM memory, ST62xx core/peripheral registers, and read-only data such as constants and look-up tables.

Data ROM. All the read-only data is physically implemented in the ROM memory in which the Program Space is also implemented. The ROM memory contains consequently the program to be executed, the constants and the look-up tables needed for the program.

The locations of Data Space in which the different constants and look-up tables are addressed by the ST62xx core can be considered as being a 64-byte window through which it is possible to access to the read-only data stored in the ROM memory .

Data RAM/EEPROM. The ST6240 offers 192 bytes of data RAM memory and 128 bytes of EEPROM. 64 bytes of RAM are directly addressed in data space in the range 080h-0BFh (static space). The additional RAM and the EEPROM are addressed using the banks of 64 bytes located between addresses 00h and 3Fh.

Additionally RAM are available in the LCD data map from E0h to F7h and are not banked.

Stack Space

The stack space consists of six 12 bit registers that are used for stacking subroutine and interrupt return addresses plus the current program counter register.

Figure 8. ST6240 Data Memory Space

DATA RAM/EEPROM BANK AREA	000h 03Fh
DATA ROM WINDOW AREA	040h 07Fh
X REGISTER	080h
Y REGISTER	081h
V REGISTER	082h
W REGISTER	083h
DATA RAM 60 BYTES	084h 0BFh
PORT A DATA REGISTER	0C0h
PORT B DATA REGISTER	0C1h
SPI INT. DISABLE REGISTER	0C2h*
RESERVED	0C3h
PORT A DIRECTION REGISTER	0C4h
PORT B DIRECTION REGISTER	0C5h
RESERVED	0C6h
RESERVED	0C7h
INTERRUPT OPTION REGISTER	0C8h*
DATA ROM WINDOW REGISTER	0C9h*
PROGRAM ROM PAGE REGISTER	0CAh*
DATA RAM/EEPROM BANK REGISTER	0CBh*
PORT A OPTION REGISTER	0CCh
RESERVED	0CDh
PORT B OPTION REGISTER	0CEh
RESERVED	0CFh
A/D DATA REGISTER	0D0h
A/D CONTROL REGISTER	0D1h
TIMER 1 PRESCALER REGISTER	0D2h
TIMER 1 COUNTER REGISTER	0D3h
TIMER 1 STATUS/CONT REGISTER	0D4h
TIMER 2 PRESCALER REGISTER	0D5h
TIMER 2 COUNTER REGISTER	0D6h
TIMER 2 STATUS/CONT REGISTER	0D7h
WATCHDOG REGISTER	0D8h
RESERVED	0D9h
PSS STATUS/CONTROL REGISTER	0DAh
32kHz OSC. CONTROL REGISTER	0DBh
LCD MODE CONTROL REGISTER	0DCh
SPI DATA REGISTER	0DDh
RESERVED	0DEh
EEPROM CONTROL REGISTER	0DFh
LCD RAM	0E0h 0F7h
DATA RAM 7 BYTES	0F8h 0FEh
ACCUMULATOR	0FFh

* WRITE ONLY REGISTER

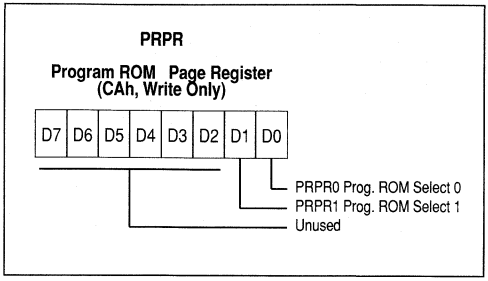
MEMORY SPACES (Continued)

Program ROM Page Register (PRPR)

The PRPR register can be addressed like a RAM location in the Data Space at the address CAh; nevertheless it is a write only register that cannot be accessed with single-bit operations. This register is used to select the 2-Kbyte ROM bank of the Program Space that will be addressed. The number of the page has to be loaded in the PRPR register. Refer to the Program Space description for additional information concerning the use of this register. The PRPR register is not modified when an interrupt or a subroutine occurs.

Care is required when handling the PRPR register as it is write only. For this reason, it is not allowed to change the PRPR contents while executing interrupt service routine, as the service routine cannot save and then restore its previous content. This operation may be necessary if common routines and interrupt service routines take more than 2K bytes; in this case it could be necessary to divide the interrupt service routine into a (minor) part in the static page (start and end) and to a second (major) part in one of the dynamic pages. If it is impossible to avoid the writing of this register in interrupt service routines, an image of this register must be saved in a RAM location, and each time the program writes to the PRPR it must write also to the image register. The image register must be written before PRPR, so if an interrupt occurs between the two instructions the PRPR is not affected.

Figure 9. Program ROM Page Register



D7-D2. These bits are not used.

PRPR1-PRPR0. These are the program ROM banking bits and the value loaded selects the corresponding page to be addressed in the lower part of the 4K program address space as specified in Table 2.

Table 2. ST6240 8Kbytes Program ROM Page Register Coding

PRPR1	PRPR0	PC bit 11	Memory Page
X	X	1	Static Page (Page1)
0	0	0	Page 0
0	1	0	Page 1 (Static Page)
1	0	0	Page 2
1	1	0	Page 3

This register is undefined on reset. Neither read nor single bit instructions may be used to address this register.

Note:

Only the lower part of address space is bank switched because interrupt vectors and common subroutines should be available at all times. The reason of this structure is due to the fact that it is not possible to jump from one dynamic page to another except by jumping back to the static page, changing contents of PRPR, and then jumping to a different dynamic page.

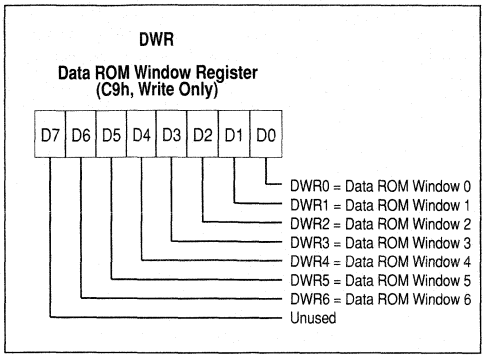
MEMORY SPACES (Continued)

Data Window register (DWR)

The Data ROM window is located from address 040h to address 7Fh in the Data space. It allows the direct reading of 64 consecutive bytes located anywhere in the ROM memory between the addresses 0000h and 1FFFh. All the bytes of the ROM memory can be used to store either instructions or read-only data. Indeed, the window can be moved by step of 64 bytes along the ROM memory in writing the appropriate code in the Write-only Data Window register (DWR register, location C9h).

The DWR register can be addressed like a RAM location in the Data Space at the address C9h, nevertheless it is a write only register that cannot be accessed with single-bit operations. This register is used to move the 64-byte read-only data window (from the 40h address to 7Fh address of the Data Space) up and down the ROM memory of the MCU in steps of 64 bytes. The effective address of the byte to be read as a data in the ROM memory is obtained by the concatenation of the 6 least significant bits of the register address given in the instruction (as least significant bits) and the content of the DWR register (as most significant bits, see Figure 10). So when addressing location 40h of dataspace, and 0 is loaded in the DWR register, the phisycal addressed location in ROM is 00h. The DWR register is not cleared at reset, therefore it must be written to before the first access to the Data ROM window area.

Figure 11. Data ROM Window Register



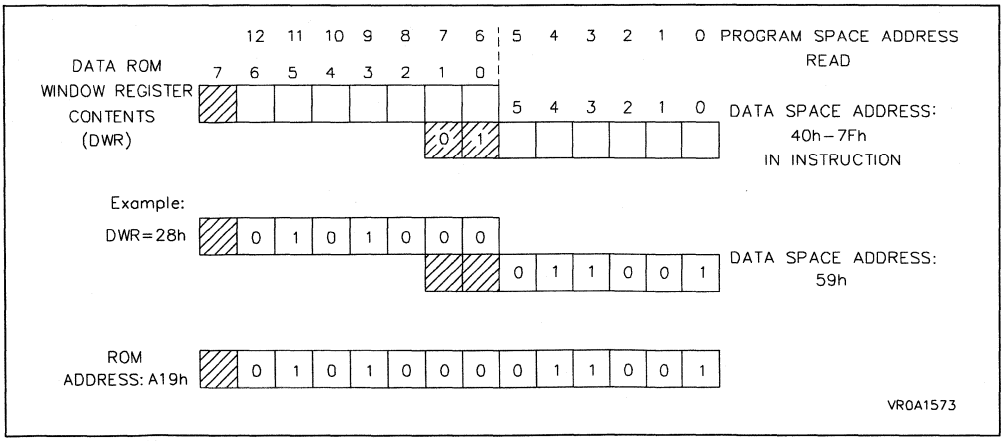
D7. This bit is not used.

DWR6-DWR0. These are the Data ROM Window bits that correspond to the upper bits of the data ROM space.

This register is undefined on reset. Neither read nor single bit instructions may be used to address this register.

Note: Care is required when handling the DWR register as it is write only. For this reason, it is not allowed to change the DWR contents while executing interrupt service routine, as the service routine cannot save and then restore its previous content. If it is impossible to avoid the writing of this register in the interrupt service routine, an image of this register must be saved in a RAM location, and each time the program writes to the DWR it must write also to the image register. The image register must be written first, so if an interrupt occurs between the two instructions the DWR is not affected.

Figure 10. Data ROM Window Memory Addressing



MEMORY SPACES (Continued)

Data RAM/EEPROM Bank Register (DRBR)

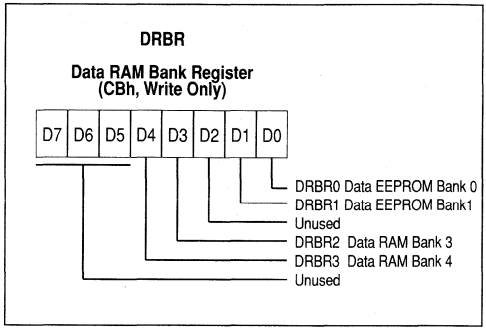
The selection of the bank is made by programming the Data RAM Bank Switch register (DRBR register) located at address CBh of the Data Space. The number of the selected bank is equal to the bit content of the DRBR register. In this way each bank of RAM or EEPROM can be selected 64 bytes at a time. No more than one bank should be set at a time.

The DRBR register can be addressed like a RAM location in the Data Space at the address CBh; nevertheless it is a write only register that cannot be accessed with single-bit operations. This register is used to select the desired 64-byte RAM/EEPROM bank of the Data Space. The number of the bank has to be loaded in the DRBR register and the instruction has to point to the selected location as if it was in bank 0 (from 00h address to 3Fh address). This register is not cleared during the MCU initialization, therefore it must be written before the first access to the Data Space bank region. Refer to the Data Space description for additional information. The DRBR register is not modified when a interrupt or a subroutine occurs.

Table 3. Data RAM Bank Register Set-up

DRBR Value	Selection
01h	EEPROM Page 0
02h	EEPROM Page 1
08h	RAM Page 1
10h	RAM Page 2

Figure 12. Data RAM Bank Register



The following table 3 summarizes how to set the data RAM bank register in order to select the various banks or pages.

D7-D5. These bits are not used.

DRBR4-DRBR3. Each of these bits, when set, will select one RAM page.

D2. This bit is not used.

DRBR1-DRBR0. Each of these bits, when set, will select one EEPROM page.

This register is undefined on reset. Neither read nor single bit instructions may be used to address this register.

Notes:

Care is required when handling the DRBR register as it is write only. For this reason, it is not allowed to change the DRBR contents while executing interrupt service routine, as the service routine cannot save and then restore its previous content. If it is impossible to avoid the writing of this register in interrupt service routine, an image of this register must be saved in a RAM location, and each time the program writes to DRBR it must write also to the image register. The image register must be written first, so if an interrupt occurs between the two instructions the DRBR is not affected.

In DRBR Register, *only 1 bit must be set.* Otherwise two or more pages are enabled in parallel, producing errors.

MEMORY SPACES (Continued)

EEPROM Description

The data space of ST62xx family from 00h to 3Fh is paged as described in Table 4. The ST6240 has 128 bytes of EEPROM located in two pages of 64 bytes (pages 0 and 1).

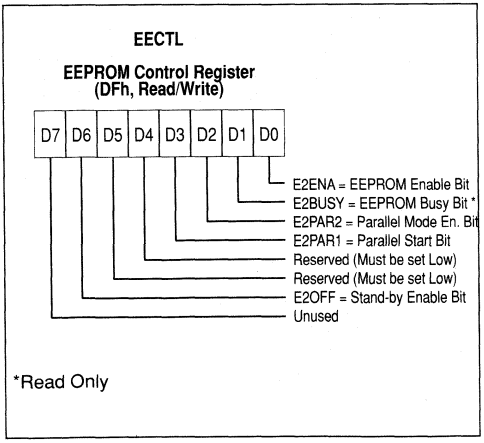
The EEPROM pages are physically organized in 32 byte modules (2 modules per page) and does not require dedicated instructions to be accessed in reading or writing. The EEPROM is controlled by the EEPROM Control Register (EECTL = DFh). In order to enable access to the EEPROM, bit 6 of this register must be cleared otherwise any access to the EEPROM will be meaningless.

Any EEPROM location can be read just like any other data location, also in terms of access time.

When writing to an EEPROM, the EEPROM is not accessible by the ST62xx. A busy flag can be read to identify the EEPROM status before attempting any access. Writing the EEPROM can work in two modes: Byte Mode (BMODE) and Parallel Mode (PMODE). BMODE is the normal way to use the EEPROM and consists in accessing one byte at a time. PMODE consists in accessing 8 bytes per time.

Readout of the EEPROM is made at the same speed as RAM acces.

Figure 13. EEPROM Control Register



D7. Not Used

E2OFF. *WRITE ONLY.* If this bit is set the EEPROM is disabled (any access will be meaningless) and the power consumption of the EEPROM is reduced to the lowest values.

D5, D4. Reserved, must be set to zero.

E2PAR1. *WRITE ONLY.* Once in Parallel Mode, as soon as the user software sets the E2PAR1 bit the parallel writing of the 8 adjacent registers will start. It is internally reset at the end of the programming procedure. Note that less than 8 bytes can be written; after parallel programming the undefined bytes will be unaffected

E2PAR2. *WRITE ONLY.* This bit must be set by the user program in order to perform parallel programming (more than one byte at a time). If E2PAR2 is set and the parallel start bit (E2PAR1) is low, up to 8 adjacent bytes can be written at maximum speed, the contents being stored in volatile registers. These 8 adjacent bytes are considered as a row, whose address lines A7, A6, A5, A4, A3 are fixed while A2, A1 and A0 are the changing bits. E2PAR2 is automatically reset at the end of any parallel programming procedure. It can be reset by the user software before starting the programming procedure, leaving the EEPROM registers unchanged.

E2BUSY. *READ ONLY.* This bit will be automatically set by the EEPROM control logic when the user program modifies an EEPROM register. The user program must test it before any read or write EEPROM operation; any attempt to access the EEPROM while the busy bit is set will be aborted and the writing procedure in progress completed.

E2•E••NA. *WRITE ONLY.* This bit MUST be set to one in order to write to any EEPROM register. If the user program attempts to write to the EEPROM when E2ENA = "0", the involved registers will be unaffected and the BS will not be set.

After RESET the content of EECTL register will be 00h.

Notes:

The data to write has to be written directly at the address that it will have inside the EEPROM space. There is no buffer memory between the data•RAM and the EEPROM spaces.

When the EEPROM is busy (E2BUSY = "1") EECTL can not be accessed in write mode, it is only possible to read the status of E2BUSY. This implies that as long as the EEPROM is busy, it is not possible to change the status of the EEPROM control register. EECTL bits 4 and 5 are reserved for test purposes, and must never be set to "1".

MEMORY SPACES (Continued)

Table 4. EEPROM Parallel Write Row Structure

Byte	0	1	2	3	4	5	6	7	Dataspace addresses. Banks 0 and 1.
ROW7									38h-3Fh
ROW6									30h-37h
ROW5									28h-2Fh
ROW4									20h-27h
ROW3									18h-1Fh
ROW2									10h-1Fh
ROW1									08h-0Fh
ROW0									00h-07h

Up to 8 bytes in each row may be programmed at the same time in Parallel Write mode

Additional Notes on Parallel Mode. If the user wishes to perform parallel programming, the first action should be to set the E2PAR2 bit to one. From this time the EEPROM will be addressed in writing, the ROW address will be latched and it will be possible to change it only at the end of the programming procedure or by resetting E2PAR2 without programming the EEPROM. After the ROW address latching the ST62xx can "see" only one EEPROM row (the selected one) and any attempt to write or read other rows will produce errors. Do not read the EEPROM while E2PAR2 is set.

As soon as E2PAR2 bit is set, the 8 volatile ROW latches are cleared. From this moment the user can load data in the whole ROW or in a subset. Setting E2PAR1 will modify the EEPROM registers corresponding to the ROW latches accessed after E2PAR2. For example, if the software sets E2PAR2 and accesses the EEPROM by writing to addresses 18h, 1Ah, 1Bh and then sets E2PAR1, these three registers will be modified at the same time; the remaining bytes will be unaffected. Note that E2PAR2 is internally reset at the end of the programming procedure. This implies that the user must set E2PAR2 bit between two parallel programming procedures. Note that if the user tries to set E2PAR1 while E2PAR2 is not set there will not be any programming procedure and the E2PAR1 bit will be unaffected. Consequently E2PAR1 bit cannot be set if E2ENA is low. E2PAR1 can be affected by the user to set it, only if E2ENA and E2PAR2 bits are also set to one.

TEST MODE

For normal operation the TEST pin must be held low. An on-chip 100kΩ pull-down resistor is internally connected to the TEST pin.

INTERRUPTS

The ST62xx core can manage 4 different maskable interrupt sources, plus one non-maskable interrupt source (top priority level interrupt). Each source is associated with a particular interrupt vector that contains a Jump instruction to the related interrupt service routine. Each vector is located in the Program Space at a particular address.
When a source provides an interrupt request, and the request processing is also enabled by the ST62xx core, then the PC register is loaded with the address of the interrupt vector (i.e. of the Jump instruction). Finally, the PC is loaded with the address of the Jump instruction and the interrupt routine is processed.

The ST6240 microcontroller has nine different interrupt sources associated to different interrupt vectors as described in Table 5.

Table 5. Interrupt Vectors - Sources Relationship

Interrupt Source	Associated Vector	Vector Address
NMI & PSS Pins	Interrupt Vector #0 (NMI)	(FFCh-FFDh)
SPI Peripheral	Interrupt Vector #1	(FF6h-FF7h)
Port A & B Pins	Interrupt Vector #2	(FF4h-FF5h)
TIMER 1, 2 & 32kHz Oscillator	Interrupt Vector #3	(FF2h-FF3h)
ADC Peripheral	Interrupt Vector #4	(FF0h-FF1h)

Interrupt Vectors Description

The ST62xx core includes 5 different interrupt vectors in order to branch to 5 different interrupt routines in the static page of the Program Space:

- The interrupt vector associated with the non-maskable interrupt source is named interrupt vector #0. It is located at addresses FFCh,FFDh in the Program Space. On ST6240 this vector is associated with the external falling edge sensitive interrupt pin (NMI) and is also connected to the Power Supply Supervisor circuit. The PSS and NMI interrupts are "ORed" together and the discrimination between PSS interrupt and NMI interrupt can be done by reading the interrupt flag (bit 7) of the PSS control register (Address DAh). An on-chip 100kΩ pull-up resistor is internally connected to the NMI pin.
- The interrupt vector located at the addresses FF6h, FF7h is named interrupt vector #1. It is associated with SPI peripheral and can be programmed by software to generate an interrupt request after the falling edge or low level of the eighth external clock pulse according to the code loaded in the Interrupt Option Register (IOR).
- The interrupt vector located at the addresses FF4h, FF5h is named interrupt vector #2. It is associated with Port A and B pins and can be programmed by software either in the falling edge detection mode or in the rising edge detection mode according to the code loaded in the Interrupt Option Register (IOR).
- The interrupt vector located at the addresses FF2h, FF3h is named interrupt vector #3. It is associated with Timer 1, Timer 2 and the 32kHz Oscillator peripherals. All these interrupts are "ORed" together and are connected to interrupt line #3 of the core. Discrimination among the three interrupts must be made by polling the Status/Control registers of Timer 1 (0D4h), Timer 2 (0D7h) and 32kHz oscillator (0DBh).
- The interrupt vector located at the addresses FF0h, FF1h is named interrupt vector #4. It is associated with the A/D converter peripheral.

All the on-chip peripherals (refer to their descriptions for further details) have an interrupt request flag bit (TMZ for timer, EOC for A/D, etc.), this bit is set to one when the device wants to generate an interrupt request and a mask bit (ETI for timer, EAI for A/D, etc.) that must be set to one to allow the transfer of the flag bit to the Core.

Interrupt Priority

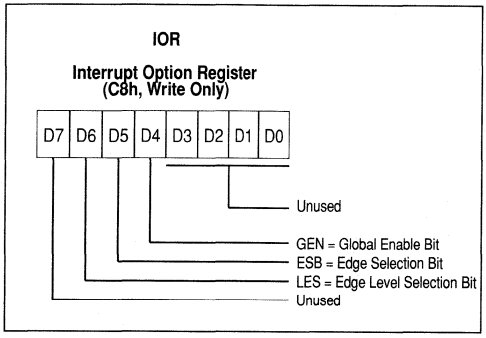
The non-maskable interrupt request has the highest priority and can interrupt any other interrupt routines at any time, nevertheless the four other interrupts cannot interrupt each other. If more than one interrupt request is pending, they are processed by the ST62xx core according to their priority level: vector #1 has the higher priority while vector #4 the lower. The priority of each interrupt source is fixed.

INTERRUPTS (Continued)

Interrupt Option Register

The Interrupt Option Register (IOR register, location C8h) is used to enable/disable the individual interrupt sources and to select the operating mode of the external interrupt inputs. This register can be addressed in the Data Space as RAM location at the address C8h, nevertheless it is a write-only register that cannot be accessed with single-bit operations. The operating modes of the external interrupt inputs associated to interrupt vectors #1 and #2 are selected through bits 5 and 6 of the IOR register.

Figure 14. Interrupt Option Register



D7. This bit is not used.

LES. Level/Edge Selection Bit. When this bit is set to one, the interrupt #1 (SPI) is low level sensitive, when cleared to zero the negative edge sensitive interrupt is selected.

ESB. Edge Selection Bit. When this bit is set to one, the interrupt #2 (Port A & B lines) is positive edge sensitive, when cleared to zero the negative edge sensitive interrupt is selected.

GEN. Global Enable Interrupt. When this bit is set to one, all the interrupts are enabled. When this bit is cleared to zero all the interrupts (including the NMI/PSS) are disabled.

This register is cleared on reset.

Table 6. Interrupt Option Register Description

GEN	SET	Enable all interrupts
	CLEARED	Disable all interrupts
ESB	SET	Rising edge mode on interrupt input #2
	CLEARED	Falling edge mode on interrupt input #2
LES	SET	Level-sensitive mode on interrupt input #1
	CLEARED	Falling edge mode on interrupt input #1
OTHERS	NOT USED	

External Interrupts Operating Modes

The NMI interrupt is associated to the NMI and PSS external pins of the ST6240. The two interrupt requests are "ORed". The highest priority interrupt request will be generated either by a falling edge applied to the NMI pin or when the voltage level applied to the PSS pin goes below the software programmed value. The discrimination between NMI and PSS interrupt can be done by polling the interrupt flag (Bit 7) of the PSS control register (DAh). The NMI interrupt pin signal is latched and is automatically reset by the core at the beginning of the non-maskable interrupt service routine. An on-chip pull-up resistor and a schmitt trigger is available with the NMI pin.

The two interrupt sources associated with the falling/rising edge mode of the external interrupt pins (SPI vector #1, Ports A and B vector #2,) are connected to two internal latches. Each latch is set when a falling/rising edge occurs and is cleared when the associated interrupt routine is started. So, the occurrence of an external interrupt request is stored: a second interrupt, that occurs during the processing of the first one, will be processed as soon as the first one has been finished (if there is not an higher priority interrupt request). If more than one interrupt occurs during the processing of the first one, these other interrupt requests will be lost.

The storage of the interrupt requests is not available in the level sensitive detection mode. To be taken into account, the low level must be present on the interrupt pin when the core samples the line after the execution of the instructions.

During the end of each instruction the core tests the interrupt lines and if there is an interrupt request the next instruction is not executed and the related interrupt routine is executed.

INTERRUPTS (Continued)

Interrupt Procedure. The interrupt procedure is very similar to a call procedure, indeed the user can consider the interrupt as an asynchronous call procedure. As this is an asynchronous event the user does not know about the context and the time at which it occurred. As a result the user should save all the data space registers which will be used inside the interrupt routines. There are separate sets of processor flags for normal, interrupt and non-maskable interrupt modes which are automatically switched and so these do not need to be saved.

The following list summarizes the interrupt procedure:

ST62xx actions

- Interrupt detection
- The flags C and Z of the main routine are exchanged with the flags C and Z of the interrupt routine (or the NMI flags)
- The value of the PC is stored in the first level of the stack
- The normal interrupt lines are inhibited (NMI still active)
- First internal latch is cleared
- The related interrupt vector is loaded in the PC.

User actions

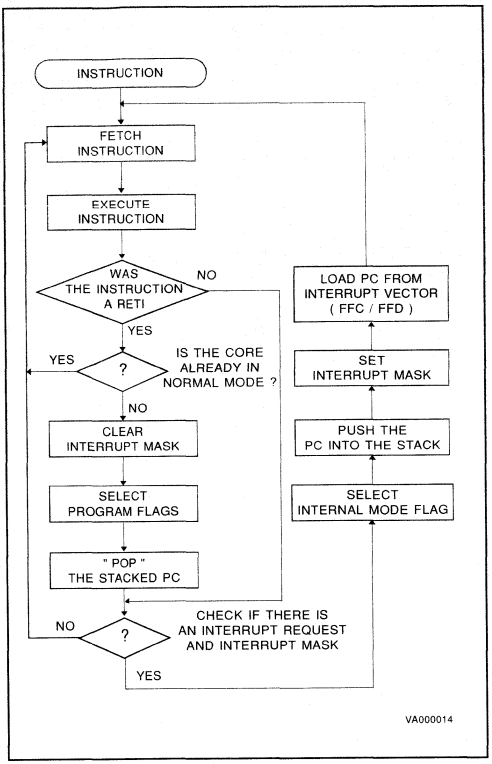
- User selected registers are saved inside the interrupt service routine (normally on a software stack)
- The source of the interrupt is found by polling (if more than one source is associated to the same vector) the interrupt flag of the source.
- Interrupt servicing
- Return from interrupt (RETI)

ST62xx actions

- Automatically the ST62xx core switches back to the normal flags (or the interrupt flags) and pops the previous PC value from the stack

The interrupt routine begins usually by the identification of the device that has generated the interrupt request (by polling). The user should save the registers which are used inside the interrupt routine (that holds relevant data) into a software stack. After the RETI instruction execution, the core carries out the previous actions and the main routine can continue.

Figure 15. Interrupt Processing Flow-Chart



INTERRUPTS (Continued)

Interrupt request and mask bits

Interrupt Option Register, IOR Location C8h

- GEN. If this bit is set, all the ST62xx interrupts are enabled, if reset all interrupts are disabled (including the NMI).
- ESB. If this bit is set, all the input lines associated to interrupt vector #2 are rising edge sensitive, if reset they are falling edge sensitive.
- LES. If this bit is set, all the inputs lines associated to interrupt vector #1 are low level sensitive, if reset they are falling edge sensitive.

All other bits in this register are not used.

Timer Peripherals, TSCR1 and TSCR2 registers, locations D4h and D7h

- TMZ. A low-to-high transition indicates that the timer count register has decremented to zero. This means that an interrupt request can be generated in relation to the state of ETI bit.
- ETI. This bit, when set, enables the timer interrupt request.

A/D Converter Peripheral, ADCR register location D0h

- EOC. This read only bit indicates when a conversion has been completed, by going to one. An interrupt request can be generated in relation to the state of EAI bit.
- EAI. This bit, when set, enables the A/D converter interrupt request.

PSS Peripheral, PSSCR Register location DAh

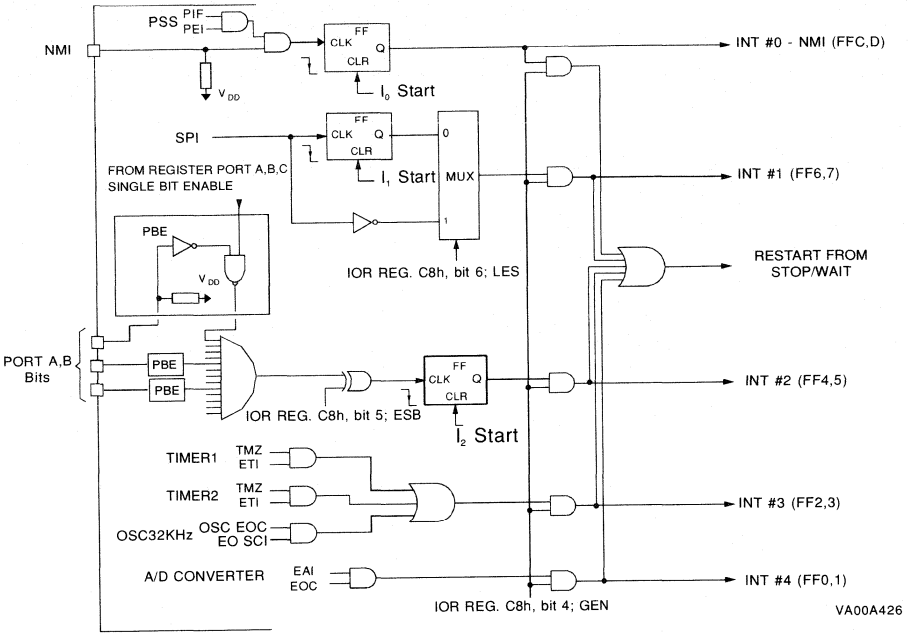
- PSSVD. This read only bit indicates when the voltage level applied to the PSS pin goes below the software programmed value. An interrupt request can be generated in relation to the state of PSSEI bit.
- PSSEI. This bit, when set, enables the Power Supply Supervisor interrupt request.

32kHz Oscillator, 320CR register location DBh

- EOSCI. This bit, when set, enables the 32kHz oscillator interrupt request.
- OSCEOC. This read only bit indicates when the 32kHz oscillator has measured a 500ms elapsed time (providing a 32.768kHz quartz crystal is connected to the 32kHz oscillator dedicated pins). An interrupt request can be generated in relation to the state of EOSCI bit.

INTERRUPTS (Continued)

Figure 16. ST6240 Interrupt Circuit Diagram



WARNING. *GEN is the global enable for all interrupts except NMI. If this bit is cleared, the NMI interrupt is accepted when the ST62xx core is in the normal RUN Mode.*
If the ST62xx core is in STOP or WAIT Mode, the NMI is not accepted as a restart is disabled. This state can only be finished by a reset (fromthe Watchdog or an external Reset Signal).
As a consequence the NMI can be masked in STOP and WAIT modes, but not in RUN mode.

RESET

The ST6240 can be reset in three ways: by the external reset input ($\overline{\text{RESET}}$) tied low, by power-on reset and by the digital watchdog/timer peripheral.

RESET Input

The $\overline{\text{RESET}}$ pin can be connected to a device of the application board in order to restart the MCU during its operation. The activation of the $\overline{\text{RESET}}$ pin may occur in the RUN, WAIT or STOP mode. This input has to be used to reset the MCU internal state and provide a correct start-up procedure. The pin is active low. The internal reset signal is generated by adding a delay to the external signal. Therefore even short pulses at the $\overline{\text{RESET}}$ pin will be accepted. This feature is valid providing that V_{DD} has finished its rising phase and the oscillator is correctly running (normal RUN or WAIT modes).

If $\overline{\text{RESET}}$ activation occurs in the RUN or Wait mode, the MCU is configured in the Reset mode for as long as the signal of the $\overline{\text{RESET}}$ pin is low. The processing of the program is stopped (in RUN mode only) and the Input/Outputs are in the High-impedance state with pull-up resistors switched on. As soon as the level on the $\overline{\text{RESET}}$ pin becomes high, the initialization sequence is executed.

If a $\overline{\text{RESET}}$ pin activation occurs in the STOP mode, the oscillator starts and all the inputs/outputs are configured in the High-impedance state with pull-up resistors switched on for as long as the level on the $\overline{\text{RESET}}$ pin remains low. When the level of the $\overline{\text{RESET}}$ pin becomes high, a delay is generated by the ST62xx core to wait that the oscillator becomes completely stabilized. Then, the initialization sequence is started.

Power-On Reset (POR)

The function of the POR consists in waking up the MCU during the power-on sequence. At the beginning of this sequence, the MCU is configured in the Reset state: every Input/Output port is configured in the input mode (High-impedance state with pull-up) and no instruction is executed. When the power supply voltage becomes sufficient, the oscillator starts to operate, nevertheless the ST62xx core generates a delay to allow the oscillator to be completely stabilized before the execution of the first instruction. The initialization sequence is then executed.

Internal circuitry generates a Reset pulse when V_{DD} is switched on. In the case of fast rising V_{DD} (transition time $\leq 100\mu s$), this reset pulse starts the internal reset procedure without the need of external components at the $\overline{\text{RESET}}$ pin. In cases of slowly or non monotonously rising V_{DD}, an external reset signal must be provided for a proper reset of the MCU.

For as long as the reset pin is kept at the low level, the processor remains in the reset state. The reset will be released after the voltage at the reset pin reaches the high level.

Note:

To have a correct ST62xx start-up, the user should take care that the reset input does not change to the high level before the V_{DD} level is sufficient to allow MCU operation at the chosen frequency (see recommended operating conditions).

An on-chip counter circuit provides a delay of 2048 oscillator cycles between the detection of the reset high level and the release of the MCU reset.

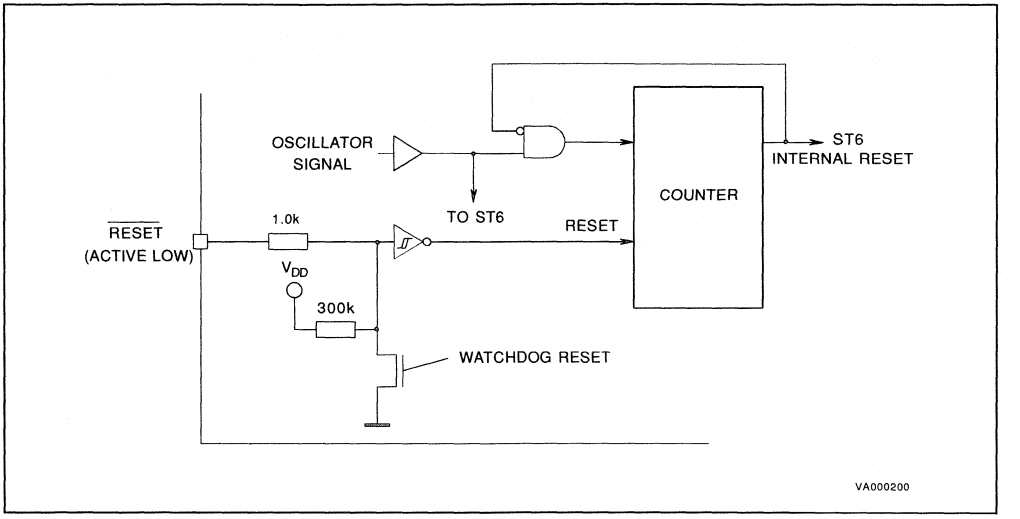
A proper reset signal for slow rising V_{DD}, i.e. the required delay between reaching sufficient operating voltage and the reset input changing to a high level, can be generally provided by an external capacitor connected between the $\overline{\text{RESET}}$ pin and V_{SS}.

RESET (Continued)

Watchdog Reset

The ST6240 provides an on-chip watchdog function in order to provide a graceful recovery from a software upset. If the watchdog register is not refreshed, preventing the end-of-count being reached, an internal circuit pulls down the $\overline{\text{RESET}}$ pin. The MCU will enter the reset state as soon as the voltage at $\overline{\text{RESET}}$ pin reaches the related low level. This also resets the watchdog which subsequently turns off the pull-down and activates the pull-up device at the $\overline{\text{RESET}}$ pin. This causes the positive transition at the $\overline{\text{RESET}}$ pin and terminates the reset state.

Figure 17. Reset Circuit

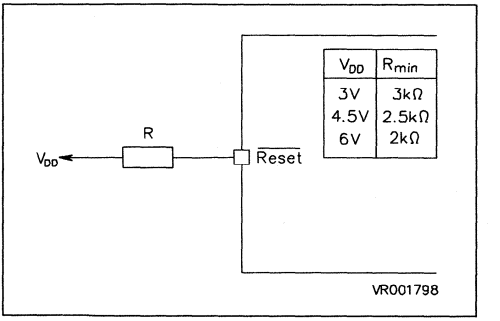


Application Notes

An external resistor between V_{DD} and reset pin is not required because an internal pull-up device is provided. If the user prefers, for any reason, to add an external pull-up resistor its value must comply with the Rmin value defined in Figure 18. If the value is lower than Rmin, the on-chip watchdog pull-down transistor might not be able to pull-down the reset pin resulting in an external deactivation of the watchdog function.

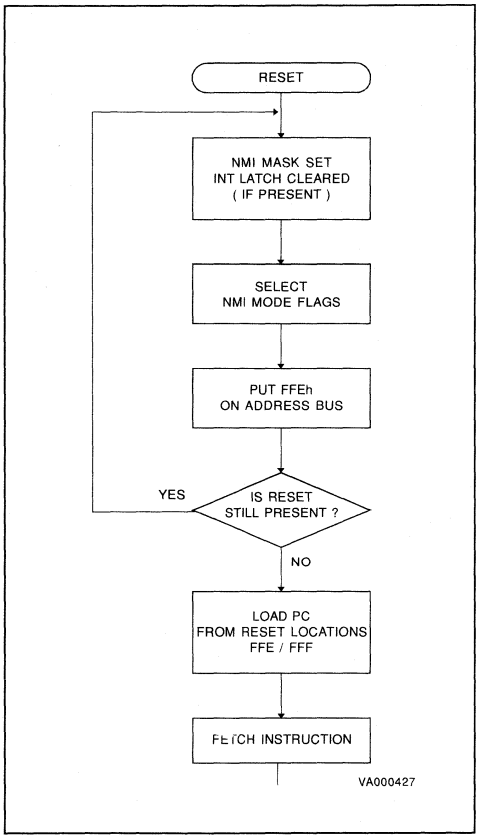
The POR function operates in a dynamic manner in the way that it brings about the initialization of the MCU when it detects a dynamic rising edge of the V_{DD} voltage. The typical detected threshold is about 2 volts, but the actual value of the detected threshold depends on the way in which the V_{DD} voltage rises up. The POR device DOES NOT allow the supervision of a static rising or falling edge of the V_{DD} voltage.

Figure 18. External Reset Resistance



RESET (Continued)

Figure 19. Reset & Interrupt Processing Flow-Chart



MCU Initialization Sequence

When a reset occurs the stack is reset to the program counter, the PC is loaded with the address of the reset vector (located in the program ROM at addresses FFEh & FFFh). A jump instruction to the beginning of the program has to be written into these locations. After a reset the interrupt mask is automatically activated so that the core is in non-maskable interrupt mode to prevent false or ghost interrupts during the restart phase. Therefore the restart routine should be terminated by a RETI instruction to switch to normal mode and enable interrupts. If no pending interrupt is present at the end of the reset routine the ST62xx will continue with the instruction after the RETI; otherwise the pending interrupt will be serviced.

Figure 20. Restart Initialization Program Flow-Chart

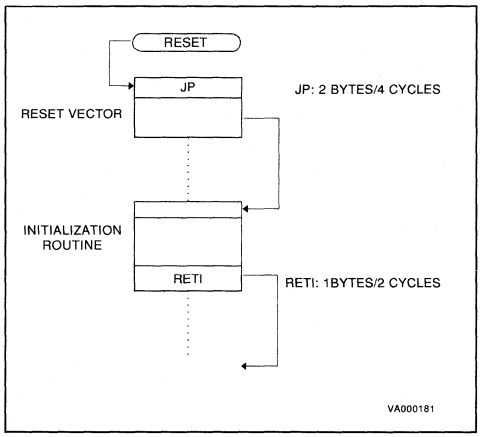


Table 7. Reset Configuration

Input/Output pins	Registers
Input Mode with pull-up and no interrupt	All cleared but A,X,Y,V,W, data RAM, LCD RAM, DWR (C9), PRPR (CA), DRBR (CB). Timers prescaler and TCR are initialized respectively at 7F and FF. Watchdog register DWDR (D8) is set to FEh.

WAIT & STOP MODES

The WAIT and STOP modes have been implemented in the ST62xx core in order to reduce the consumption of the product when the latter has no instruction to execute. These two modes are described in the following paragraphs

WAIT Mode

The configuration of the MCU in the WAIT mode occurs as soon as the WAIT instruction is executed. The microcontroller can also be considered as being in a "software frozen" state where the core stops processing the instructions of the routine, the contents of the RAM locations and peripheral registers are saved as long as the power supply voltage is higher than the RAM retention voltage but where the peripherals are still working.

The WAIT mode is used when the user wants to reduce the consumption of the MCU when it is in idle, while not losing count of time or monitoring of external events. The oscillator is not stopped in order to provide a clock signal to the peripherals. The timer counting may be enabled (writing the PSI bit in TSCR register) and the timer interrupt may be also enabled before entering the WAIT mode; this allows the WAIT mode to be left when timer interrupt occurs. The above explanation related to the timers applies also to the A/D converter.

If the exit from the WAIT mode is performed with a general RESET (either from the activation of the external pin or by watchdog reset) the MCU will enter a normal reset procedure as described in the RESET chapter. If an interrupt is generated during WAIT mode the MCU behavior depends on the state of the ST62xx core before the initialization of the WAIT sequence, but also of the kind of the interrupt request that is generated. This case will be described in the following paragraphs. In any case, the ST62xx core does not generate any delay after the occurrence of the interrupt because the oscillator clock is still available.

STOP Mode

If the Watchdog is disabled the STOP mode is available. When in STOP mode the MCU is placed in the lowest power consumption mode. In this operating mode the microcontroller can be considered as being "frozen", no instruction is executed, the oscillator is stopped, the contents of the RAM locations and peripheral registers are saved as long as the power supply voltage is higher than the RAM retention voltage, and the ST62xx core waits for the occurrence of an external interrupt request or Reset activation to output from the STOP state.

If the exit from the STOP mode is performed with a general RESET (by the activation of the external pin) the MCU will enter a normal reset procedure as described in the RESET chapter. The case of an interrupt depends on the state of the ST62xx core before the initialization of the STOP sequence and also of the kind of the interrupt request that is generated.

This case will be described in the following paragraphs. In any case, the ST62xx core generates a delay after the occurrence of the interrupt request in order to wait the complete stabilization of the oscillator before the execution of the first instruction.

Exit from WAIT and STOP Modes

The following paragraphs describe the output procedure of the ST62xx core from WAIT and STOP modes when an interrupt occurs (not a RESET). It must be noted that the restart sequence depends on the original state of the MCU (normal, interrupt or non-maskable interrupt mode) before the start of the WAIT or STOP sequence, but also of the type of the interrupt request that is generated.

Normal Mode. If the ST62xx core was in the main routine when the WAIT or STOP instruction has been executed, the ST62xx core outputs from the stop or wait mode as soon as any interrupt occurs; the related interrupt routine is executed and at the end of the interrupt service routine the instruction that follows the STOP or the WAIT instruction is executed if no other interrupts are pending.

WAIT & STOP MODES (Continued)

Not Maskable Interrupt Mode. If the STOP or WAIT instruction has been executed during the execution of the non-maskable interrupt routine, the ST62xx core outputs from the stop or wait mode as soon as any interrupt occurs: the instruction that follows the STOP or the WAIT instruction is executed and the ST62xx core is still in the non-maskable interrupt mode even if another interrupt has been generated.

Normal Interrupt Mode. If the ST62xx core was in the interrupt mode before the initialization of the STOP or WAIT sequence, it outputs from the stop or wait mode as soon as any interrupt occurs. Nevertheless, two cases have to be considered:

- If the interrupt is a normal interrupt, the interrupt routine in which the WAIT or STOP was entered will be completed with the execution of the instruction that follows the STOP or the WAIT and the ST62xx core is still in the interrupt mode. At the end of this routine pending interrupts will be serviced in accordance to their priority.
- If the interrupt is a non-maskable interrupt, the non-maskable routine is processed at first. Then the routine in which the WAIT or STOP was entered will be completed with the execution of the instruction that follows the STOP or the WAIT and the ST62xx core remains in the normal interrupt mode.

Notes :

To reach the lowest power consumption the user software must take care of:

- placing the A/D converter in its power down mode by clearing the PDS bit in the A/D control register before entering the STOP instruction.
- switching off the 32kHz oscillator by clearing the oscillator start/stop bit in the 32kHz oscillator control register.
- putting the EEPROM on-chip memory in stand-by mode by setting the E2OFF bit in EEPROM Control Register to one.

The LCD Driver peripheral is automatically switched-off by the STOP instruction when the 32kHz oscillator operation is not selected.

When the hardware activated watchdog is selected or the software watchdog enabled, the STOP instruction is deactivated and any attempt to execute the STOP instruction will cause an execution of a WAIT instruction.

If all the interrupt sources are disabled (including NMI if GEN="0"), the restart of the MCU can only be done by a $\overline{\text{RESET}}$ activation. The WAIT and STOP instructions are not executed if an enabled interrupt request is pending.

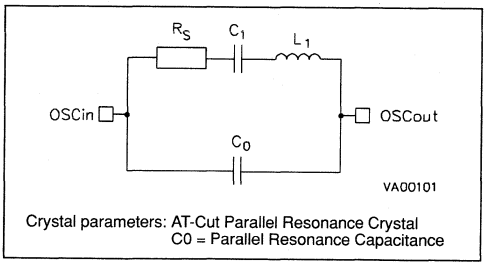
ON-CHIP CLOCK OSCILLATOR

The internal oscillator circuit is designed to require a minimum of external components. A crystal, a ceramic resonator, or an external signal (provided to the OSCin pin) may be used to generate a system clock with various stability/cost tradeoffs. The different clock generator options connection methods are shown in Figure 22.

One machine cycle takes 13 oscillator pulses; 12 clock pulses are needed to increment the PC while and additional 13th pulse is needed to stabilize the internal latches during memory addressing. This means that with a clock frequency of 8MHz the machine cycle is 1.625µs.

The crystal oscillator start-up time is a function of many variables: crystal parameters (especially R_S), oscillator load capacitance (CL), IC parameters, ambient temperature, supply voltage. It must be observed that the crystal or ceramic leads and circuit connections must be as short as possible. Typical values for CL1, CL2 are 15-22pF for a 4/8MHz crystal. The oscillator output frequency is internally divided by 13 to produce the machine cycle and by 12 to produce the Timer, the Watchdog and the A/D peripheral clock. A machine cycle is the smallest unit needed to execute any operation (i.e., increment the program counter). An instruction may need two, four, or five machine cycles to be executed.

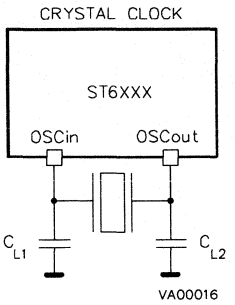
Figure 21. Crystal Parameters

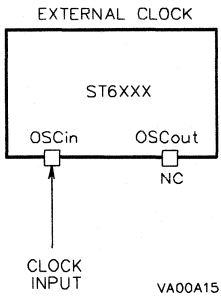


Crystal parameters: AT-Cut Parallel Resonance Crystal
C0 = Parallel Resonance Capacitance

ON-CHIP CLOCK OSCILLATOR (Continued)

Figure 22. Oscillator Connection





CL1 = CL2 = 12 to 22pF for a 4/8MHz crystal (typical)

INPUT/OUTPUT PORTS

The ST6240 microcontroller has 16 Input/Output lines that can be individually programmed either in the input mode or the output mode with the following options that can be selected by software:

- Input without pull-up and without interrupt
- Input with pull-up and with interrupt
- Input with pull-up without interrupt
- Analog inputs (PA0-PA7, PB0-PB3)
- SPI control signals (PB5-PB7)
- Push-pull output
- Standard Open drain output
- 20mA Open drain output (PB4-PB7)

The lines are organized in two ports (port A,B).

Each port occupies 3 registers in the data space. Each bit of these registers is associated with a particular line (for instance, the bits 0 of the Port A Data, Direction and Option registers are associated with the PA0 line of Port A).

The two DATA registers (DRA, DRB), are used to read the voltage level values of the lines programmed in the input mode, or to write the logic value of the signal to be output on the lines configured in the output mode. The port data registers can be read to get the effective logic levels of the pins, but they can be also written by the user software, in conjunction with the related option registers, to select the different input mode options.

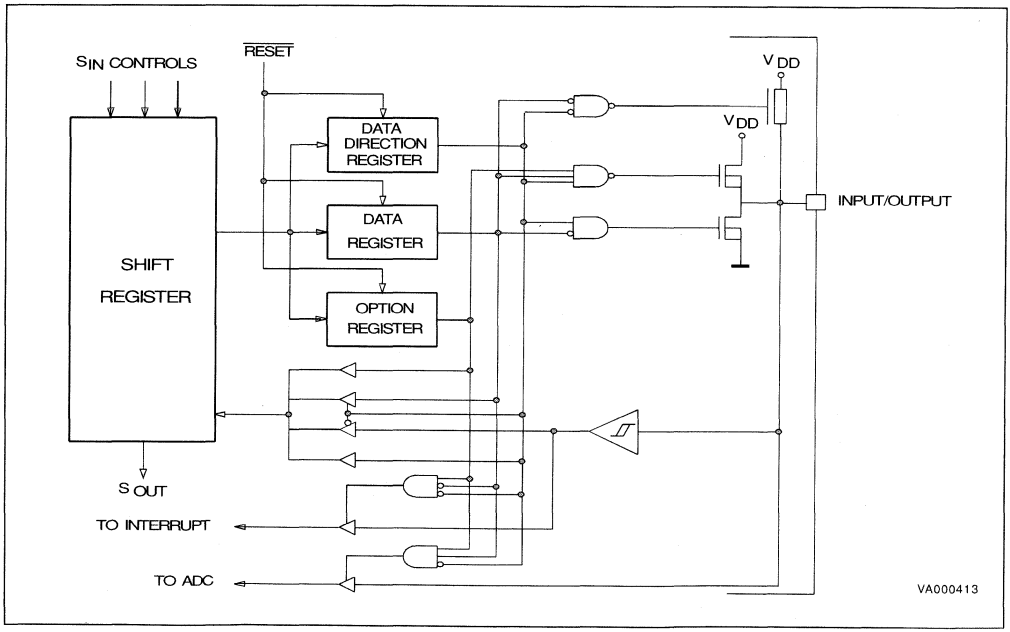
Single-bit operations on I/O registers are possible but care is necessary because reading in input mode is done from I/O pins while writing will directly affect the Port data register causing an undesired change of the input configuration.

The two Data Direction registers (DDRA, DDRB) allow the selection of the data direction of each pin (input or output).

The two Option registers (ORPA, ORPB) are used to select the different port options available both in input and in output mode.

All the I/O registers can be read or written as any other RAM location of the data space, so no extra RAM cell is needed for port data storing and manipulation. During the initialization of the MCU, all the I/O registers are cleared and the input mode with pull-up/no-interrupt is selected on all the pins, thus avoiding pin conflicts.

Figure 23. I/O Port Block Diagram



INPUT/OUTPUT PORTS (Continued)

I/O Pin Programming

Each pin can be individually programmed as input or output with different input and output configurations.

This is achieved by writing to the relevant bit in the data (DR), data direction register (DDR) and option registers (OR). Table 8 shows all the port configurations that can be selected by user software.

Input Option Description

Pull-up, High Impedance Option. All the input lines can be individually programmed with or without an internal pull-up according to the codes programmed in the OR and DR registers . If the pull-up option is not selected, the input pin is in the high-impedance state.

Interrupt Option. All the input lines can be individually connected by software to the interrupt lines of the ST62xx core according to the codes programmed in the OR and DR registers. The pins of Port A and B are "ORed" and are connected to the interrupt associated to the vector #2. The interrupt modes (falling edge sensitive, rising edge sensitive) can be selected by software for each port by programming the IOR register.

Analog Input Option. The twelve PA0-PA7, PB0-PB3 pins can be configured to be analog inputs according to the codes programmed in the OR and DR registers. These analog inputs are connected to the on-chip 8-bit Analog to Digital Converter. *ONLY ONE* pin should be programmed as analog input at a time, otherwise the selected inputs will be shorted.

Figure 24. I/O Port Data Registers

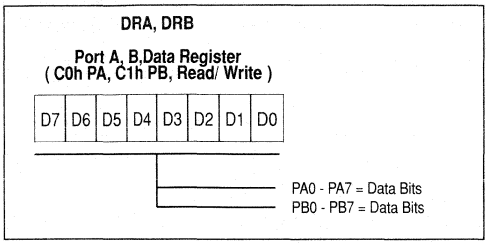


Figure 25. I/O Port Data Direction Registers

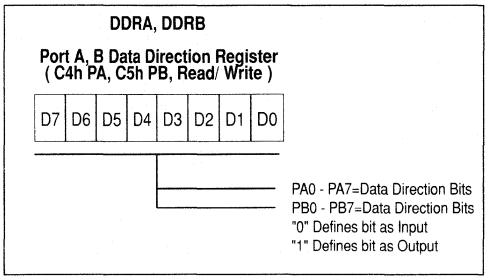
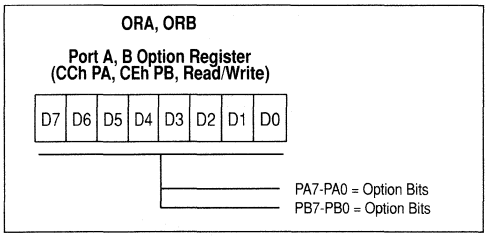


Figure 26. I/O Port Option Registers



Note: For complete coding explanation refer to Table 8.

Table 8. I/O Port Options Selection

DDR	OR	DR	MODE	OPTION
0	0	0	Input	With pull-up, no interrupt (Reset state)
0	0	1	Input	No pull-up, no interrupt
0	1	0	Input	With pull-up, with interrupt
0	1	1	Input	No pull-up, no interrupt (for PB4-PB7).
			Input	Analog input (for PA0-PA7, PB0-PB3)
1	0	X	Output	Open-drain output (20mA sink current for PB4-PB7)
1	1	X	Output	Push-pull output (20mA sink current for PB4-PB7)

Note: X. Means don't care.

INPUT/OUTPUT PORTS (Continued)

SPI alternate function Option. The I/O pins PB5-PB7 are also used by serial peripheral interface SPI. PB5 is connected with the SPI clock input SCL, PB6 is connected with the SPI data input SIN and PB7 is connected with the SPI data output SOUT.

For serial input operation PB5 and PB6 have to be programmed as inputs. For serial output operation PB7 has to be programmed as open-drain output (DDR = "1", OPR = "0"). In this operating mode the output of the SPI shift register instead of the port data register is connected to the port buffer. When PB7 is programmed as push-pull output (DDR = "1", OPR = "1"), the port data register is connected to the port buffer. When the SPI peripheral is not used PB5-PB7 can be used as general purpose I/O lines (provided that PB7 is not selected to be open-drain in output mode).

Notes:

Switching the I/O ports from one state to another should be done in a way that no unwanted side effects can happen. The recommended safe transitions are shown below. All other transitions are risky and should be avoided during change of operation mode as it is most likely that there will be an unwanted side-effect such as interrupt generation or two pins shorted together by the analog input lines.

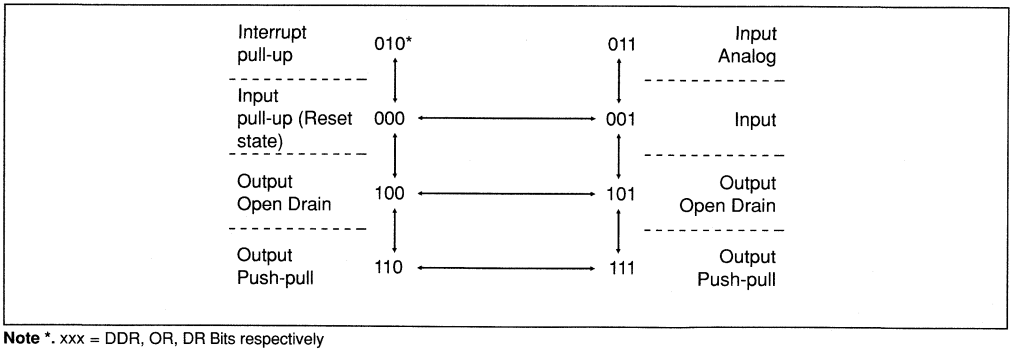
Single bit SET and RES instructions should be used very carefully with Port A and B data registers because these instructions make an implicit read and write back of the whole addressed register byte. In port input mode however data register address reads from input pins, not from data register latches and data register information in input mode is used to set characteristics of the input pin (interrupt, pull-up, analog input), therefore these characteristics may be unintentionally reprogrammed depending on the state of input pins. As general rule is better to use SET and RES instructions on data register only when the whole port is in output mode. If input or mixed configuration is needed it is recommended to keep a copy of the data register in RAM. On this copy it is possible to use single bit instructions, then the copy register could be written into the port data register.

```
SET     bit, datacopy
LD      a, datacopy
LD      DRA, a
```

The WAIT and STOP instructions allow the ST62xx to be used in situations where low power consumption is needed. The lowest power consumption is achieved by configuring I/Os in input mode with well-defined logic levels.

The user has to take care not to switch outputs with heavy loads during the conversion of one of the analog inputs in order to avoid any disturbance in the measurement.

Figure 27. I/O Port StateTransition Diagram for Safe Transitions



Note *. xxx = DDR, OR, DR Bits respectively

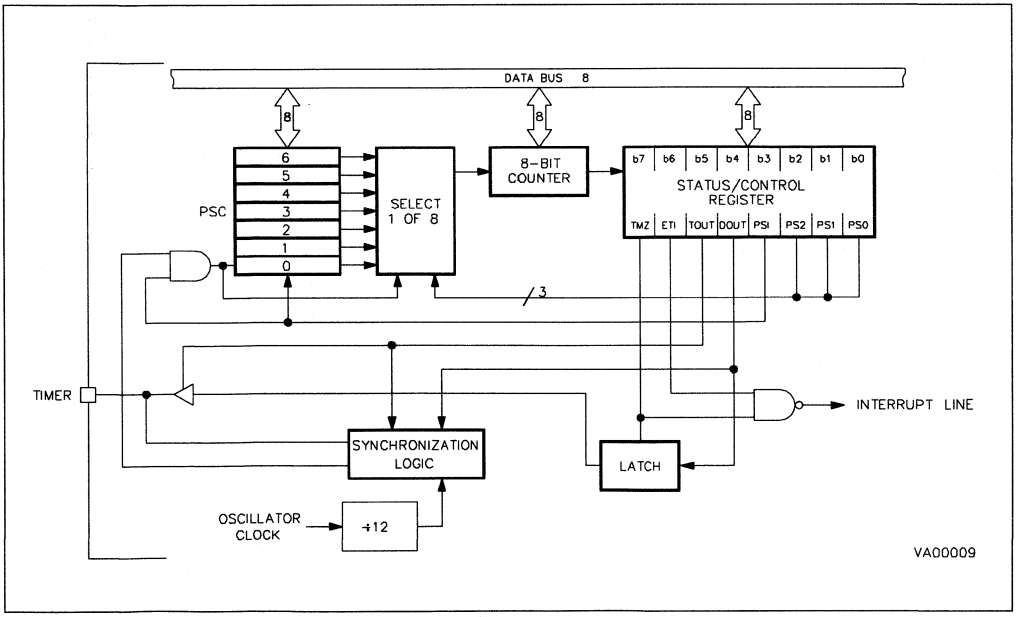
TIMERS

The ST6240 offers two on-chip Timer peripherals named Timer 1 and Timer 2. Each of these timers consists of an 8-bit counter with a 7-bit programmable prescaler, thus giving a maximum count of 2^{15}, and control logic that allows configuring the peripheral in three operating modes. Figure 28 shows the Timer block diagram. Timer 1 only has the external TIMER pin available for the user. The content of the 8-bit counter can be read/written in the Timer/Counter register TCR which is addressed in the data space as a RAM location at addresses D3h or D6h. The state of the 7-bit prescaler can be read in the PSC register at addresses D2h or D5h. The control logic device is managed in the TSCR register (addresses D4h or D7h) as described in the following paragraphs.

The 8-bit counter is decrement by the output (rising edge) coming from the 7-bit prescaler and can be loaded and read under program control. When it decrements to zero then the TMZ (Timer Zero) bit in the TSCR is set to one. If the ETI (Enable Timer Interrupt) bit in the TSCR is also set to one an interrupt request, associated to interrupt vector #3, is generated. The interrupt service routine then would determine which timer reached the end of count by poling the TMZ bits. The Timer interrupt can be used to exit the MCU from the WAIT mode.

The prescaler input can be the oscillator frequency divided by 12 or an external clock at TIMER pin (only Timer 1). The prescaler decrements on the rising edge. Depending on the division factor programmed by PS2, PS1 and PS0 bits in the TSCR, the clock input of the timer/counter register is multiplexed to different sources. On division factor 1, the clock input of the prescaler is also that of timer/counter; on factor 2, bit 0 of prescaler register is connected to the clock input of TCR. This bit changes its state with the half frequency of prescaler clock input. On factor 4, bit 1 of PSC is connected to clock input of TCR, and so on. The prescaler initialize bit (PSI) in the TSCR register must be set to one to allow the prescaler (and hence the counter) to start. If it is cleared to zero then all of the prescaler bits are set to one and the counter is inhibited from counting. The prescaler can be given any value between 0 and 7Fh by writing to addresses D2h or D5h, if bit PSI in the TSCR register is set to one. The tap of the prescaler is selected using the PS2,PS1,PS0 bits in the control register. Figure 28 shows the Timer working principle.

Figure 28. Timer Peripheral Block Diagram



TIMERS (Continued)

Timer Operating Modes

There are 3 operating modes of the Timer peripheral. They are selected by the bits TOUT and DOUT (see TSCR register). These three modes correspond to the two clock frequencies that can be connected on the 7-bit prescaler ($f_{OSC}/12$ or TIMER pin signal) and to the output mode. For this reason, only Timer 1 has all three modes, while Timer 2, which does not have a dedicated TIMER pin, must be programmed in Output Mode only.

Clock Input Mode (TOUT = "0", DOUT = "0"). In this mode the TIMER pin is an input and the prescaler is decremented on rising edge. The maximum input frequency that can be applied to the external pin in this mode is 1/8 of the oscillator frequency. This operating mode is not available on Timer 2.

Gated Mode (TOUT = "0", DOUT = "1"). In this mode the prescaler is decremented by the Timer clock input (oscillator divided by 12) but ONLY when the signal at TIMER pin is held high (giving a pulse width measurement potential). This mode is selected by the TOUT bit in TSCR register cleared to "0" (i.e. as input) and DOUT bit set to "1". This operating mode is not available on Timer 2.

Output Mode (TOUT = "1", DOUT = data out). The TIMER pin is connected to the DOUT latch. Therefore the timer prescaler is clocked by the prescaler clock input ($f_{OSC}/12$).

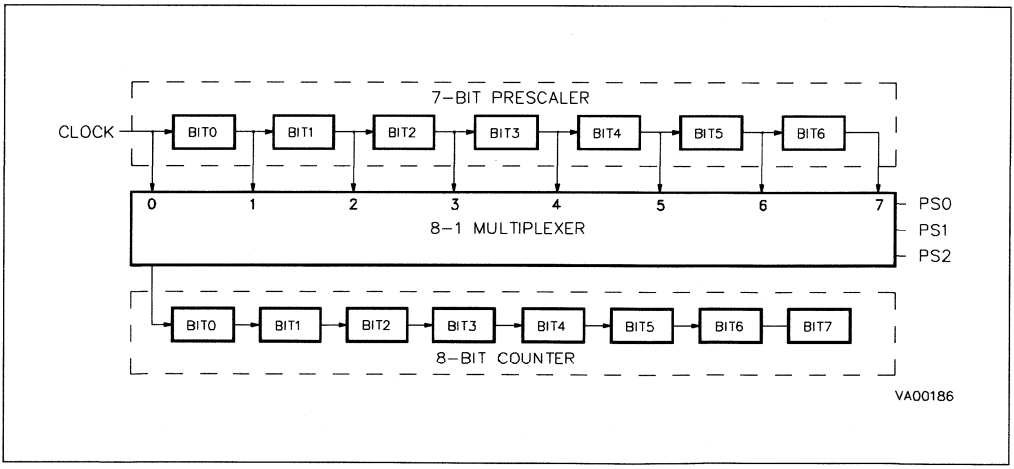
The user can select the desired prescaler division ratio through the PS2, PS1, PS0 bits. When TCR count reaches 0, it sets the TMZ bit in the TSCR. The TMZ bit can be tested under program control to perform a timer function whenever it goes high. The low-to-high TMZ bit transition is used to latch the DOUT bit of the TSCR and pass it to TIMER pin. This operating mode allows external signal generation on the TIMER pin. This is the only operating Mode allowed on Timer 2.

Table 9. Timer Operating Modes

TOUT	DOUT	Timer Pin	Timer Function
0	0	Input	Event Counter (1)
0	1	Input	Input Gated (1)
1	0	Output	Output
1	1	Output	Output

Note 1. Not allowed on Timer 2

Figure 29. Timer Working Principle



TIMERS (Continued)

Timer Interrupt

When the counter register decrements to zero and the software controlled ETI (Enable Timer Interrupt) bit is set to one then an interrupt request associated to interrupt vector #3 is generated. When the counter decrements to zero also the TMZ bit in the TSCR register is set to one.

Since only one interrupt vector is available for the two timers (ORed also with 32kHz oscillator interrupt), the interrupt service routine should determine from which source the interrupt came by polling the TMZ bits (and the OSCEOC bit of the 32kHz oscillator control register).

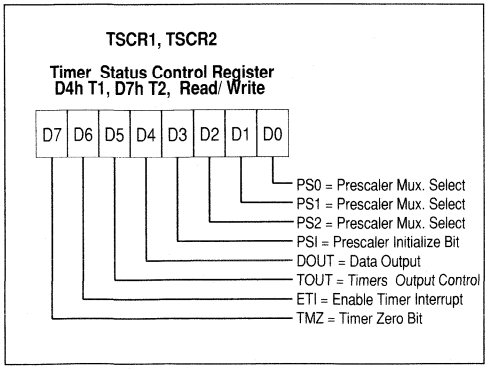
Notes:

TMZ is set when the counter reaches 00h; however, it may be set by writing 00h in the TCR register or setting bit 7 of the TSCR register. TMZ bit must be cleared by user software when servicing the timer interrupt to avoid undesired interrupts when leaving the interrupt service routine. After reset, the 8-bit counter register is loaded to FFh while the 7-bit prescaler is loaded to 7Fh , and the TSCR register is cleared which means that timer is stopped (PSI="0") and the timer interrupt is disabled.

If the Timer is programmed in output mode, DOUT bit is transferred to the TIMER pin when TMZ is set to one (by software or due to counter decrement). When TMZ is high, the latch is transparent and DOUT is copied to the timer pin. When TMZ goes low, DOUT is latched.

A write to the TCR register will predominate over the 8-bit counter decrement to 00h function, i.e. if a write and a TCR register decrement to 00h occur simultaneously, the write will take precedence, and the TMZ bit is not set until the 8-bit counter reaches 00h again. The values of the TCR and the PSC registers can be read accurately at any time.

Figure 30. Timer Status Control Register



TMZ. Low-to-high transition indicates that the timer count register has decremented to zero. This bit must be cleared by user software before starting with a new count.

ETI. This bit, when set, enables the timer interrupt request (vector #3). If ETI="0" the timer interrupt is disabled. If ETI="1" and TMZ="1" an interrupt request is generated.

TOUT. When low, this bit selects the input mode for the TIMER pin. When high the output mode is selected.

DOUT. Data sent to the timer output when TMZ is set high (output mode only). Input mode selection (input mode only). This bit is meaningless for Timer 2.

PSI. Used to initialize the prescaler and inhibit its counting. When PSI="0" the prescaler is set to 7Fh and the counter is inhibited. When PSI="1" the prescaler is enabled to count downwards. As long as PSI="0" both counter and prescaler are not running.

PS2, PS1, PS0. These bits select the division ratio of the prescaler register.

Table 10. Prescaler Division Factors

PS2	PS1	PS0	Divided by	PS2	PS1	PS0	Divided by
0	0	0	1	1	0	0	16
0	0	1	2	1	0	1	32
0	1	0	4	1	1	0	64
0	1	1	8	1	1	1	128

Figure 31. Timer Counter Register

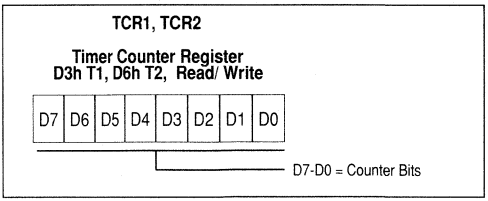
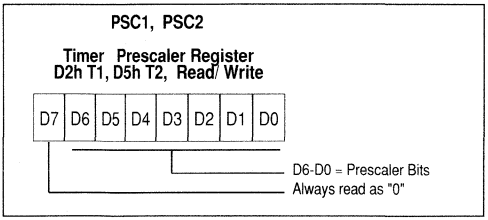


Figure 32. Prescaler Register



DIGITAL WATCHDOG

The digital Watchdog of the ST62 device consists of a down counter that can be used to provide a controlled recovery from a software upset.

The ST6240 watchdog has two watchdog options, the software activated watchdog/timer and the hardware activated watchdog function. The user can select one of the two options by connecting the pin WDON to V_{DD} (software watchdog) or V_{SS} (hardware watchdog). If the pin is kept non connected, an internal pull-up resistance selects the software watchdog option.

Hardware Watchdog Function

The hardware activated digital watchdog function consists of a down counter that is automatically initialized after reset so that this function does not need to be activated by the user program. As the watchdog function is always activated this down counter cannot be used as a timer. The watchdog uses one data space register (DWDR location D8h). The watchdog register is set to FFh on reset and immediately starts to count down, requiring no software start. Similarly the hardware activated watchdog cannot be stopped or delayed by software.

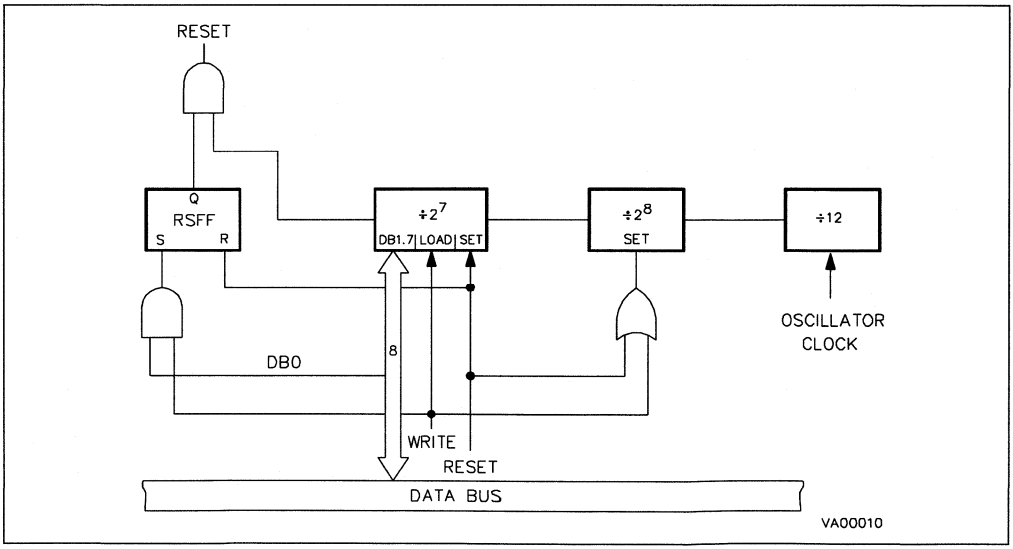
The watchdog time can be programmed using the 6 Most Significant bits in the watchdog register, this gives the possibility to generate a reset in a time between 3072 to 196608 clock cycles in 64 possible steps (with a clock frequency of 8MHz, this means from 384µs to 24.576ms). The reset is prevented if the register is reloaded with the desired value before bits 2-7 decrement from all zeros to all ones. The check time can be set differently for different routines within the general program. The presence of the hardware watchdog deactivates the STOP instruction and a WAIT instruction is automatically executed instead of a STOP. Bit 1 of the watchdog register (set to one at reset) can be used to generate a software reset if cleared to zero.

Software Watchdog

The software activated digital watchdog consists of a down counter that can be used to provide a controlled recovery from a software upset. The watchdog uses one data space register (DWDR location D8h). The watchdog register is set to FEh after reset and the watchdog function is disabled. The watchdog time can be programmed using the 6 Most Significant Bits in the Watchdog register. The check time can be set differently for different routines within the general program.

After a reset the software Watchdog is in the off-state. The watchdog should be activated inside the Reset restart routine by writing a "1" in watchdog timer register bit 0. Bit one of this register must be

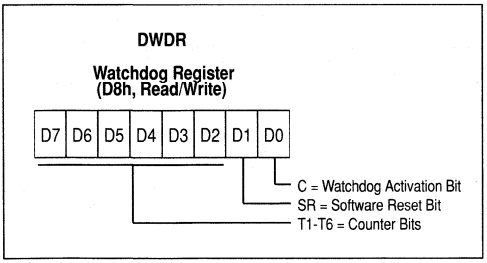
Figure 33. Digital Watchdog Block Diagram



DIGITAL WATCHDOG (Continued)

set to one before programming bit zero as otherwise a reset will be immediately generated when bit 0 is set. This allows the user to generate a reset by software (bit 0 = "1", bit 1 = "0"). Once bit 0 is set, it can not be cleared by software without generating a Reset. The delay time is defined by programming bits 2-7 of the watchdog register. Bit 7 is the Least Significant Bit while bit 2 is the MSB. This gives the possibility to generate a reset in a time between 3072 to 196608 clock cycles in 64 possible steps: (With a clock frequency of 8MHz this means from 384µs to 24.576ms). The reset is prevented if the register is reloaded with the desired value before bits 2-7 decrement from all zeros to all ones. If the watchdog is active the STOP instruction is deactivated and a WAIT instruction is automatically executed instead of a STOP. If bit 0 of the watchdog register is never set to one then bits 1-7 of the register can be used as a simple 7-bit counter which is decrement every 3072 clock cycles.

Figure 35. Watchdog Register

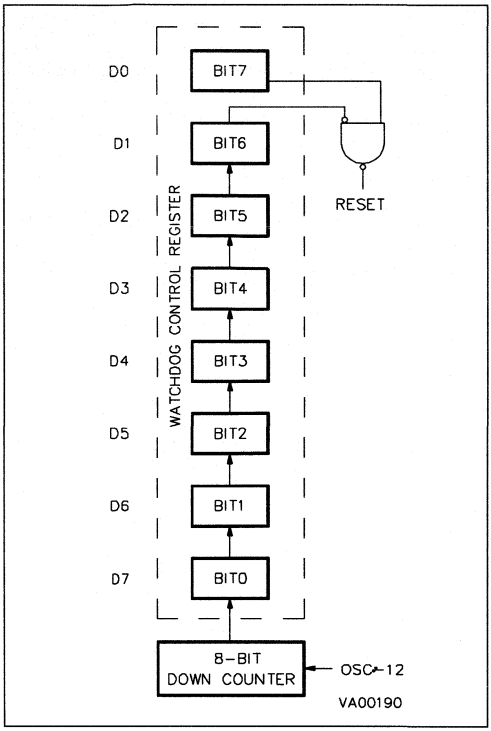


C. This is the watchdog activation bit, that, if set to one, will activate the watchdog function. When cleared to zero it allows the use of the counter as a 7-bit timer. If the user selects the hardware function this bit is automatically set on reset and the user cannot change its state. When the software function is selected, this bit is cleared on reset.

SR. This bit is set to one during the reset and will generate a software reset if cleared to zero. When C = "0" (watchdog disabled software option) it is the MSB of the 7-bit timer.

T1-T6. These are the watchdog counter bits. It should be noted that D7 (T1) is the LSB of the counter and D2 (T6) is the MSB of the counter. These bits are in the opposite order to normal.

Figure 34. Watchdog Working Principle



DIGITAL WATCHDOG (Continued)

Application note:

The hardware activation function is very useful when the external circuitry may inject noise on the reset pin, where there is an unstable supply voltage, or RF influence or other similar phenomena.

If the Watchdog software activation is selected and the Watchdog is not used during power-on reset external noise may cause the undesired activation of the Watchdog with a generation of an unexpected reset. To avoid this risk, two additional instructions, that check the state of the watchdog and eventually reset the chip are needed within the first 27 instructions, after the reset. These instructions are:

```
jrx 0, WD, #+3
ldi WD, 0FDH
```

These instructions should be executed at the very beginning of the customer program.

If the Watchdog is used (both hardware or software activated), during power-on reset the Watchdog register may be set to a low value, that could give a reset after 28 instructions earliest. To avoid undesired resets, the Watchdog must be set to the desired value within the first 27 instructions, the best is to put at the very beginning.

Alternatively the normal legal state can be checked with the following short routine:

```
ldi a, 0FEH
and a, WD
cpi a, 0FEH
jrz #+3
ldi WD, 0FDH
```

This sequence is recommended for security applications, where possible stack confusion error loops must be avoided and the Watchdog must only be refreshed after extensive checks.

8-BIT A/D CONVERTER

The A/D converter of ST6240 is an 8-bit analog to digital converter with up to 12 analog inputs (as alternate functions of I/O lines PA0-PA7, PB0-PB3) offering 8-bit resolution with total accuracy ±2 LSB and a typical conversion time of 70µs (clock frequency of 8MHz).

The A/D peripheral converts the input voltage by a process of successive approximations using a clock frequency derived from the oscillator with a division factor of twelve. With an oscillator clock frequency less than 1.2MHz, the A/D converter accuracy is decreased.

The selection of the pin signal that has to be converted is done by configuring the related I/O line as analog input through the I/O ports option and data registers (refer to I/O ports description for additional information). Only one I/O line must be configured as analog input at a time. The user must avoid the situation in which more than one I/O pin is selected to be analog input to avoid malfunction of the ST62xx.

The ADC uses two registers in the data space: the ADC data conversion register which stores the conversion result and the ADC control register used to program the ADC functions.

A conversion is started by writing a "1" to the Start bit (STA) in the ADC control register. This automatically clears (resets to "0") the End Of Conversion Bit (EOC). When a conversion has been finished this EOC bit is automatically set to "1" in order to flag that conversion is complete and that the data in the ADC data conversion register is valid. Each conversion has to be separately initiated by writing to the STA bit.

The STA bit is continually being scanned so that if the user sets it to "1" while a previous conversion is in progress then a new conversion is started before the previous one has been completed.The

8-BIT A/D CONVERTER(Continued)

start bit (STA) is a write only bit, any attempt to read it will show a logical "0".

The A/D converter has a maskable interrupt associated to the end of conversion. This interrupt is associated to the interrupt vector #4 and occurs when the EOC bit is set, i.e. when a conversion is completed. The interrupt is masked using the EAI (interrupt mask) bit in the control register.

The power consumption of the device can be reduced by turning off the ADC peripheral. That is achieved when the PDS bit in the ADC control register is cleared to "0". If PDS="1", the A/D is supplied and enabled for conversion. This bit must be set at least one instruction before the beginning of the conversion to allow the stabilization of the A/D converter. *This action is needed also before entering the STOP instruction as the A/D comparator is not automatically disabled by the STOP mode*

Figure 37. A/D Converter Block Diagram

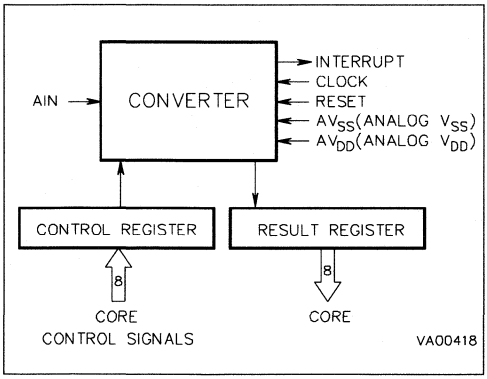
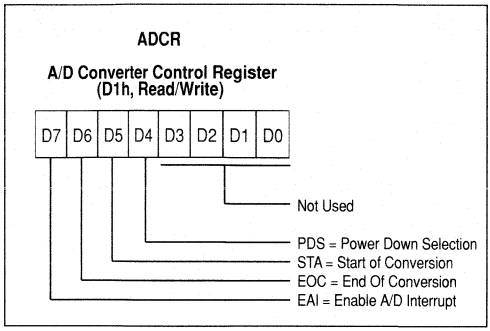


Figure 36. A/D Converter Control Register



During reset any conversion in progress is stopped, the control register is reset to all zeros and the A/D interrupt is masked (EAI=0).

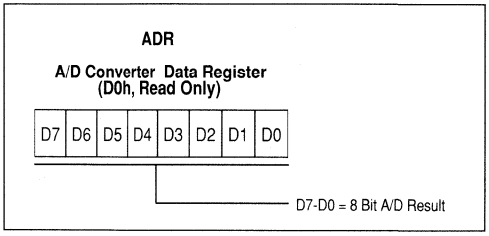
EAI. If this bit is set to one the A/D interrupt (vector #4) is enabled, when EAI=0 the interrupt is disabled.

EOC. *Read Only*;This read only bit indicates when a conversion has been completed. This bit is automatically reset to zero when the STA bit is written. If the user is using the interrupt option then this bit can be used as an interrupt pending bit. Data in the data conversion register are valid only when this bit is set to one.

STA. *Write Only*; Writing a "1" in this bit will start a conversion on the selected channel and automatically reset to zero the EOC bit. If the bit is set again when a conversion is in progress, the present conversion is stopped and a new one will take place. This bit is write only, any attempt to read it will show a logical zero.

PDS. This bit activates the A/D converter if set to

Figure 38. A/D Converter Data Register



1. Writing a zero into this bit will put the ADC in power down mode (idle mode).

D3-D0. Not used

D7-D0. *Read Only*;These are the conversion result bits; the register is read only and stores the result

8-BIT A/D CONVERTER(Continued)

of the last conversion. The contents of this register are valid only when EOC bit in the ADCR register is set to one (end-of-conversion).

Notes:

The ST62xx A/D converter does not feature a sample and hold. The analog voltage to be measured should therefore be stable during the conversion time. Variation should not exceed ±1/2 LSB for the best accuracy in measurement.

Since the ADC is on the same chip as the microprocessor the user should not switch heavily loaded output signals during conversion if high precision is needed. This is because such switching will affect the supply voltages which are used for comparisons.

A low pass filter can be used at the analog input pins to reduce input voltage variation during the conversion. For true 8 bit conversions the impedance of the analog voltage sources should be less than 30kΩ while the impedance of the reference voltage should not exceed 2kΩ.

The accuracy of the conversion depends on the quality of the power supply voltages (V_{DD} and V_{SS}). The user must specially take care of applying regulated reference voltage on the V_{DD} and V_{SS} pins (the variation of the power supply voltage must be inferior to 5V/ms).

The converter can resolve the input voltage with an resolution of:

$$\frac{V_{DD} - V_{SS}}{256}$$

So if operating with a supply voltage of 5V the resolution is about 20mV.

The Input voltage (Ain) which has to be converted must be constant for 1µs before conversion and remain constant during the conversion.

The resolution of the conversion can be improved if the power supply voltage (V_{DD}) of the microcontroller becomes lower.

In order to optimize the resolution of the conversion, the user can configure the microcontroller in the WAIT mode because this mode allows the minimization of the noise disturbances and the variations of the power supply voltages due to the switching of the outputs. Nevertheless, it must be take care of executing the WAIT instruction as soon as possible after the beginning of the conversion because the execution of the WAIT instruction may provide a small variation of the V_{DD} voltage (the negative effect of this variation is minimized at the beginning of the conversion because the latter is less sensitive than the end of the conversion when the less significant bits are determined).

The best configuration from a accuracy point of view is the WAIT mode with the Timer and LCD driver stopped. Indeed, only the ADC peripheral and the oscillator are still working. The MCU has to be wake-up from the WAIT mode by the interrupt of the ADC peripheral at the end of the conversion. It must be noticed that the wake-up of the microcontroller could be done also with the interrupt of the TIMER, but in this case, the Timer is working and some noise could disturb the converter in terms of accuracy.

POWER SUPPLY SUPERVISOR DEVICE (PSS)

The Power Supply Supervisor device, described in the Figure 39, permits supervising the crossing of the PSS pin voltage (VPSS) through a programmable voltage (mxV_{DD}/n), where n and m can be chosen by software. This device includes:

- An internal comparator which is connected to the internal INT line to make an interrupt request to the Core.
- 2 resistive voltage dividers that are, respectively, supplied by the PSS pin and the V_{DD} pin. These two voltage dividers are both connected to the two inputs of the internal comparator. They consist of 13 identical resistors. It is possible to select by software 5 voltage rates on the PSS divider ($nxVPSS/13$) and 4 voltage rates on the V_{DD} divider ($mxV_{DD}/13$). The n and m values can be chosen by software. These two voltage dividers are disconnected in STOP mode, and when the PSS device is OFF.
- An internal device that allows the detection with an hysteresis of $V_{DD}/13$.

The PSS device is supplied by an internal connection to V_{DD} supply. The following paragraphs describe the operating mode of the PSS device and the PSS register that permits control over the PSS device. The PSS device is switched off as soon as the Core executes the STOP instruction, but continues to work in the WAIT mode.

Figure 40. PSS Device Operating Modes Description

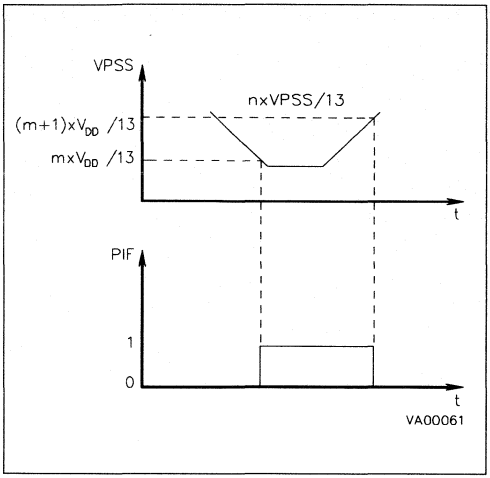
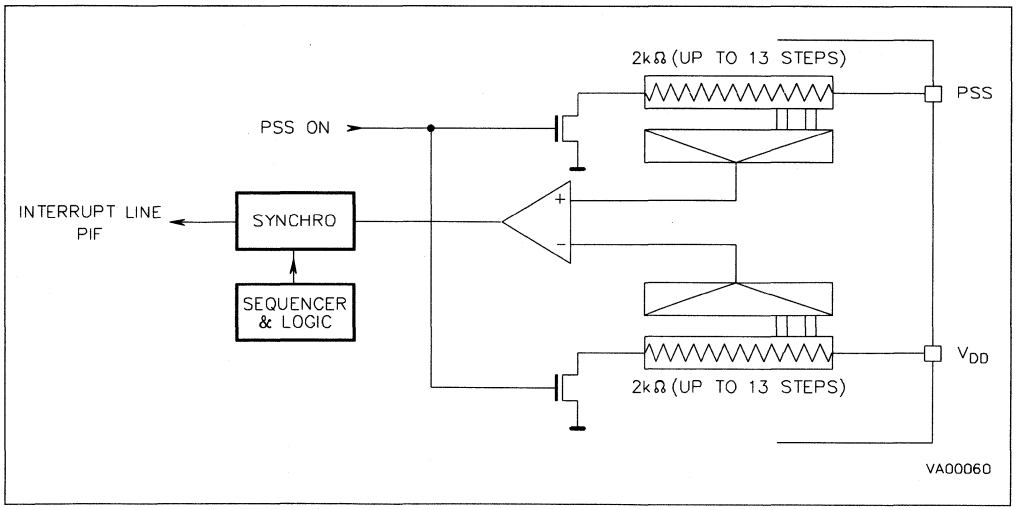


Figure 39. PSS Device Block Diagram



POWER SUPPLY SUPERVISOR(Continued)

PSS Operating Mode Description

The resistive voltage divider connected to the PSS pin provides the internal comparator with the $nxV_{PSS}/13$ voltage. The resistive voltage divider connected to the V_{DD} pin provides the internal comparator with the $mxV_{DD}/13$ voltage. The n and m values are selected with the PSS register. It must be observed that the n and m values must be selected, taking into consideration the following electrical constraints:

$0.5V < nxV_{PSS}/13$ at detection $< V_{DD} - 2V$

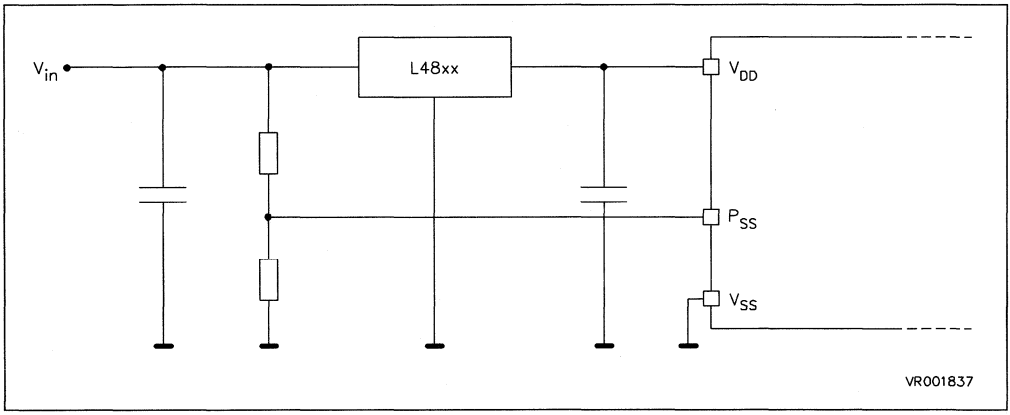
$0.5V < mxV_{DD}/13$ at detection $< V_{DD} - 2\ V$

we must also have:

$$\frac{m}{n} V_{DD} \leq V_{PSS} \leq V_{DD}$$

The PIF bit is the interrupt request flag of the PSS device. This bit follows PSS comparator output.

Figure 41. Typical application using the PSS

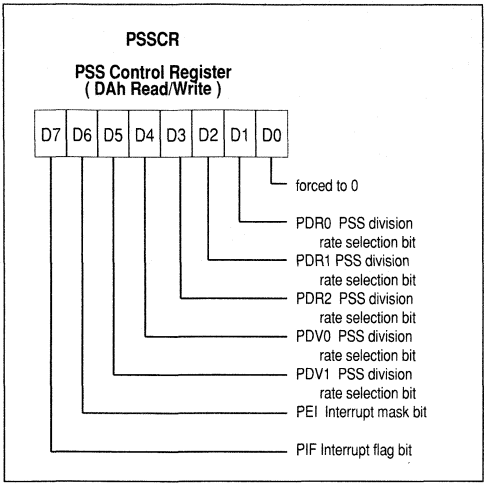


POWER SUPPLY SUPERVISOR(Continued)

PSS Register

The PSS register permits control over the PSS device. The register can be addressed in the data space as a RAM location at DAh. This register is cleared after Reset.

Figure 42. PSS Status Control Register



PIF. This bit is the interrupt flag. This bit is set (resp. cleared) as soon as the equality between nxVPSS and $(m+1)xV_{DD}/13$ (resp. $mxV_{DD}/13$) occurs.

PEI. This bit is the authorization bit of the interrupt request:

- If PEI is set, the interrupt request can reach the Core.
- If PEI is cleared, the interrupt request cannot reach the Core.

PDV1, PDV0. The PDV1/0 bits are used to select the rate of division of the V_{DD} voltage ($mxV_{DD}/13$ or $(m+1)xV_{DD}/13$, according to the hysteresis).

Table 12. V_{DD} Voltage division rate selection bits

PDV1	PDV0	$mxV_{DD}/13$	$(m+1)xV_{DD}/13$
0	0	$3xV_{DD}/13$	$4xV_{DD}/13$
0	1	$5xV_{DD}/13$	$6xV_{DD}/13$
1	0	$6xV_{DD}/13$	$7xV_{DD}/13$
1	1	$7xV_{DD}/13$	$8xV_{DD}/13$

PDR2, PDR1, PDR0. The PDR2/1/0 bits are used to inhibit the PSS device and to select the division rate of the PSS voltage (nxVPSS/13).

The PSS comparator output is valid 8 cycle times after the programming of the PDR2/1/0 bits. It is forced to zero in the meantime.

Table 11. P_{SS} Voltage division rate selection bits

PDR2	PDR1	PDR0	PSS State	nxVPSS/13
0	0	0	IDLE	
0	0	1	BUSY	4xVPSS/13
0	1	0	BUSY	5xVPSS/13
0	1	1	BUSY	6xVPSS/13
1	0	0	BUSY	7xVPSS/13
1	0	1	BUSY	VPSS

32kHz STAND-BY OSCILLATOR

The 32kHz stand-by oscillator allows the ST6240 to generate real time interrupts and to supply the clock to the LCD driver. This enables the ST6240 to provide real time functions with the LCD display capability and lower power consumption. Figure 43 shows the 32kHz oscillator block diagram.

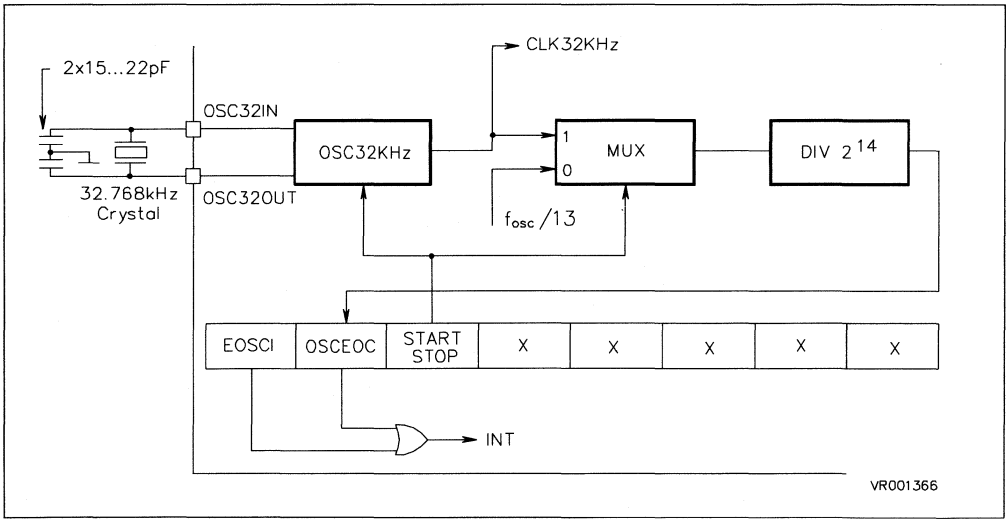
A 32.768kHz quartz crystal must be connected to the OSC32in and OSC32out pins to perform the real time clock operation. Two external capacitors of 15-22pF each must be connected between the oscillator pins and ground. The 32kHz oscillator is managed by the dedicated status/control register located at address 0DBh .

When the 32kHz stand-by oscillator is stopped (bit 5 of the Status/Control register cleared) the divider chain is supplied with a clock signal synchronous with machine cycle ($f_{OSC}/13$), this produces an interrupt request every $13x2^{14}$ clock cycle (i.e. 26.624ms) with an 8MHz quartz crystal.

When the 32kHz stand-by oscillator is enabled (bit 5 of the Status/Control register set to one) the divider chain is directly supplied with the 32kHz oscillator clock. The 32kHz clock from the standby oscillator can also be used as the LCD clock. This allows operation of the LCD in STOP mode. The interrupt output of the 32kHz oscillator peripheral generates an interrupt request every half second (500ms). This can be used to perform a real time clock function when the MCU is in STOP mode.

This interrupt signal is "ORed" with the interrupt request signals of the two on-chip timers and connected to the low level sensitive interrupt input associated to the interrupt vector #3 (FF2h, FF3h). The interrupt request has to be cleared by user software before leaving the interrupt service routine. Discrimination between the three interrupt sources is made by polling the Status/Control registers of Timer 1 (D4h), Timer 2 (D7h) and 32kHz oscillator (DBh).

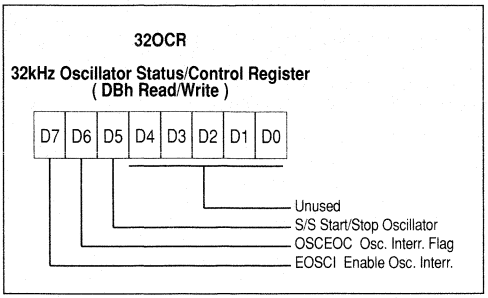
Figure 43. 32kHz Oscillator Block Diagram



32kHz STAND-BY OSCILLATOR (Continued)

32kHz Oscillator Status/Control Register

Figure 44. 32kHz Oscillator Register



EOSCI. Enable Oscillator Interrupt. This bit, when set, enables the 32kHz oscillator interrupt request.

OSCEOC. Oscillator Interrupt Flag. This bit indicates when the 32kHz oscillator has measured a 500ms elapsed time (providing a 32.768kHz quartz crystal is connected to the 32kHz oscillator dedicated pins). An interrupt request can be generated in relation to the state of EOSCI bit. This bit must be cleared by the user program before leaving the interrupt service routine.

START/STOP. Oscillator Start/Stop bit. This bit, when set, enables the 32kHz stand-by oscillator and the free running divider chain is supplied by the 32kHz oscillator signal. When this bit is cleared to zero the divider chain is supplied with the clock signal from the LCD Controller.

This register is cleared during reset.

Note:
To achieve minimum power consumption in STOP mode (no system clock), the stand-by oscillator must be switched off (real time function not available) by clearing the Start/Stop bit in the oscillator status/control register.

SERIAL PERIPHERAL INTERFACE (SPI)

The ST6240 SPI is an optimized serial synchronous interface that supports a wide range of industry standard SPI specifications. The ST6240 SPI is controlled by small and simple user software to perform serial data exchange. The serial shift clock can be implemented either by software (using the bit-set and bit-reset instructions), with the on-chip Timer 1 by externally connecting the SPI clock pin to the timer pin or by directly applying an external clock to the SPI.

The peripheral is composed by an 8-bit Data/shift Register (address DDh) and a 4-bit binary counter. The SCL, Sin and Sout SPI data and clock signals are connected to the PA5, PA6 and PA7 I/O lines. With the 3 I/O pins, the SPI can operate in the following operating modes: Software SPI, S-BUS, I^2C-bus and as a standard serial I/O (clock, data, enable). An interrupt request can be generated after eight clock pulses. Figure 46 shows the SPI block diagram.

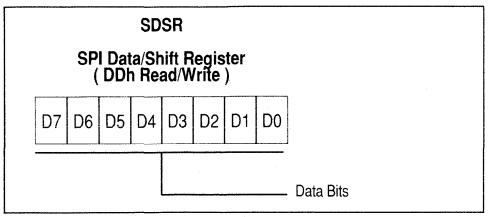
The PA5/SCL line clocks, on the falling edge, the shift register and the counter. To allow SPI operation the PA5/SCL must be programmed as input, an external clock supplied to this pin will drive the SPI peripheral (slave mode).

If PA5/SCL is programmed as output, a clock signal can be generated by software, setting and resetting the port line by software (master mode).

The SCL clock signal is the shift clock for the SPI data/shift register. The PA6/Sin pin is the serial shift input and PA7/Sout is the serial shift output. These two lines can be tied together to implement two wires protocols (I^2C-bus, etc). When data is serialized, the MSB is the first bit. PA6/Sin has to be programmed as input. For serial output operation PA7/Sout has to be programmed as open-drain output.

After 8 clock pulses (D7..D0) the output $\overline{Q4}$ of the 4-bit binary counter becomes low, disabling the clock from the counter and the data/shift register. Q4 enables the clock to generate an interrupt on the 8th clock falling edge as long as no reset of the counter (processor write into the 8-bit data/shift register) takes place. After a processor reset the interrupt is disabled. The interrupt is active when writing data in the shift register (DDh) and desactivated when writing any data in the register SPI Interrupt Disable (C2h).

Figure 45. SPI Data/Shift Register



SERIAL PERIPHERAL INTERFACE (Continued)

The generation of an interrupt to the Core provides information that new data is available (input mode) or that transmission is completed (output mode), allowing the Core to generate an acknowledge on the 9th clock pulse (I^2C-bus).

Since the SPI interrupt is connected to interrupt #1, the falling edge interrupt option should be selected by clearing to zero bit 6 of the Interrupt Option Register (IOR, C8h).

After power on reset, or after writing the data/shift register, the counter is reset to zero and the clock is enabled. In this condition the data shift register is ready for reception. No start condition has to be detected. Through the user software the Core may pull down the Sin line (Acknowledge) and slow down the SCL, as long as it is needed to carry out data from the shift register.

I^2C-bus Master-Slave, Receiver-Transmitter

When pins Sin and Sout are externally connected together it is possible to use the SPI as a receiver as well as a transmitter. With a simple software routine (by using bit-set and bit-reset on I/O line) a clock can be generated allowing I^2C-bus to work in master mode.

When implementing an I^2C-bus protocol, the start condition can be detected by setting the processor into a "wait for start" condition by simply enabling the interrupt of the PA6/Sin I/O port. This frees the processor from polling the Sin and SCL lines. After the transmission/reception the processor has to poll for the STOP condition.

In slave mode the user software can slow down the SCL clock frequency by simply putting the SCL I/O line in output open-drain mode and writing a zero into the corresponding data register bit.

As it is possible to directly read the Sin pin directly through the port register, the software can detect a difference between internal data and external data (master mode). Similar condition can be applied to the clock.

The typical speed of transmission in I^2C master or slave mode is in the range of 10kHz.

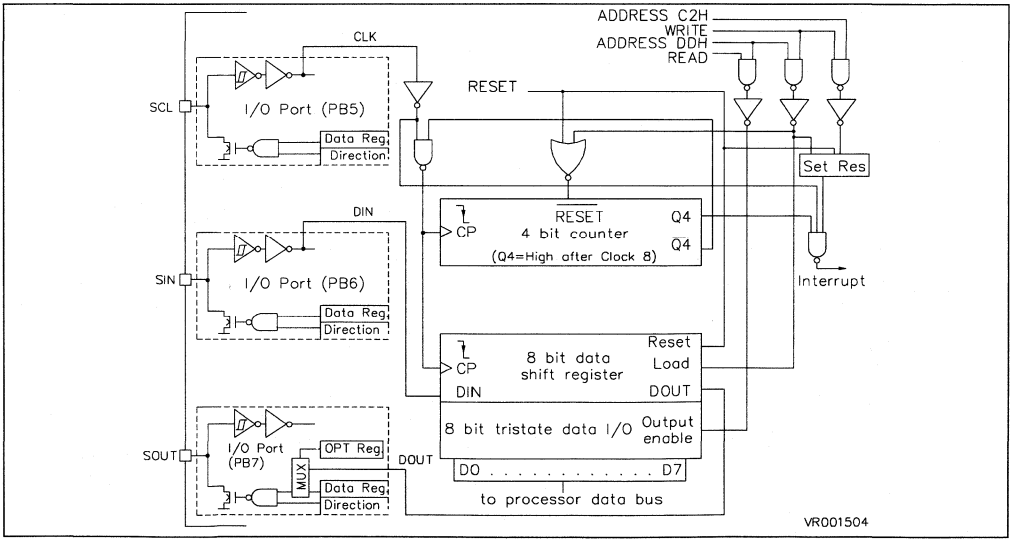
Three (Four) Wire Serial Bus

It is possible to use a single general purpose I/O pin (with the corresponding interrupt enabled) as a "chip enable" pin. SCL acts as active or passive clock pin, Sin as data in and Sout as data out (four wire bus). Sin and Sout can be connected together externally to implement three wire bus.

Note:

When the SPI is not used, the three I/O lines (Sin, SCL, Sout) can be used as normal I/O, with the following limitation: bit Sout cannot be used in open drain mode as this enables the shift register output to the port.
It is recommended, in order to avoid spurious interrupts from the SPI, to disable the SPI interrupt (the default state after reset) i.e. no write must be made to the 8-bit shift register (DDh). An explicit interrupt disable may be made in software by a dummy write to address C2h.

Figure 46. SPI Block Diagram



LCD CONTROLLER-DRIVER

The ST6240 LCD driver consists of a LCD control logic, a programmable prescaler, a 24 bytes wide dedicated LCD RAM, 45 segment and 4 common outputs. This allows a direct driving of up to 180 LCD segments.

The LCD driver is managed by the LCD Mode/Control register located at data RAM address DCh. Different display modes (1/1 duty, 1/2 duty, 1/3 duty and 1/4 duty) are available to cover a wide range of application requirements. The multiplexing display modes are software selectable by programming bits 6 and 7 of the LCD control register. Bits 0-5 are used to select the LCD drive and frame frequency (in relation to the system clock) and to switch off all segments. The LCD Driver can also be supplied by the 32kHz real-time oscillator allowing working in low power conditions and performing real time clock operation.

According to the data in the LCD RAM, the segment and the common drivers generate the segment and common signals which can directly drive an LCD panel.

The LCD control logic reads automatically the data from the LCD RAM independently and without interruption of the processor. The part of the LCD RAM that is not used for displaying can be used as normal data memory.

The scale factor of the clock prescaler can be fixed by software, therefore different frame frequencies can be defined.

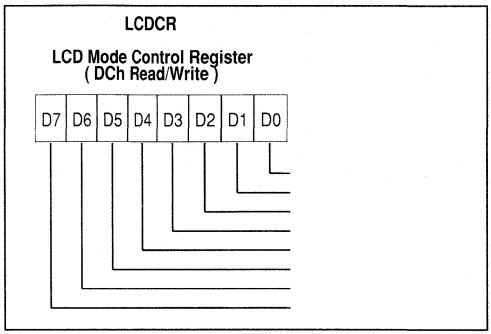
The ST6240 oscillator should operate with a 1.0486, 2.0972, 4.1943, 8.3886MHz frequency quartz crystal. This allows the associated division rates to achieve an internal reference frequency of 32.768kHz. The different division rates can be achieved by programming bits 3, 4, 5 in the LCD control register (see Table 14). It is not recommended to select an internal frequency lower than 32.768kHz as the clock supervisor circuit may switch off the LCD peripheral if the lower frequency is detected.

When the display is turned off, all segment and common outputs are switched to ground, causing all the segments to be switched off regardless of the contents of the LCD RAM.

When the Stand-by oscillator function is selected, the 32kHz stand-by oscillator is selected as clock source for the LCD.

To avoid incomplete frames of the LCD, the mode control bits do not immediately influence the LCD controller when the LCD control register is written. They are stored in a temporary register and change the LCD function only at the end of the frame. Special care must be taken when entering the STOP mode. After switching the LCD clock source from the main oscillator to the 32kHz standby oscillator it must be guaranteed that enough clock pulses are delivered to complete the current frame before entering the STOP mode. Otherwise the LCD function will not be changed and the LCD will be switched OFF after entering the STOP mode. Different LCD frame frequencies for each display mode are selected by bits in the LCD control register (see Table 15).

Figure 47. LCD Mode Control Register



DS0, DS1. Duty cycle select bits. These bits select the number of common backplanes used by the LCD control. This allows different multiplexing conditions.

HF0, HF1, HF2. These bits allow the LCD controller to be supplied with the correct frequency when different high main oscillator frequencies are selected as system clock. Table 14 shows the set-up for different clock crystals.

LF0, LF1, LF2. These bits control the LCD base operational frequency of the LCD common lines. Table 15 shows the set-up to select the different frequencies while table 16 shows the correspond-

Table 13. Duty Cycle Selection

DS1	DS0	Display Mode	Active Blackplanes	Max. Number of Segments Driven
0	0	1/4 duty	COM1, 2, 3, 4	180
0	1	1/1 duty	COM1	45
1	0	1/2 duty	COM1, 2	90
1	1	1/3 duty	COM1, 2, 3	135

LCD CONTROLLER-DRIVER (Continued)

Table 14. High Frequency Select Bits

HF2	HF1	HF0	Function	f_{OSC}
0	0	0	Display off	
0	0	1	for stand-by Oscillator	32.768kHz
0	1	0	NOT TO BE USED	
0	1	1	÷ 32 for main oscillator	1.048MHz
1	0	0	÷ 64 for main oscillator	2.097MHz
1	0	1	÷ 128 for main oscillator	4.194MHz
1	1	0	÷ 256 for main oscillator	8.388MHz
1	1	1	NOT TO BE USED	

Notes :

1. The usage f_{osc} values different from those defined in this table cause the LCD to operate at a reference frequency different from 32.768kHz.
2. It is not recommended to select an internal frequency lower than 32.768kHz as the clock supervisor circuit may switch off the LCD peripheral if lower frequency is detected.

Table 15. LCD Frequency Select Bits

LF2	LF1	LF0	f_{LCD} (Hz)
0	0	0	64
0	0	1	85
0	1	0	128
0	1	1	171
1	0	0	256
1	0	1	341
1	1	0	512
1	1	1	Not to be Used

ing frame values with the different multiplexing conditions.

According to the selected LCD drive frequency f_{LCD} the frame frequencies come out as shown in Table 16.

The Figure 54 illustrates the waveforms of the different duty signals.•

Table 16. Available Frame Frequencies for LCD

f_{LCD} (Hz)	Frame Frequency f_F (Hz)			
	1/1 duty	1/2 duty	1/3 duty	1/4 duty
512	512	256	171	128
341	341	171	114	85
256	256	128	85	64
171	171	85	57	43
128	128	64	43	32
85	85	43	28	21
64	64	32	21	16

The value of the VLCD voltage can be chosen independently from V_{DD} according to the display requirements. The intermediate VLCD levels 2/3 VLCD, 1/3 VLCD and 1/2 VLCD are generated by an internal resistor network as shown in Figures 52 and 53. The half VLCD level for 1/2 duty cycle is obtained by the external connection of VLCD1/3 and VLCD2/3 pins. All intermediate VLCD levels are connected to pins to enable external capacitive buffering or resistive shunting.

LCD CONTROLLER-DRIVER (Continued)

The internal resistive divider network is realized with two parallel dividers. One has high resistivity, the other one low resistivity. The high resistive divider (R_H) is permanently switched on during the LCD operation. The low resistive divider (R_L) is only switched on for a short period of time when the levels of common lines and segment lines are changed. This method combines low source impedance for fast switching of the LCD pixels with high source impedance for low power consumption. Fig. 48 shows the typical current into V_{LCD} pin in dependency of the display voltage V_{LCD}. When the display is switched off (by program or reset) the internal resistor network is also switched off to achieve minimum power consumption. The low resistivity divider is active at each edge of f_{LCD} during 8 clock cycles of F_{32kHz}.

The internal resistor network is implemented with resistive transistor elements to achieve high precision. For display voltages $V_{LCD} < 4.5V$ the resistivity of the divider may be too high for some applications (especially using 1/3 or 1/4 duty display mode). In that case an external resistive divider must be used to achieve the desired resistivity.

Figure 48. Typical Current Consumption on VLCD Pin (25°C, no load, fLCD= 512 Hz, mux=1/3 -1/4)

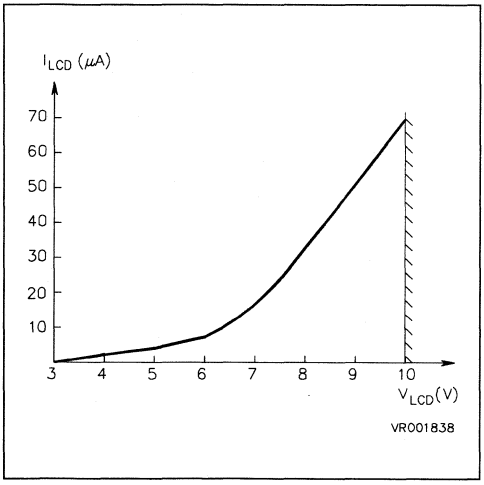
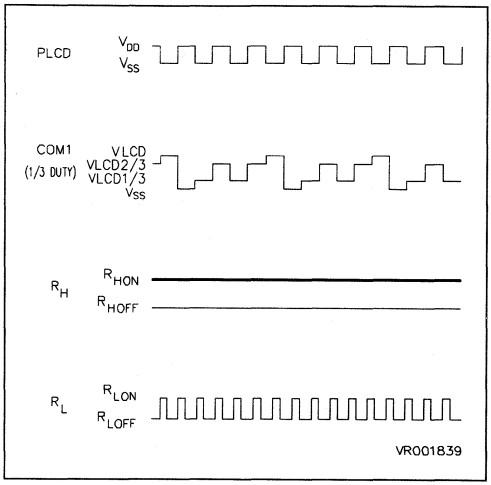


Figure 49. Typical Chronogram of Activation of the V_{LCD} Divider Network



LCD CONTROLLER-DRIVER (Continued)

Typical External resistances values are in the range of 100 kΩ to 150 kΩ. External capacitances in the range of 10 to 47 nF can be added to V_{LCD} 2/3 and V_{LCD} 1/3 pins and to V_{LCD} if the V_{LCD} connection is highly impedant.

When the program is switched off (by program or reset) the internal resistor network is also switched off to achieve minimum power consumption.

Figure 50. Typical Network to connect to V_{LCD} pins if $V_{LCD} \leq 4.5V$

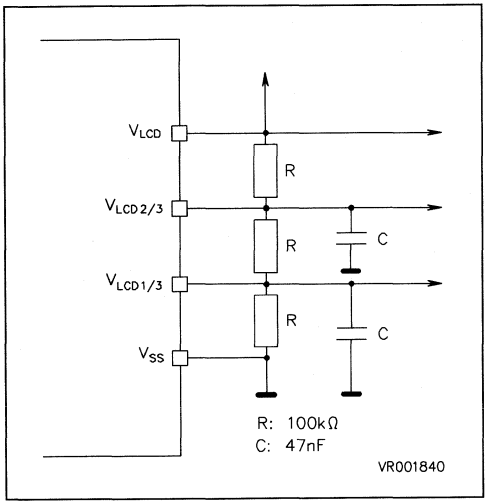


Figure 51. Generation of the 32kHz clock

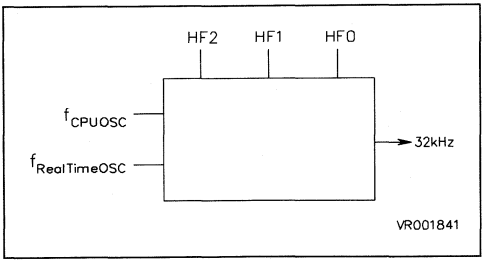


Figure 52. Bias Config for 1/2 Duty

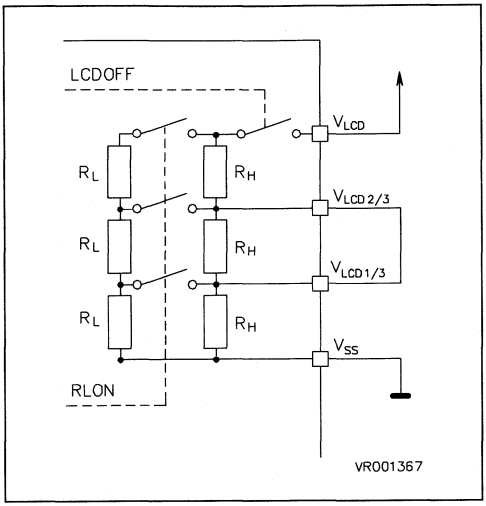
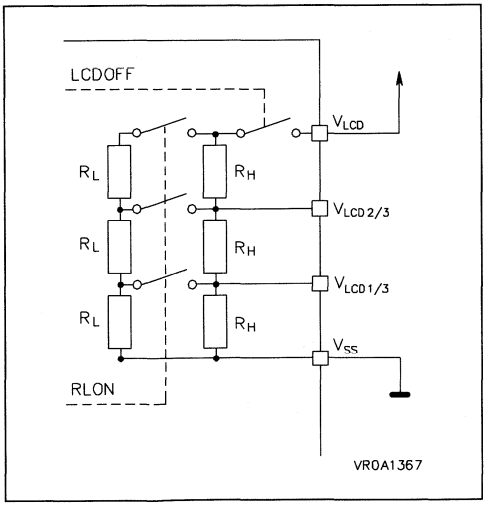


Figure 53. Bias Configuration for 1/1, 1/3 and 1/4 Duty Operation of LCD



LCD CONTROLLER-DRIVER (Continued)

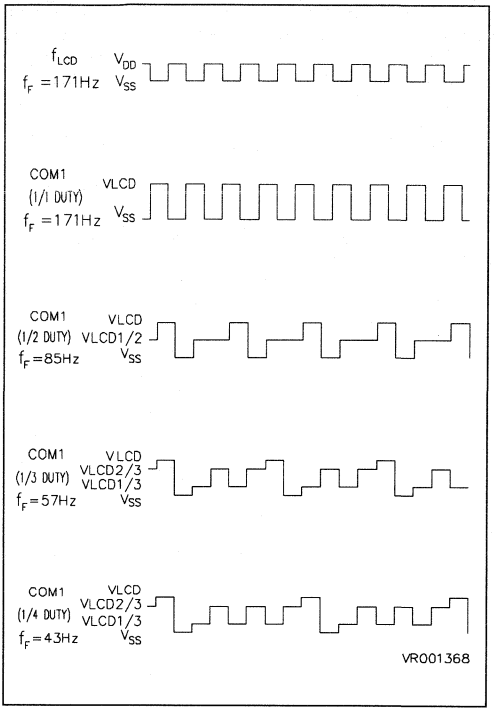
Address Mapping of the Display Segments.

The LCD RAM is located in the ST6240 data space from addresses E0h to F7h. The LCD forms a matrix of 45 segment lines (rows) and up to 4 common lines (columns). Each bit of the LCD RAM is mapped to one element of the LCD matrix, as described in Figure 55. If a bit is set, the corresponding LCD segment is switched on; if it is reset, the segment is switched off. The segments outputs S1, S2 and S3 are not connected to any pin.

When multiplex rates lower than 1/4 are selected, the unused LCD RAM is free for general use. In the 1/2 duty mode, for instance, half of the LCD RAM is available for storing general purpose data. The address range from F8h to FEh can be used as general purpose data RAM, but not for displaying data (it is reserved for future LCD expansion).

After a reset, the LCD RAM is not initialized and contains arbitrary information. As the LCD control register is reset, the LCD is completely switched off.

Figure 54. Common Signal Waveforms



LCD CONTROLLER-DRIVER (Continued)

Figure 55. Addressing Map of the LCD RAM

Data RAM Address	MSB							LSB	
E0	S8	S7	S6	S5	S4	NA	NA	NA	
E1	S16	S15	S14	S13	S12	S11	S10	S9	
E2	S24	S23	S22	S21	S20	S19	S18	S17	
E3	S32	S31	S30	S29	S28	S27	S26	S25	COM1
E4	S40	S39	S38	S37	S36	S35	S34	S33	
E5	S48	S47	S46	S45	S44	S43	S42	S41	
E6	S8	S7	S6	S5	S4	NA	NA	NA	
E7	S16	S15	S14	S13	S12	S11	S10	S9	
E8	S24	S23	S22	S21	S20	S19	S18	S17	
E9	S32	S31	S30	S29	S28	S27	S26	S25	COM2
EA	S40	S39	S38	S37	S36	S35	S34	S33	
EB	S48	S47	S46	S45	S44	S43	S42	S41	
EC	S8	S7	S6	S5	S4	NA	NA	NA	
ED	S16	S15	S14	S13	S12	S11	S10	S9	
EE	S24	S23	S22	S21	S20	S19	S18	S17	
EF	S32	S31	S30	S29	S28	S27	S26	S25	COM3
F0	S40	S39	S38	S37	S36	S35	S34	S33	
F1	S48	S47	S46	S45	S44	S43	S42	S41	
F2	S8	S7	S6	S5	S4	NA	NA	NA	
F3	S16	S15	S14	S13	S12	S11	S10	S9	
F4	S24	S23	S22	S21	S20	S19	S18	S17	
F5	S32	S31	S30	S29	S28	S27	S26	S25	COM4
F6	S40	S39	S38	S37	S36	S35	S34	S33	
F7	S48	S47	S46	S45	S44	S43	S42	S41	
F8 - FE	TO BE USED AS GENERAL PURPOSE DATA RAM (NOT FOR DISPLAY DATA)								

Notes:

In STOP mode no clock is available for the LCD controller from the main oscillator. If the 32kHz oscillator is activated the LCD can also operate in STOP mode. If the stand-by oscillator is not active, the LCD controller is switched off when STOP instruction is executed; this mode has to be selected to reach the lowest power consumption.

A missing LCD clock (no oscillator active, broken crystal, etc.) is detected by a clock supervisor circuit that switches all the segments and common lines to ground to avoid destructive DC levels at the LCD.

The LCD function change is only effective at the end of a frame. For this reason special care has to be taken when entering the STOP mode. After switching the LCD clock source from the main oscillator to the 32kHz stand-by oscillator it must be guaranteed that enough clock pulses are delivered to complete the current frame before entering the STOP mode. Otherwise the LCD function will not be changed and the LCD is switched off after entering the STOP mode.

The RAM address F8-FEh are not used for LCD display purposes. So they are available as 7 additional Data RAM registers.

SOFTWARE DESCRIPTION

The ST62xx software has been designed to fully use the hardware in the most efficient way possible while keeping byte usage to a minimum, in short to provide byte efficient programming capability. The ST62xx core has the ability to set or clear any register or RAM location bit of the Data space with a single instruction. Furthermore, the program may branch to a selected address depending on the status of any bit of the Data space. The carry bit is stored with the value of the bit when the SET or RES instruction is processed.

Addressing Modes

The ST62xx core has nine addressing modes which are described in the following paragraphs. The ST62xx core uses three different address spaces: Program space, Data space, and Stack space. Program space contains the instructions which are to be executed, plus the data for immediate mode instructions. Data space contains the Accumulator, the X,Y,V and W registers, peripheral and Input/Output registers, the RAM locations and Data ROM locations (for storage of tables and constants). Stack space contains six 12-bit RAM cells used to stack the return addresses for subroutines and interrupts.

Immediate. In the immediate addressing mode, the operand of the instruction follows the opcode location. As the operand is a ROM byte, the immediate addressing mode is used to access constants which do not change during program execution (e.g., a constant used to initialize a loop counter).

Direct. In the direct addressing mode, the address of the byte that is processed by the instruction is stored in the location that follows the opcode. Direct addressing allows the user to directly address the 256 bytes in Data Space memory with a single two-byte instruction.

Short Direct. The core can address the four RAM registers X,Y,V,W (locations 80h, 81h, 82h, 83h) in the short-direct addressing mode . In this case, the instruction is only one byte and the selection of the location to be processed is contained in the opcode. Short direct addressing is a subset of the direct addressing mode. (Note that 80h and 81h are also indirect registers).

Extended. In the extended addressing mode, the 12-bit address needed to define the instruction is obtained by concatenating the four less significant bits of the opcode with the byte following the opcode. The instructions (JP, CALL) that use the extended addressing mode are able to branch to any address of the 4K bytes Program space.

An extended addressing mode instruction is two-byte long.

Program Counter Relative. The relative addressing mode is only used in conditional branch instructions. The instruction is used to perform a test and, if the condition is true, a branch with a span of -15 to +16 locations around the address of the relative instruction. If the condition is not true, the instruction that follows the relative instruction is executed. The relative addressing mode instruction is one-byte long. The opcode is obtained in adding the three most significant bits that characterize the kind of the test, one bit that determines whether the branch is a forward (when it is 0) or backward (when it is 1) branch and the four less significant bits that give the span of the branch (0h to Fh) that must be added or subtracted to the address of the relative instruction to obtain the address of the branch.

Bit Direct. In the bit direct addressing mode, the bit to be set or cleared is part of the opcode, and the byte following the opcode points to the address of the byte in which the specified bit must be set or cleared. Thus, any bit in the 256 locations of Data space memory can be set or cleared.

Bit Test & Branch. The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit test and branch instruction is three-byte long. The bit identification and the tested condition are included in the opcode byte. The address of the byte to be tested follows immediately the opcode in the Program space. The third byte is the jump displacement, which is in the range of -126 to +129. This displacement can be determined using a label, which is converted by the assembler.

Indirect. In the indirect addressing mode, the byte processed by the register-indirect instruction is at the address pointed by the content of one of the indirect registers, X or Y (80h,81h). The indirect register is selected by the bit 4 of the opcode. A register indirect instruction is one byte long.

Inherent. In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. These instructions are one byte long.

SOFTWARE DESCRIPTION (Continued)

Instruction Set

The ST62xx core has a set of 40 basic instructions. When these instructions are combined with nine addressing modes, 244 usable opcodes can be obtained. They can be divided into six different types:load/store, arithmetic/logic, conditional branch, control instructions, jump/call, bit manipulation. The following paragraphs describe the different types.

All the instructions within a given type are presented in individual tables.

Load & Store. These instructions use one,two or three bytes in relation with the addressing mode. One operand is the Accumulator for LOAD and the other operand is obtained from data memory using one of the addressing modes.

For Load Immediate one operand can be any of the 256 data space bytes while the other is always immediate data.

Table 17. Load & Store Instructions

Instruction	Addressing Mode	Bytes	Cycles	Flags	
				Z	C
LD A, X	Short Direct	1	4	Δ	*
LD A, Y	Short Direct	1	4	Δ	*
LD A, V	Short Direct	1	4	Δ	*
LD A, W	Short Direct	1	4	Δ	*
LD X, A	Short Direct	1	4	Δ	*
LD Y, A	Short Direct	1	4	Δ	*
LD V, A	Short Direct	1	4	Δ	*
LD W, A	Short Direct	1	4	Δ	*
LD A, rr	Direct	2	4	Δ	*
LD rr, A	Direct	2	4	Δ	*
LD A, (X)	Indirect	1	4	Δ	*
LD A, (Y)	Indirect	1	4	Δ	*
LD (X), A	Indirect	1	4	Δ	*
LD (Y), A	Indirect	1	4	Δ	*
LDI A, #N	Immediate	2	4	Δ	*
LDI rr, #N	Immediate	3	4	*	*

Notes:
X,Y. Indirect Register Pointers, V & W Short Direct Registers
#. Immediate data (stored in ROM memory)
rr. Data space register
Δ. Affected
*. Not Affected

SOFTWARE DESCRIPTION (Continued)

Arithmetic and Logic. These instructions are used to perform the arithmetic calculations and logic operations. In AND, ADD, CP, SUB instructions one operand is always the accumulator while the other can be either a data space memory content or an immediate value in relation with the addressing mode. In CLR, DEC, INC instructions the operand can be any of the 256 data space addresses. In COM, RLC, SLA the operand is always the accumulator.

Table 18. Arithmetic & Logic Instructions

Instruction	Addressing Mode	Bytes	Cycles	Flags	
				Z	C
ADD A, (X)	Indirect	1	4	Δ	Δ
ADD A, (Y)	Indirect	1	4	Δ	Δ
ADD A, rr	Direct	2	4	Δ	Δ
ADDI A, #N	Immediate	2	4	Δ	Δ
AND A, (X)	Indirect	1	4	Δ	*
AND A, (Y)	Indirect	1	4	Δ	*
AND A, rr	Direct	2	4	Δ	*
ANDI A, #N	Immediate	2	4	Δ	*
CLR A	Short Direct	2	4	Δ	Δ
CLR rr	Direct	3	4	*	*
COM A	Inherent	1	4	Δ	Δ
CP A, (X)	Indirect	1	4	Δ	Δ
CP A, (Y)	Indirect	1	4	Δ	Δ
CP A, rr	Direct	2	4	Δ	Δ
CPI A, #N	Immediate	2	4	Δ	Δ
DEC X	Short Direct	1	4	Δ	*
DEC Y	Short Direct	1	4	Δ	*
DEC V	Short Direct	1	4	Δ	*
DEC W	Short Direct	1	4	Δ	*
DEC A	Direct	2	4	Δ	*
DEC rr	Direct	2	4	Δ	*
DEC (X)	Indirect	1	4	Δ	*
DEC (Y)	Indirect	1	4	Δ	*
INC X	Short Direct	1	4	Δ	*
INC Y	Short Direct	1	4	Δ	*
INC V	Short Direct	1	4	Δ	*
INC W	Short Direct	1	4	Δ	*
INC A	Direct	2	4	Δ	*
INC rr	Direct	2	4	Δ	*
INC (X)	Indirect	1	4	Δ	*
INC (Y)	Indirect	1	4	Δ	*
RLC A	Inherent	1	4	Δ	Δ
SLA A	Inherent	2	4	Δ	Δ
SUB A, (X)	Indirect	1	4	Δ	Δ
SUB A, (Y)	Indirect	1	4	Δ	Δ
SUB A, rr	Direct	2	4	Δ	Δ
SUBI A, #N	Immediate	2	4	Δ	Δ

Notes:
X,Y. Indirect Register Pointers, V & W Short Direct Registers
#. Immediate data (stored in ROM memory)
rr. Data space register
Δ. Affected
*. Not Affected

SOFTWARE DESCRIPTION (Continued)

Conditional Branch. The branch instructions achieve a branch in the program when the selected condition is met.

Bit Manipulation Instructions. These instructions can handle any bit in data space memory. One group either sets or clears. The other group (see Conditional Branch) performs the bit test branch operations.

Control Instructions. The control instructions control the MCU operations during program execution.

Jump and Call. These two instructions are used to perform long (12-bit) jumps or subroutines call inside the whole program space.

Table 19. Conditional Branch Instructions

Instruction	Branch If	Bytes	Cycles	Flags	
				Z	C
JRC e	C = 1	1	2	*	*
JRNC e	C = 0	1	2	*	*
JRZ e	Z = 1	1	2	*	*
JRNZ e	Z = 0	1	2	*	*
JRR b, rr, ee	Bit = 0	3	5	*	Δ
JRS b, rr, ee	Bit = 1	3	5	*	Δ

Notes:
b. 3-bit address
e. 5 bit signed displacement in the range -15 to +16
ee. 8 bit signed displacement in the range -126 to +129
rr. Data space register
Δ. Affected
*. Not Affected

Table 20. Bit Manipulation Instructions

Instruction	Addressing Mode	Bytes	Cycles	Flags	
				Z	C
SET b,rr	Bit Direct	2	4	*	*
RES b,rr	Bit Direct	2	4	*	*

Notes:
b. 3-bit address;
rr. Data space register;
*. Not Affected

Table 21. Control Instructions

Instruction	Addressing Mode	Bytes	Cycles	Flags	
				Z	C
NOP	Inherent	1	2	*	*
RET	Inherent	1	2	*	*
RETI	Inherent	1	2	Δ	Δ
STOP (1)	Inherent	1	2	*	*
WAIT	Inherent	1	2	*	*

Notes:
1. This instruction is deactivated and a WAIT is automatically executed instead of a STOP if the Watchdog function is selected.
Δ. Affected
*. Not Affected

Table 22. Jump & Call Instructions

Instruction	Addressing Mode	Bytes	Cycles	Flags	
				Z	C
CALL abc	Extended	2	4	*	*
JP abc	Extended	2	4	*	*

Notes:
abc. 12-bit address;
*. Not Affected

SOFTWARE DESCRIPTION (Continued)

Opcode Map Summary. The following table contains an opcode map for the instructions used on the MCU.

LOW / HI	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111	8 1000	9 1001	A 1010	B 1011	C 1100	D 1101	E 1110	F 1111	LOW / HI
0 0000	2 **JRNZ** e 1 pcr	4 **CALL** abc 2 ext	2 **JRNC** e 1 pcr	5 **JRR** b0,rr,ee 3 bt	2 **JRZ** e 1 pcr	# 	2 **JRC** e 1 prc	4 **LD** a,(x) 1 ind	2 **JRNZ** e 1 pcr	4 **JP** abc 2 ext	2 **JRNC** e 1 pcr	4 **RES** b0,rr 2 b.d	2 **JRZ** e 1 pcr	4 **LDI** rr,nn 3 imm	2 **JRC** e 1 pcr	4 **LD** a,(y) 1 ind	0 0000
1 0001	2 **JRNZ** e 1 pcr	4 **CALL** abc 2 ext	2 **JRNC** e 1 pcr	5 **JRS** b0,rr,ee 3 bt	2 **JRZ** e 1 pcr	4 **INC** x 1 sd	2 **JRC** e 1 prc	4 **LDI** a,nn 2 imm	2 **JRNZ** e 1 pcr	4 **JP** abc 2 ext	2 **JRNC** e 1 pcr	4 **SET** b0,rr 2 b.d	2 **JRZ** e 1 pcr	4 **DEC** x 1 sd	2 **JRC** e 1 pcr	4 **LD** a,rr 2 dir	1 0001
2 0010	2 **JRNZ** e 1 pcr	4 **CALL** abc 2 ext	2 **JRNC** e 1 pcr	5 **JRR** b4,rr,ee 3 bt	2 **JRZ** e 1 pcr	# 	2 **JRC** e 1 prc	4 **CP** a,(x) 1 ind	2 **JRNZ** e 1 pcr	4 **JP** abc 2 ext	2 **JRNC** e 1 pcr	4 **RES** b4,rr 2 b.d	2 **JRZ** e 1 pcr	4 **COM** a 1 inh	2 **JRC** e 1 pcr	4 **CP** a,(y) 1 ind	2 0010
3 0011	2 **JRNZ** e 1 pcr	4 **CALL** abc 2 ext	2 **JRNC** e 1 pcr	5 **JRS** b4,rr,ee 3 bt	2 **JRZ** e 1 pcr	4 **LD** a,x 1 sd	2 **JRC** e 1 prc	4 **CPI** a,nn 2 imm	2 **JRNZ** e 1 pcr	4 **JP** abc 2 ext	2 **JRNC** e 1 pcr	4 **SET** b4,rr 2 b.d	2 **JRZ** e 1 pcr	4 **LD** x,a 1 sd	2 **JRC** e 1 pcr	4 **CP** a,rr 2 dir	3 0011
4 0100	2 **JRNZ** e 1 pcr	4 **CALL** abc 2 ext	2 **JRNC** e 1 pcr	5 **JRR** b2,rr,ee 3 bt	2 **JRZ** e 1 pcr	# 	2 **JRC** e 1 prc	4 **ADD** a,(x) 1 ind	2 **JRNZ** e 1 pcr	4 **JP** abc 2 ext	2 **JRNC** e 1 pcr	4 **RES** b2,rr 2 b.d	2 **JRZ** e 1 pcr	2 **RETI** 1 inh	2 **JRC** e 1 pcr	4 **ADD** a,(y) 1 ind	4 0100
5 0101	2 **JRNZ** e 1 pcr	4 **CALL** abc 2 ext	2 **JRNC** e 1 pcr	5 **JRS** b2,rr,ee 3 bt	2 **JRZ** e 1 pcr	4 **INC** y 1 sd	2 **JRC** e 1 prc	4 **ADDI** a,nn 2 imm	2 **JRNZ** e 1 pcr	4 **JP** abc 2 ext	2 **JRNC** e 1 pcr	4 **SET** b2,rr 2 b.d	2 **JRZ** e 1 pcr	4 **DEC** y 1 sd	2 **JRC** e 1 pcr	4 **ADD** a,rr 2 dir	5 0101
6 0110	2 **JRNZ** e 1 pcr	4 **CALL** abc 2 ext	2 **JRNC** e 1 pcr	5 **JRR** b6,rr,ee 3 bt	2 **JRZ** e 1 pcr	# 	2 **JRC** e 1 prc	4 **INC** (x) 1 ind	2 **JRNZ** e 1 pcr	4 **JP** abc 2 ext	2 **JRNC** e 1 pcr	4 **RES** b6,rr 2 b.d	2 **JRZ** e 1 pcr	2 **STOP** 1 inh	2 **JRC** e 1 pcr	4 **INC** (y) 1 ind	6 0110
7 0111	2 **JRNZ** e 1 pcr	4 **CALL** abc 2 ext	2 **JRNC** e 1 pcr	5 **JRS** b6,rr,ee 3 bt	2 **JRZ** e 1 pcr	4 **LD** a,y 1 sd	2 **JRC** e 1 prc	# 	2 **JRNZ** e 1 pcr	4 **JP** abc 2 ext	2 **JRNC** e 1 pcr	4 **SET** b6,rr 2 b.d	2 **JRZ** e 1 pcr	4 **LD** y,a 1 sd	2 **JRC** e 1 pcr	4 **INC** rr 2 dir	7 0111
8 1000	2 **JRNZ** e 1 pcr	4 **CALL** abc 2 ext	2 **JRNC** e 1 pcr	5 **JRR** b1,rr,ee 3 bt	2 **JRZ** e 1 pcr	# 	2 **JRC** e 1 prc	4 **LD** (x),a 1 ind	2 **JRNZ** e 1 pcr	4 **JP** abc 2 ext	2 **JRNC** e 1 pcr	4 **RES** b1,rr 2 b.d	2 **JRZ** e 1 pcr	# 	2 **JRC** e 1 pcr	4 **LD** (y),a 1 ind	8 1000
9 1001	2 **JRNZ** e 1 pcr	4 **CALL** abc 2 ext	2 **JRNC** e 1 pcr	5 **JRS** b1,rr,ee 3 bt	2 **JRZ** e 1 pcr	4 **INC** v 1 sd	2 **JRC** e 1 prc	# 	2 **JRNZ** e 1 pcr	4 **JP** abc 2 ext	2 **JRNC** e 1 pcr	4 **SET** b1,rr 2 b.d	2 **JRZ** e 1 pcr	4 **DEC** v 1 sd	2 **JRC** e 1 pcr	4 **LD** rr,a 2 dir	9 1001
A 1010	2 **JRNZ** e 1 pcr	4 **CALL** abc 2 ext	2 **JRNC** e 1 pcr	5 **JRR** b5,rr,ee 3 bt	2 **JRZ** e 1 pcr	# 	2 **JRC** e 1 prc	4 **AND** a,(x) 1 ind	2 **JRNZ** e 1 pcr	4 **JP** abc 2 ext	2 **JRNC** e 1 pcr	4 **RES** b5,rr 2 b.d	2 **JRZ** e 1 pcr	4 **RLC** a 1 inh	2 **JRC** e 1 pcr	4 **AND** a,(y) 1 ind	A 1010
B 1011	2 **JRNZ** e 1 pcr	4 **CALL** abc 2 ext	2 **JRNC** e 1 pcr	5 **JRS** b5,rr,ee 3 bt	2 **JRZ** e 1 pcr	4 **LD** a,v 1 sd	2 **JRC** e 1 prc	4 **ANDI** a,nn 2 imm	2 **JRNZ** e 1 pcr	4 **JP** abc 2 ext	2 **JRNC** e 1 pcr	4 **SET** b5,rr 2 b.d	2 **JRZ** e 1 pcr	4 **LD** v,a 1 sd	2 **JRC** e 1 pcr	4 **AND** a,rr 2 dir	B 1011
C 1100	2 **JRNZ** e 1 pcr	4 **CALL** abc 2 ext	2 **JRNC** e 1 pcr	5 **JRR** b3,rr,ee 3 bt	2 **JRZ** e 1 pcr	# 	2 **JRC** e 1 prc	4 **SUB** a,(x) 1 ind	2 **JRNZ** e 1 pcr	4 **JP** abc 2 ext	2 **JRNC** e 1 pcr	4 **RES** b3,rr 2 b.d	2 **JRZ** e 1 pcr	2 **RET** 1 inh	2 **JRC** e 1 pcr	4 **SUB** a,(y) 1 ind	C 1100
D 1101	2 **JRNZ** e 1 pcr	4 **CALL** abc 2 ext	2 **JRNC** e 1 pcr	5 **JRS** b3,rr,ee 3 bt	2 **JRZ** e 1 pcr	4 **INC** w 1 sd	2 **JRC** e 1 prc	4 **SUBI** a,nn 2 imm	2 **JRNZ** e 1 pcr	4 **JP** abc 2 ext	2 **JRNC** e 1 pcr	4 **SET** b3,rr 2 b.d	2 **JRZ** e 1 pcr	4 **DEC** w 1 sd	2 **JRC** e 1 pcr	4 **SUB** a,rr 2 dir	D 1101
E 1110	2 **JRNZ** e 1 pcr	4 **CALL** abc 2 ext	2 **JRNC** e 1 pcr	5 **JRR** b7,rr,ee 3 bt	2 **JRZ** e 1 pcr	# 	2 **JRC** e 1 prc	4 **DEC** (x) 1 ind	2 **JRNZ** e 1 pcr	4 **JP** abc 2 ext	2 **JRNC** e 1 pcr	4 **RES** b7,rr 2 b.d	2 **JRZ** e 1 pcr	2 **WAIT** 1 inh	2 **JRC** e 1 pcr	4 **DEC** (y) 1 ind	E 1110
F 1111	2 **JRNZ** e 1 pcr	4 **CALL** abc 2 ext	2 **JRNC** e 1 pcr	5 **JRS** b7,rr,ee 3 bt	2 **JRZ** e 1 pcr	4 **LD** a,w 1 sd	2 **JRC** e 1 prc	# 	2 **JRNZ** e 1 pcr	4 **JP** abc 2 ext	2 **JRNC** e 1 pcr	4 **SET** b7,rr 2 b.d	2 **JRZ** e 1 pcr	4 **LD** w,a 1 sd	2 **JRC** e 1 pcr	4 **DEC** rr 2 dir	F 1111

Abbreviations for Addressing Modes:

dir	Direct
sd	Short Direct
imm	Immediate
inh	Inherent
ext	Extended
b.d	Bit Direct
bt	Bit Test
pcr	Program Counter Relative
ind	Indirect

Legend:

#	Indicates Illegal Instructions
e	5 Bit Displacement
b	3 Bit Address
rr	1byte dataspace address
nn	1 byte immediate data
abc	12 bit address
ee	8 bit Displacement

ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings

This product contains devices to protect the inputs against damage due to high static voltages, however it is advised to take normal precaution to avoid application of any voltage higher than maximum rated voltages.

For proper operation it is recommended that V_I and V_O must be higher than V_{SS} and smaller than V_{DD}. Reliability is enhanced if unused inputs are connected to an appropriated logic voltage level (V_{DD} or V_{SS}).

Power Considerations. The average chip-junction temperature, Tj, in Celsius can be obtained from:

$$Tj = T_A + PD \times RthJA$$

Where : T_A = Ambient Temperature.

RthJA = Package thermal resistance (junction-to ambient).

PD = Pint + Pport.

Pint = $I_{DD} \times V_{DD}$ (chip internal power).

Pport = Port power dissipation (determinated by the user).

Symbol	Parameter	Value	Unit
V_{DD}	Supply Voltage	-0.3 to 7.0	V
V_{LCD}	Display Voltage	-0.3 to 11.0	V
V_I	Input Voltage	V_{SS} - 0.3 to V_{DD} + 0.3	V
V_O	Output Voltage	V_{SS} - 0.3 to V_{DD} + 0.3	V
I_O	Current Drain per Pin Excluding V_{DD} & V_{SS}	± 10	mA
IV_{DD}	Total Current into V_{DD} (source)	50	mA
IV_{SS}	Total Current out of V_{SS} (sink)	50	mA
Tj	Junction Temperature	150	°C
T_{STG}	Storage Temperature	-60 to 150	°C

Note : Stresses above those listed as "absolute maximum ratings" may cause permanent damage to the device . This is a stress rating only and functional operation of the device at these conditions is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

THERMAL CHARACTERISTIC

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
RthJA	Thermal Resistance	PQFP80		70		°C/W

RECOMMENDED OPERATING CONDITIONS

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
T_A	Operating Temperature	1 Suffix Version 6 Suffix Version	0 -40		70 85	°C
V_{DD}	Operating Supply Voltage		3		6	V
V_{LCD}	Display Voltage		3		10	V
V_{DD}	RAM Retention Voltage		2			V

RECOMMENDED OPERATING CONDITIONS (Continued)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
f_{OSC}	Oscillator Frequency (1)(4)	$V_{DD} \geq 4.5V$ $V_{DD} \geq 3V$	0.01 0.01		8.388 2	MHz
I_{INJ+}	Pin Injection Current (positive) Digital Input (2) Analog Input (3)	V_{DD} = 4.5 to 5.5V			+5	mA
I_{INJ-}	Pin Injection Current (negative) Digital Input (2) Analog Input	V_{DD} = 4.5 to 5.5V			-5	mA

Notes :
1. An oscillator frequency above 1MHz is recommanded for reliable A/D results.
2. A current of ± 5mA can be forced on each pin of the digital section without affecting the functional behaviour of the device. For a positive current injected into one pin, a part of this current (~ 10%) can be expected to flow from the neighbouring pins. A current of -5mA can be forced on one input of the analog section at a time (or -2.5mA for all inputs at a time) without affecting the conversion.
3. If a total current of +1mA is flowing into the single analog channel or if the total current flowing into all the analog inputs is of 1mA, all the conversion is resulting shifted of +1LSB. If a total positive current of +5mA is flowing into the single analog channel or if the total current flowing into all the analog inputs is of 5mA, all the conversion is resulting shifted of +2LSB.
4. Operation below 0.01 MHz is possible but requires increased supply current.

EEPROM INFORMATION

The ST62xx EEPROM single poly process has been specially developed to achieve 300.000 Write/Erase cycles and a 10 years data retention.

DC ELECTRICAL CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
V_{IL}	Input Low Level Voltage	RESET, NMI, TIMER, WDON Pin			$0.3V_{DD}$	V
V_{IH}	Input High Level Voltage	TIMER	$0.80V_{DD}$			V
		RESET, NMI, WDON Pin	$0.70V_{DD}$			V
I_{IL} I_{IH}	Input Leakage Current	RESET Pin V_{DD} =5V $V_{IN} = V_{DD}$ (1) $V_{IN} = V_{DD}$ (2) $V_{IN} = V_{SS}$ (5)			10 1 50	µA mA µA
V_{OL}	Low Level Output Voltage	TIMER, I_{OL} = 5.0mA			$0.2V_{DD}$	V
V_{OH}	High Level Output Voltage	TIMER, I_{OL} = -5.0mA	$0.65V_{DD}$			V
R_{PU}	Pull-up Resistor	V_{IN}=0V V_{DD}=5V WDON - NMI	40	100	200	kΩ
		RESET	200	300	500	kΩ

Notes on next page

DC ELECTRICAL CHARACTERISTICS (Continued)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
I_{IL} I_{IH}	Input Leakage Current	TIMER $V_{IN} = V_{DD}$ or V_{SS}		0.1	1.0	µA
I_{IL} I_{IH}	Input Leakage Current	NMI VDD = 5.V $V_{IN} = V_{SS}$ (5) $V_{IN} = V_{DD}$			 100 1.0	µA
I_{IL} I_{IH}	Input Leakage Current	WDON VDD = 5V $V_{IN} = V_{SS}$ (5) $V_{IN} = V_{DD}$			 100 1.0	µA
I_{DD}	Supply Current RUN Mode	f_{OSC} = 8MHz, I_{LOAD} = 0mA V_{DD} = 5.5V		4	7	mA
	Supply Current WAIT Mode (4)	f_{OSC} = 8MHz, I_{LOAD} = 0mA V_{DD} = 5.0V		1	3	mA
	Supply Current RESET Mode	f_{OSC} = 8MHz, $V_{RESET} = V_{SS}$		1	7	mA
	Supply Current STOP Mode (3)	I_{LOAD} = 0mA V_{DD} = 5.5V		1	10	µA

Notes :
1. No Watchdog Reset activated.
2. Reset generated by Watchdog.
3. When the watchdog function is actvated the STOP instruction is deactivated. WAIT instruction is automatically executed.
4. All on-chip peripherals in OFF state
5. Pull-up resistor

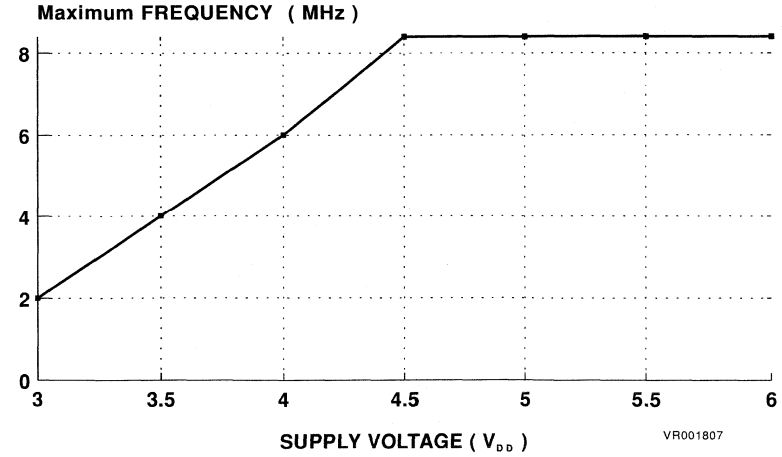
AC ELECTRICAL CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
f_{OSC}	Oscillator Frequency (2)	$V_{DD} \geq 4.5V$ $V_{DD} \geq 3V$	0.01		8.388 2	MHz
t_{SU}	Oscillator Start-up Time	$C_{L1} = C_{L2}$ = 22pF - crystal		5	10	ms
t_{SR}	Supply Rise Time	10% to 90%	0.01		100	
t_{REC}	Supply Recovery Time (1)		100			
T_W	Minimum Pulse Width	NMI Pin V_{DD} = 5V	100			ns
		RESET Pin	100			ns
T_{WEE}	EEPROM Write Time	T_A = 25°C One Byte T_A = 85°C One Byte		5 15	10 25	ms
Endurance	EEPROM WRITE/ERASE Cycles	Q_A LOT Acceptance Criteria	300.000	> 1 million		cycles
Retention	EEPROM Data Retention	T_A = 55°C	10			years
C_{IN}	Input Capacitance	All Inputs Pins			10	pF

Notes:
1. Period for which V_{DD} has to be connected or at 0V to allow internal Reset function at next power-up.
2. Operation below 0.01 MHz is possible but requires increased supply current.

Maximum Operating FREQUENCY (Fmax) Versus SUPPLY VOLTAGE (V_{DD})



I/O PORTS
(T_A = -40 to +85°C unless otherwhise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
C_{OUT}	Output Capacitance	All Outputs Pins			10	pF
V_{IL}	Input Low Level Voltage	I/O Pins			$0.3V_{DD}$	V
V_{IH}	Input High Level Voltage	I/O Pins	$0.7V_{DD}$			V
V_{OL}	Low Level Output Voltage	I/O Pins, I_O = 10µA (sink)			0.1	V
		I/O Pins, $I_{OL} = V_{DD}$x1mA V_{DD} = 4.5 to 6V			$0.16xV_{DD}$	V
		I/O Pins, I_{OL} = 1.6mA V_{DD} = 3V			0.4	V
	Low Level Output Voltage, PB4-PB7 Only	I/O Pins, $I_{OL} = V_{DD}$x2mA V_{DD} = 4.5 to 6V			$0.16xV_{DD}$	V
		I/O Pins, I_{OL} = 3.2mA V_{DD} = 3V			0.4	V
		I/O Pins, $I_{OL} = V_{DD}$x4mA V_{DD} = 4.5 to 6V			$0.26xV_{DD}$	V
		I/O Pins, I_{OL} = 6.4mA V_{DD} = 3V			0.8	V
V_{OH}	High Level Output Voltage	I/O Pins, I_O = -10µA (source)	V_{DD}-0.1			V
		I/O Pins, $I_{OL} = -V_{DD}$x1mA V_{DD} = 5.0V	$0.6xV_{DD}$			V
I_{IL} I_{IH}	Input Leakage Current	I/O Pins,(1)		0.1	1.0	µA
R_{PU}	Pull-up Resistor	I/O Pins V_{IN} = 0V, V_{DD} = 5.0V	40	100	200	KΩ

Note 1. Pull-up resistor off

SPI ELECTRICAL CHARACTERISTICS
(T_A = -40 to +85°C unless otherwhise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
F_{CL}	Clock Frequency	applied on PB5/SCL			500	kHz
t_{SU}	Set-up Time	applied on PB6/Sin		50		ns
t_h	Hold Time	applied on PB6/Sin		100		ns

A/D CONVERTER CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
Res	Resolution (3)			8		Bit
A_{TOT}	Total Accuracy (3)	f_{OSC} > 1.2 MHz f_{OSC} > 32kHz			± 2 ±4	LSB
t_C (1)	Conversion Time	f_{OSC} = 8MHz		70		µs
V_{AN}	Conversion Range		V_{SS}		V_{DD}	V
ZIR	Zero Input Reading	Conversion result when $V_{IN} = V_{SS}$	00			Hex
FSR	Full Scale Reading	Conversion result when $V_{IN} = V_{DD}$			FF	Hex
AD_I	Analog Input Current During Conversion	V_{DD}= 4.5V			1.0	µA
AC_{IN} (2)	Analog Input Capacitance			2	5	pF
ASI	Analog Source Impedance				30	kΩ
SSI	Analog Reference Supply Impedence				2	kΩ

Notes:
1. With oscillator frequencies less than 1MHz, the A/D Converter accuracy is decreased.
2. Excluding Pad Capacitance
3. Noise at $V_{DD}, V_{SS} \leq 10mV$

TIMER CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
t_{RES}	Resolution		$\frac{12}{f_{OSC}}$			second
f_{IN}	Input Frequency on TIMER Pin				$\frac{f_{OSC}}{8}$	MHz
t_W	Pulse Width at TIMER Pin	$V_{DD} \geq 3V$ $V_{DD} \geq 4.5V$	1 125			µs ns

PSS ELECTRICAL CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
V_{PSS}	PSS pin Input Voltage		V_{SS}		V_{DD}	V
I_{PSS}	PSS pin Input Current	PSS RUN PSS STOP V_{PSS} = 5V, T_A = 25°C			350 1	µA

LCD ELECTRICAL CHARACTERISTICS

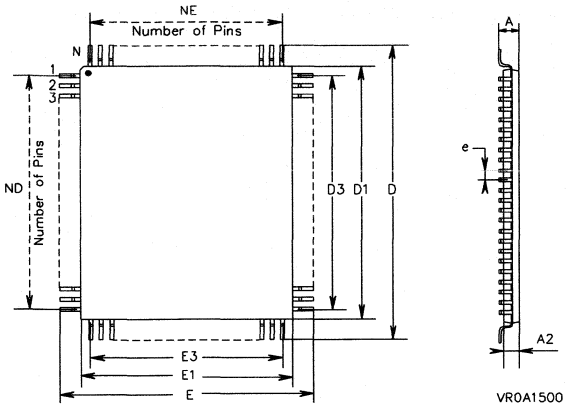
(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
f_{FR}	Frame Frequency	1/4 Duty f_{OSC} = 1, 2, 4, 8MHz	16		128	Hz
V_{OS}	DC Offset Voltage(1)	$V_{LCD} = V_{DD}$, no load			50	mV
V_{OH}	COM High Level, Output Voltage	I = 100µA, V_{LCD} = 5V	4.5V			V
V_{OL}	COM Low Level, Output Voltage	I = 100µA, V_{LCD} = 5V			0.5V	V
V_{OH}	SEG High Level, Output Voltage	I = 50µA, V_{LCD} = 5V	4.5V			V
V_{OL}	SEG Low Level, Output Voltage	I = 50µA, V_{LCD} = 5V			0.5V	V
V_{LCD}	Display Voltage	Note 2	3		10	V

Notes :
1. The DC offset voltage refers to all segment and common outputs. It is the difference between the measured voltage value and nominal value for every voltage level. Ri of voltage meter must be greater than or equal to 100MΩ.
2. An external resistances network is required when $V_{LCD} \leq 4.5V$.

PACKAGE MECHANICAL DATA

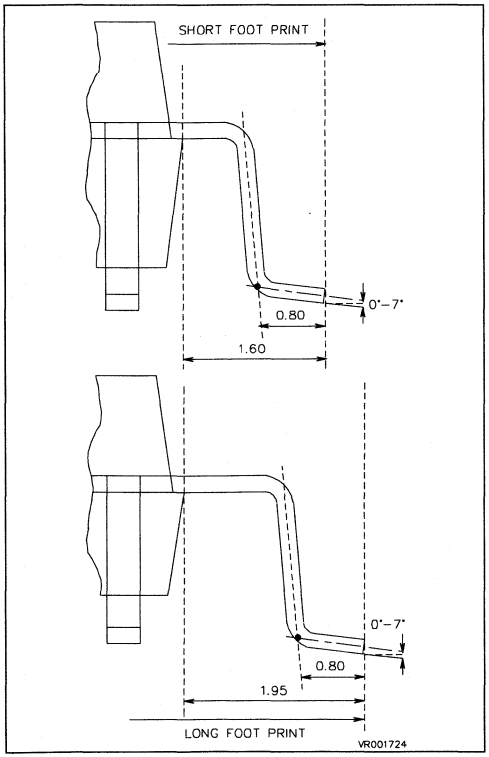
Figure 50. ST6240 80 Pin Plastic Quad Flat Pack Package



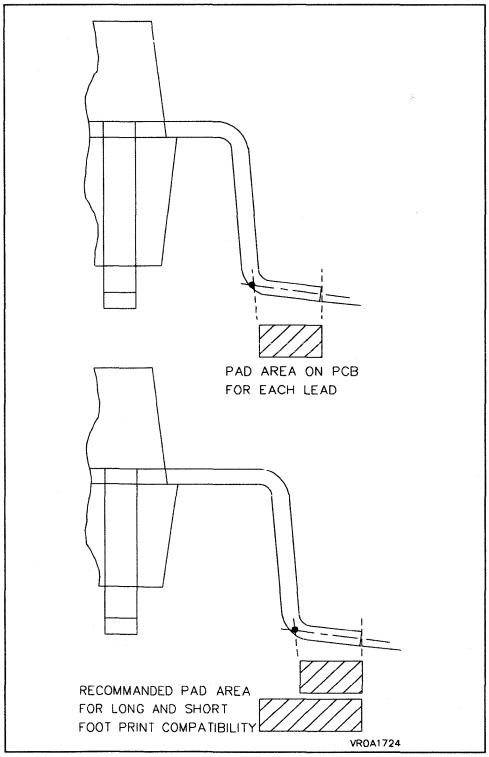
Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A			3.30			0.130
A2	2.55	2.80	3.05	0.100	0.110	0.120
D*	23.65	23.90	24.15	0.931	0.941	0.951
D1	19.90	20.00	20.10	0.783	0.787	0.791
D3		18.40			0.724	
E*	17.65	17.90	18.15	0.695	0.705	0.715
E1	13.90	14.00	14.10	0.547	0.551	0.555
E3		12.00			0.472	
e		0.80			0.032	
Number of Pins						
ND	24					
NE	16					
N	80					

* Subject to change,
Typ. D will change to 23.20mm
Typ. E will change to 17.20mm

Short/Long Footprint Measurement

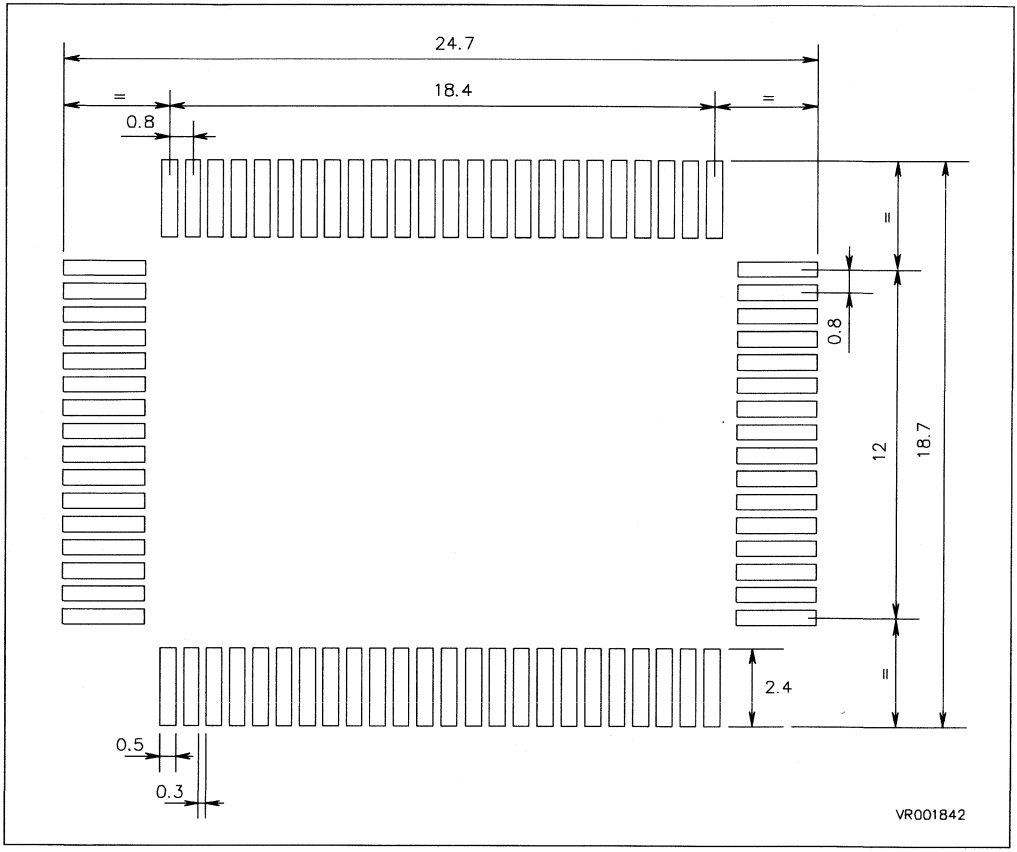


Short/Long Footprint recommended Padding



PACKAGE MECHANICAL DATA (Continued)

Recommanded Solder Pad Footprint For QFP80 (in mm)



ORDERING INFORMATION

The following chapter deals with the procedure for transfer the Program/Data ROM codes to SGS-THOMSON.

Communication of the ROM Codes. To communicate the contents of Program/Data ROM memories to SGS-THOMSON, the customer has to send :

- one file in INTEL INTELLEC 8/MDS FORMAT (in an MS-DOS 5" diskette) for the PROGRAM Memory
- one file in INTEL INTELLEC 8/MDS FORMAT (in a MS-DOS 5" diskette) for the EEPROM initial content (this file is optional)
- a filled Option List form as described in the OPTION LIST paragraph.

The program ROM should respect the ROM Memory Map as in Table 22.

The ROM code must be generated with ST6 assembler. Before programming the EPROM, the buffer of the EPROM programmer must be filled with FFh.

Table 22. ROM Memory Map

ROM Page	Device Address	Description
Page 0	0000h-007Fh 0080h-07FFh	Reserved User ROM
Page 1 "STATIC"	0800h-0F9Fh 0FA0h-0FEFh 0FF0h-0FF7h 0FF8h-0FFBh 0FFCh-0FFDh 0FFEh-0FFFh	User ROM Reserved Interrupt Vectors Reserved NMI Vector Reset Vector
Page 2	0000h-000Fh 0010h-07FFh	Reserved User ROM
Page 3	0000h-000Fh 0010h-07FFh	Reserved User ROM

Note : EPROM addresses are related to the ROM file to be processed.

Customer EEPROM Initial Contents : Format

a. The content should be written into an INTEL INTELLEC format file.

b. In the case of 128 bytes of EEPROM, the starting address in 000h and the end in 7Fh.

c. Undefined or don't care bytes should have the content FFh.

Listing Generation & Verification. When SGS-THOMSON receives the Codes, they are compared and a computer listing is generated from them. This listing refers exactly to the mask that will be used to produce the microcontroller. Then the listing is returned to the customer that must thoroughly check, complete, sign and return it to SGS-THOMSON. The signed list constitutes a part of the contractual agreement for the creation of the customer mask. SGS-THOMSON sales organization will provide detailed information on contractual points.

ORDERING INFORMATION TABLE

Sales Types	Temperature Range	Package
ST6240Q1/XX	0 to + 70°C	PQFP80
ST6240Q6/XX	-40 to + 85°C	PQFP80

Note : "XX" is the ROM code identifier allocated by SGS-THOMSON after receipt of all required options and the related ROM file.

ST6240 MICROCONTROLLER OPTION LIST

Customer . . . . . .

Address . . . . . .

Contact . . . . . .

Phone No . . . . . .

Reference . . . . . .

SGS-THOMSON Microelectronics references

Device	[ ] ST6240	
Package	[ ] Plastic Quad Flat Package	
Temperature Range	[ ] 0°C to + 70°C	[ ] -40°C to + 85°C

Special Marking [ ] No
[ ] Yes "_ _ _ _ _ _ _ _ _ _"

Authorized characters are Letters, digits, '.', '–', '/' and spaces only.
For marking one line with 10 characters maximum is possible.

Comments :
- Number of LCD segments used :
- Number of LCD backplanes used :
- PSS used:

Note :

Signature

Date

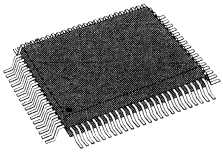
SGS-THOMSON MICROELECTRONICS

ST62E40
ST62T40

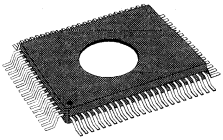
8-BIT EPROM HCMOS MCU WITH LCD DRIVER, EEPROM AND A/D CONVERTER

PRELIMINARY DATA

- 3 to 6V supply operating range
- 8.4MHz Maximum Clock Frequency
- Run, Wait & Stop Modes
- 5 different interrupt vectors
- Look-up table capability in EPROM
- User EPROM: 7948 bytes
  Data RAM: 192 bytes
  LCD RAM: 24 bytes
  EEPROM: 128 bytes
- PQFP80 and CQFP80-W packages
- 16 fully software programmable I/O as:
  - Input with/without pull-up resistor
  - Input with interrupt generation
  - Open-Drain or Push-pull outputs
  - Analog Inputs (12 pins)
- 4 I/O lines can sink up to 20mA for direct LED or TRIAC driving and have SPI alternate functions
- Two 8-bit counters and 7-bit programmable prescalers (Timer 1 and 2)
- Software or hardware activated digital watchdog
- 8-bit A/D converter with up to 12 analog inputs
- 8-bit synchronous serial peripheral interface (SPI)
- LCD driver with 45 segment outputs, 4 backplane outputs and selectable duty cycle for up to 180 LCD segments direct driving
- 32kHz oscillator for stand-by LCD operation
- Power Supply Supervisor (PSS)
- One external not maskable interrupt
- 9 powerful addressing modes
- The accumulator, the X, Y, V & W registers, the port and peripherals data & control registers are addressed in the data space as RAM locations.
- The ST62E40 is the EPROM version, ST62T40 is the OTP version, fully compatible with ST6240 ROM version.



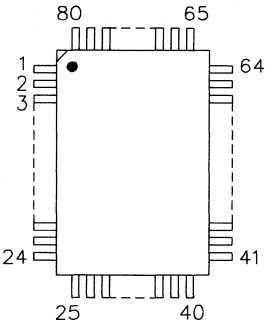
PQFP80



CQFP80-W

(Ordering Information at the end of the datasheet)

Figure 1. 80 Pin Quad Flat Pack (QFP) Package Pinout



ST62E40/T40 Pin Description

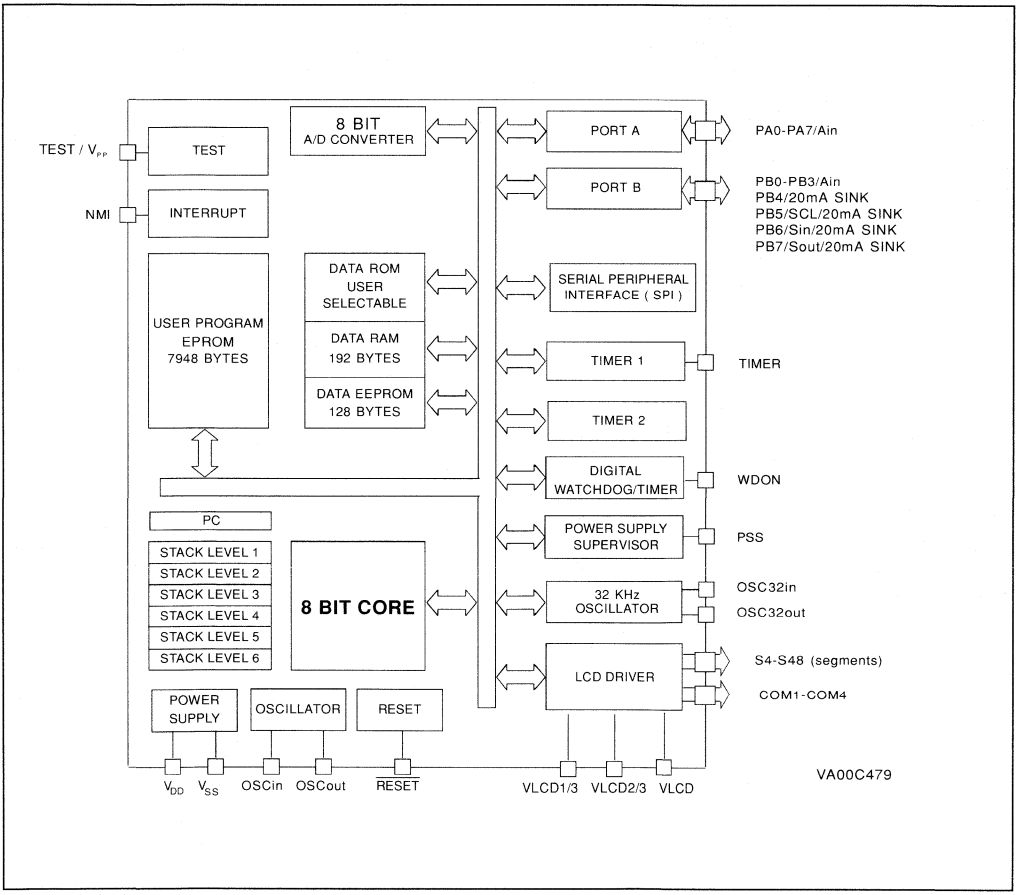
Pin number	Pin name	Pin number	Pin name	Pin number	Pin name	Pin number	Pin name
1	S43	25	$\overline{\text{RESET}}$	64	S26	65	S27
2	S44	26	OSCout	63	S25	66	S28
3	S45	27	OSCin	62	S24	67	S29
4	S46	28	WDON	61	S23	68	S30
5	S47	29	NMI	60	S22	69	S31
6	S48	30	TIMER	59	S21	70	S32
7	COM4	31	PB7/Sout (1)	58	S20	71	S33
8	COM3	32	PB6/Sin (1)	57	S19	72	S34
9	COM2	33	PB5/SCL (1)	56	S18	73	S35
10	COM1	34	PB4 (1)	55	S17	74	S36
11	VLCD1/3	35	PB3/Ain	54	S16	75	S37
12	VLCD2/3	36	PB2/Ain	53	S15	76	S38
13	VLCD	37	PB1/Ain	52	S14	77	S39
14	PA7/Ain	38	PB0/Ain	51	S13	78	S40
15	PA6/Ain	39	OSC32out	50	S12	79	S41
16	PA5/Ain	40	OSC32in	49	S11	80	S42
17	PA4/Ain			48	S10		
18	TEST/V_{PP}			47	S9		
19	PA3/Ain			46	S8		
20	PA2/Ain			45	S7		
21	PA1/Ain			44	S6		
22	PA0/Ain			43	S5		
23	V_{DD}			42	S4		
24	V_{SS}			41	PSS		

Note 1: 20mA Sink

GENERAL DESCRIPTION

The ST62E40,T40 microcontrollers are members of the 8-bit HCMOS ST62xx family, a series of devices oriented to low-medium complexity applications. They are the EPROM/OTP versions of the ST6240 ROM device and are suitable for product prototyping and low volume production. All ST62xx members are based on a building block approach: a common core is associated with a combination of on-chip peripherals (macrocells). The macrocells of the ST6240 family are: a high performance LCD controller/driver with 45 segment outputs and 4 backplanes able to drive up to 180 segments, two Timer peripherals each including an 8-bit counter with a 7-bit software programmable prescaler (Timer), the digital watchdog timer (DWD), an 8-bit A/D Converter with up to 12 analog inputs, a Power Supply Supervisor and an 8-bit synchronous Serial Peripheral Interface (SPI). In addition these devices offer 128 bytes of EEPROM for storage of non volatile data. Thanks to these peripherals the ST6240 family is well suited for general purpose, automotive, security, appliance and industrial applications.

Figure 2. ST62E40 Block Diagram



Note: Ain = Analog Input

PIN DESCRIPTION

V_{DD} and V_{SS}. Power is supplied to the MCU using these two pins. V_{DD} is power and V_{SS} is the ground connection.

OSCin and OSCout. These pins are internally connected with the on-chip oscillator circuit. A quartz crystal or a ceramic resonator can be connected between these two pins in order to allow the correct operation of the MCU with various stability/cost trade-offs. OSCin is the input pin, OSCout is the output pin. An external clock signal can be applied to OSCin.

$\overline{\text{RESET}}$. The active low $\overline{\text{RESET}}$ pin is used to restart the microcontroller at the beginning of its program. The $\overline{\text{RESET}}$ pin is provided with an on-chip pull-up resistor and schmitt trigger input characteristics.

TEST/V_{PP}. The TEST must be held at V_{SS} for normal operation (an internal pull-down resistor selects normal operating mode if TEST pin is not connected).

NMI. The NMI pin provides the capability for asynchronous applying an external top priority interrupt to the MCU.This pin is falling edge sensitive. The NMI pin is provided with an on-chip pull-up resistor and schmitt trigger input characteristics.

TIMER. This is the TIMER 1 I/O pin. In input mode it is connected to the prescaler and acts as external timer clock or as control gate for the internal timer clock. In the output mode the TIMER pin outputs the data bit when a time-out occurs.

WDON. This pin selects the watchdog enabling option (hardware or software). A low level selects the hardware activated option (the watchdog is always active), a high level selects the software activated option (the watchdog can be activated by software, deactivated only by reset, thus enabling STOP mode). An internal pull-up resistance selects the software watchdog option if the WDON pin is not connected.

PA0-PA7. These 8 lines are organized as one I/O port (A). Each line may be configured under software control as an input with or without pull-up resistor, interrupt generating input with pull-up resistor, open-drain or push-pull output or as analog input for the A/D converter. Port A has a 5mA drive capability in output mode.

PB0-PB3,PB4-PB7. These 8 lines are organized as one I/O port (B). Each line may be configured under software control as an input with or without internal pull-up resistor, interrupt generating input with pull-up resistor, open-drain or push-pull output. PB0-PB3 can be programmed as analog inputs for the A/D converter while PB4-PB7 can also sink 20mA for direct LED driving. PB5-PB7 can also be used as respectively Clock, Data in and Data out pins for the on-chip SPI to carry the synchronous serial I/O signals.

COM1-COM4. These four pins are the LCD peripheral common outputs. They are the outputs of the on-chip backplane voltage generator which is used for multiplexing the 45 LCD lines allowing up to 180 segments to be driven.

S4-S48. These pins are the 45 LCD peripheral driver outputs of ST6240. Segments S1-S3 are not connected to any pin.

VLCD. Display voltage supply. It determines the high voltage level on COM1-COM4 and S4-S48 pins.

VLCD1/3, VLCD2/3. Display supply voltage inputs for determining the display voltage levels on COM1-COM4 and S4-S48 pins during multiplex operation.

PSS. This is the Power Supply Supervisor sensing pin. When the voltage applied to this pin is falling below a software programmed value the highest priority (NMI) interrupt can be generated. This pin has to be connected to the voltage to be supervised.

OSC32in and OSC32out. These pins are internally connected with the on-chip 32kHz oscillator circuit. A 32.768kHz quartz crystal can be connected between these two pins if it is necessary to provide the LCD stand-by clock and real time interrupt. OSC32in is the input pin, OSC32out is the output pin.

ST62E40,T40 EPROM/OTP DESCRIPTION

The ST62E40 is the EPROM version of the ST6240 ROM product. It is intended for use during the development of an application, and for pre-production and small volume production. The ST62T40 OTP has the same characteristics. Both include EPROM memory instead of the ROM memory of the ST6240, and so the program and constants of the program can be easily modified by the user with the ST62E40 EPROM programming board from SGS-THOMSON.

From a user point of view (with the following exception) the ST62E40,T40 products have exactly the same software and hardware features of the ROM version. An additional mode is used to configure the part for programming of the EPROM, this is set by a +12.5V voltage applied to the TEST/V_{PP} pin. The programming of the ST62E40,T40 is described in the User Manual of the EPROM Programming board.

On the ST62E40, all the 8192 bytes of PROGRAM memory are available for the user, as all the EPROM memory can be erased by exposure to UV light. On the ST62T40 (OTP) device) a reserved area for test purposes exists, as for the ST6240 ROM device. In order to avoid any discrepancy between program functionality when using the EPROM, OTP and ROM it is recommended not to use these reserved areas, even when using the ST62E40.

Other than this exception, the ST62E40,T40 parts are fully compatible with the ROM ST6240 equivalent, this datasheet thus provides only information specific to the EPROM based devices.

THE READER IS ASKED TO REFER TO THE DATASHEET OF THE ST6240 ROM-BASED DEVICE FOR FURTHER DETAILS.

EPROM ERASING

The EPROM of the windowed package of the ST62E40 may be erased by exposure to Ultra Violet light.

The erasure characteristic of the ST62E40 EPROM is such that erasure begins when the memory is exposed to light with wave lengths shorter than approximately 4000Å. It should be noted that sunlight and some types of fluorescent lamps have wavelengths in the range 3000-4000Å. It is thus recommended that the window of the ST62E40 package be covered by an opaque label to prevent unintentional erasure problems when testing the application in such an environment.
The recommended erasure procedure of the ST62E40 EPROM is exposure to short wave ultraviolet light which has wavelength 2537Å. The integrated dose (i.e. UV intensity x exposure time) for erasure should be a minimum of 15 W-sec/cm^2. The erasure time with this dosage is approximately 15 to 20 minutes using an ultraviolet lamp with 12000µW/cm^2 power rating. The ST62E40 should be placed within 2.5 cm (1 inch) of the lamp tubes during erasure.

ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings

This product contains devices to protect the inputs against damage due to high static voltages, however it is advised to take normal precaution to avoid application of any voltage higher than maximum rated voltages.

For proper operation it is recommended that V_I and V_O must be higher than V_{SS} and smaller than V_{DD}. Reliability is enhanced if unused inputs are connected to an appropriated logic voltage level (V_{DD} or V_{SS}).

Power Considerations. The average chip-junction temperature, Tj, in Celsius can be obtained from:

$Tj = T_A + PD \times RthJA$

Where : T_A = Ambient Temperature.

RthJA = Package thermal resistance (junction-to ambient).

PD = Pint + Pport.

Pint = $I_{DD} \times V_{DD}$ (chip internal power).

Pport = Port power dissipation (determinated by the user).

Symbol	Parameter	Value	Unit
V_{DD}	Supply Voltage	-0.3 to 7.0	V
V_{LCD}	Display Voltage	-0.3 to 11.0	V
V_I	Input Voltage	V_{SS} - 0.3 to V_{DD} + 0.3	V
V_O	Output Voltage	V_{SS} - 0.3 to V_{DD} + 0.3	V
I_O	Current Drain per Pin Excluding V_{DD} & V_{SS}	± 10	mA
IV_{DD}	Total Current into V_{DD} (source)	50	mA
IV_{SS}	Total Current out of V_{SS} (sink)	50	mA
Tj	Junction Temperature	150	°C
T_{STG}	Storage Temperature	-60 to 150	°C

Note : Stresses above those listed as "absolute maximum ratings" may cause permanent damage to the device . This is a stress rating only and functional operation of the device at these conditions is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

THERMAL CHARACTERISTIC

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
RthJA	Thermal Resistance	PQFP80 CQFP80-W		70		°C/W

RECOMMENDED OPERATING CONDITIONS

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
T_A	Operating Temperature	1 Suffix Version 6 Suffix Version	0 -40		70 85	°C
V_{DD}	Operating Supply Voltage		3		6	V
V_{PP}	EPROM Prog. Voltage		12	12.5	13	V
V_{LCD}	Display Voltage		3		10	V

RECOMMENDED OPERATING CONDITIONS (Continued)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
V_{RM}	RAM Retention Voltage		2			V
f_{OSC}	Oscillator Frequency (1)(4)	$V_{DD} \geq 4.5V$ $V_{DD} \geq 3V$	0.01 0.01		8.388 2	MHz
I_{INJ+}	Pin Injection Current (positive) Digital Input (2) Analog Input (3)	V_{DD} = 4.5 to 5.5V			+5	mA
I_{INJ-}	Pin Injection Current (negative) Digital Input (2) Analog Input	V_{DD} = 4.5 to 5.5V			-5	mA

Notes :

1. An oscillator frequency above 1MHz is recommanded for reliable A/D results.
2. A current of ± 5mA can be forced on each pin of the digital section without affecting the functional behaviour of the device. For a positive current injected into one pin, a part of this current (~ 10%) can be expected to flow from the neighbouring pins. A current of -5mA can be forced on one input of the analog section at a time (or -2.5mA for all inputs at a time) without affecting the conversion.
3. If a total current of +1mA is flowing into the single analog channel or if the total current flowing into all the analog inputs is of 1mA, all the conversion is resulting shifted of +1LSB. If a total positive current of +5mA is flowing into the single analog channel or if the total current flowing into all the analog inputs is of 5mA, all the conversion is resulting shifted of +2LSB.
4. Operation below 0.01 MHz is possible but requires increased supply current.

EEPROM INFORMATION

The ST62xx EEPROM single poly process has been specially developed to achieve 300.000 Write/Erase cycles and a 10 years data retention.

DC ELECTRICAL CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
V_{IL}	Input Low Level Voltage	RESET, NMI, TIMER, WDON Pin			$0.3V_{DD}$	V
V_{IH}	Input High Level Voltage	TIMER	$0.80V_{DD}$			V
		RESET, NMI, WDON Pin	$0.70V_{DD}$			V
I_{IL} I_{IH}	Input Leakage Current	RESET Pin V_{DD} =5V $V_{IN} = V_{DD}$ (1) $V_{IN} = V_{DD}$ (2) $V_{IN} = V_{SS}$ (5)			10 1 50	µA mA µA
V_{OL}	Low Level Output Voltage	TIMER, I_{OL} = 5.0mA			$0.2V_{DD}$	V
V_{OH}	High Level Output Voltage	TIMER, I_{OL} = -5.0mA	$0.65V_{DD}$			V

Notes on next page

DC ELECTRICAL CHARACTERISTICS (Continued)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
R_{PU}	Pull-up Resistor	V_{IN}=0V V_{DD}=5V WDON - NMI	40	100	200	kΩ
		RESET	200	300	500	kΩ
I_{IL} I_{IH}	Input Leakage Current	TIMER $V_{IN} = V_{DD}$ or V_{SS}		0.1	1.0	µA
I_{IL} I_{IH}	Input Leakage Current	NMI VDD = 5.0V $V_{IN} = V_{SS}$ (5) $V_{IN} = V_{DD}$			 100 1.0	µA
I_{IL} I_{IH}	Input Leakage Current	WDON VDD = 5V $V_{IN} = V_{SS}$ (5) $V_{IN} = V_{DD}$			 100 1.0	µA
I_{DD}	Supply Current RUN Mode	f_{OSC} = 8MHz, I_{LOAD} = 0mA V_{DD} = 5.5V		4	7	mA
	Supply Current WAIT Mode (4)	f_{OSC} = 8MHz, I_{LOAD} = 0mA V_{DD} = 5.0V		1	3	mA
	Supply Current RESET Mode	f_{OSC} = 8MHz, $V_{RESET} = V_{SS}$		1	7	mA
	Supply Current STOP Mode (3)	I_{LOAD} = 0mA V_{DD} = 5.5V		1	10	µA

Notes :

1. No Watchdog Reset activated.
2. Reset generated by Watchdog.
3. When the watchdog function is actvated the STOP instruction is deactivated. WAIT instruction is automatically executed.
4. All on-chip peripherals in OFF state
5. Pull-up resistor

AC ELECTRICAL CHARACTERISTICS

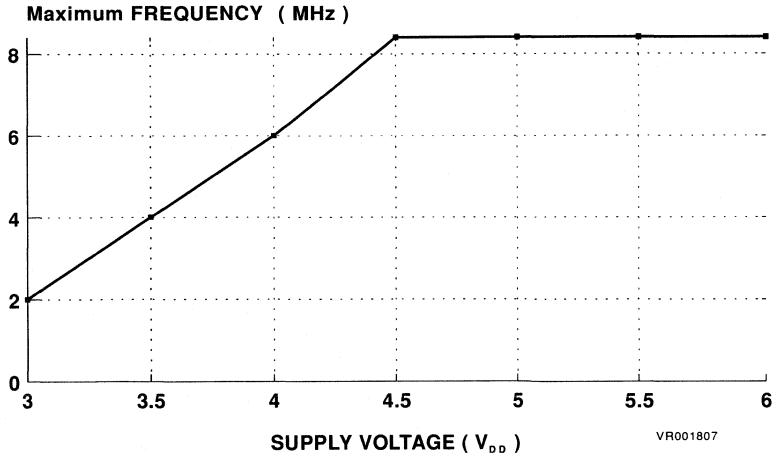
(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
f_{OSC}	Oscillator Frequency (2)	$V_{DD} \geq 4.5V$ $V_{DD} \geq 3V$	0.01		8.388 2	MHz
t_{SU}	Oscillator Start-up Time	$C_{L1} = C_{L2}$ = 22pF - crystal		5	10	ms
t_{SR}	Supply Rise Time	10% to 90%	0.01		100	
t_{REC}	Supply Recovery Time (1)		100			
T_W	Minimum Pulse Width	NMI Pin V_{DD} = 5V	100			ns
		RESET Pin	100			ns
T_{WEE}	EEPROM Write Time	T_A = 25°C One Byte T_A = 85°C One Byte		5 15	10 25	ms
Endurance	EEPROM WRITE/ERASE Cycles	Q_A L_{OT} Acceptance Criteria	300.000	> 1 million		cycles
Retention	EEPROM Data Retention	T_A = 55°C	10			years
C_{IN}	Input Capacitance	All Inputs Pins			10	pF

Notes:

1. Period for which V_{DD} has to be connected or at 0V to allow internal Reset function at next power-up.
2. Operation below 0.01 MHz is possible but requires increased supply current.

Maximum Operating FREQUENCY (Fmax) Versus SUPPLY VOLTAGE (V_{DD})



I/O PORTS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
C_{OUT}	Output Capacitance	All Outputs Pins			10	pF
V_{IL}	Input Low Level Voltage	I/O Pins			$0.3V_{DD}$	V
V_{IH}	Input High Level Voltage	I/O Pins	$0.7V_{DD}$			V
V_{OL}	Low Level Output Voltage	I/O Pins, I_O = 10µA (sink)			0.1	V
		I/O Pins, $I_{OL} = V_{DD}$x1mA V_{DD} = 4.5 to 6V			$0.16xV_{DD}$	V
		I/O Pins, I_{OL} = 1.6mA V_{DD} = 3V			0.4	V
	Low Level Output Voltage, PB4-PB7 Only	I/O Pins, $I_{OL} = V_{DD}$x2mA V_{DD} = 4.5 to 6V			$0.16xV_{DD}$	V
		I/O Pins, I_{OL} = 3.2mA V_{DD} = 3V			0.4	V
		I/O Pins, $I_{OL} = V_{DD}$x4mA V_{DD} = 4.5 to 6V			$0.26xV_{DD}$	V
		I/O Pins, I_{OL} = 6.4mA V_{DD} = 3V			0.8	V
V_{OH}	High Level Output Voltage	I/O Pins, I_O = -10µA (source)	V_{DD}-0.1			V
		I/O Pins, $I_{OL} = -V_{DD}$x1mA V_{DD} = 5.0V	$0.6xV_{DD}$			V
I_{IL} I_{IH}	Input Leakage Current	I/O Pins,(1)		0.1	1.0	µA
R_{PU}	Pull-up Resistor	I/O Pins V_{IN} = 0V, V_{DD} = 5.0V	40	100	200	kΩ

Note 1. Pull-up resistor off

SPI ELECTRICAL CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
F_{CL}	Clock Frequency	applied on PB5/SCL			500	kHz
t_{SU}	Set-up Time	applied on PB6/Sin		50		ns
t_h	Hold Time	applied on PB6/Sin		100		ns

A/D CONVERTER CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
Res	Resolution (3)			8		Bit
A_{TOT}	Total Accuracy (3)	f_{OSC} > 1.2 MHz f_{OSC} > 32kHz			± 2 ±4	LSB
t_C (1)	Conversion Time	f_{OSC} = 8MHz		70		µs
V_{AN}	Conversion Range		V_{SS}		V_{DD}	V
ZIR	Zero Input Reading	Conversion result when $V_{IN} = V_{SS}$	00			Hex
FSR	Full Scale Reading	Conversion result when $V_{IN} = V_{DD}$			FF	Hex
AD_I	Analog Input Current During Conversion	V_{DD}= 4.5V			1.0	µA
AC_{IN} (2)	Analog Input Capacitance			2	5	pF
ASI	Analog Source Impedance				30	kΩ
SSI	Analog Reference Supply Impedence				2	kΩ

Notes:

1. With oscillator frequencies less than 1MHz, the A/D Converter accuracy is decreased.
2. Excluding Pad Capacitance
3. Noise at $V_{DD}, V_{SS} \leq 10mV$

TIMER CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
t_{RES}	Resolution		$\frac{12}{f_{OSC}}$			second
f_{IN}	Input Frequency on TIMER Pin				$\frac{f_{OSC}}{8}$	MHz
t_W	Pulse Width at TIMER Pin	$V_{DD} \geq 3V$ $V_{DD} \geq 4.5V$	1 125			µs ns

PSS ELECTRICAL CHARACTERISTICS

(T_A = -40 to +85°C unless otherwhise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
V_{PSS}	PSS pin Input Voltage		V_{SS}		V_{DD}	V
I_{PSS}	PSS pin Input Current	PSS RUN PSS STOP V_{PSS} = 5V, T_A = 25°C			350 1	µA

LCD ELECTRICAL CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

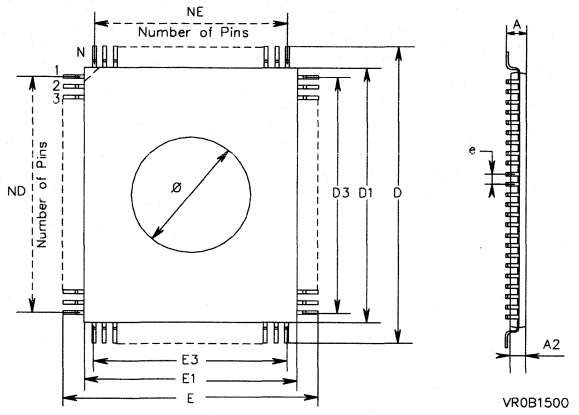
Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
f_{FR}	Frame Frequency	1/4 Duty f_{OSC} = 1, 2, 4, 8MHz	16		128	Hz
V_{OS}	DC Offset Voltage(1)	$V_{LCD} = V_{DD}$, no load			50	mV
V_{OH}	COM High Level, Output Voltage	I = 100µA V_{LCD} = 5V	4.5V			V
V_{OL}	COM Low Level, Output Voltage	I = 100µA V_{LCD} = 5V			0.5V	V
V_{OH}	SEG High Level, Output Voltage	I = 50µA V_{LCD} = 5V	4.5V			V
V_{OL}	SEG Low Level, Output Voltage	I = 50µA V_{LCD} = 5V			0.5V	V
V_{LCD}	Display Voltage	Note 2	3		10	V

Notes :

1. The DC offset voltage refers to all segment and common outputs. It is the difference between the measured voltage value and nominal value for every voltage level. Ri of voltage meter must be greater than or equal to 100MΩ.
2. An external resistances network is required when $V_{LCD} \leq 4.5V$.

PACKAGE MECHANICAL DATA

Figure 3. ST62E40 80 Pin Ceramic Quad Flat Package with Window



Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A		3.55			0.14	
A2		3.40			0.133	
D		23.90			0.941	
D1		20.00			0.787	
D3		18.40			0.724	
E		17.90			0.705	
E1		14.00			0.551	
E3		12.00			0.472	
Ø		7.62			0.3	
e		0.80			0.032	
Number of Pins						
ND	24					
NE	16					
N	80					

ORDERING INFORMATION TABLE

Sales Types	Temperature Range	Package
ST62E40G1	0 to + 70°C	CQFP80-W
ST62T40Q6	-40 to + 85°C	PQFP80

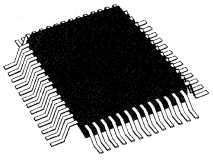
SGS-THOMSON MICROELECTRONICS

ST6242

8-BIT HCMOS MCU WITH LCD DRIVER, AND A/D CONVERTER

PRELIMINARY DATA

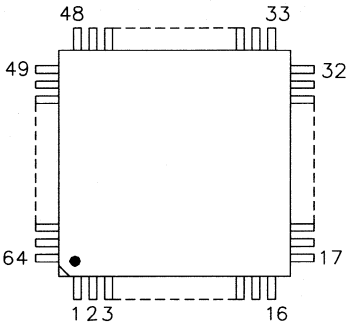
- 3 to 6V supply operating range
- 8.4MHz Maximum Clock Frequency
- -40 to +85°C Operating Temperature Range
- Run, Wait & Stop Modes
- 5 different interrupt vectors
- Look-up table capability in ROM
- User ROM: 7948 bytes
  Data RAM: 128 bytes
  LCD RAM: 24 bytes
- PQFP64 package
- 10 fully software programmable I/O as:
  - Input with/without pull-up resistor
  - Input with interrupt generation
  - Open-Drain or Push-pull outputs
  - Analog Inputs (6 pins)
- 4 I/O lines can sink up to 20mA for direct LED or TRIAC driving and have SPI alternate functions
- 8-bit counter with 7-bit programmable prescaler
- Software activated digital watchdog
- 8-bit A/D converter with up to 6 analog inputs
- 8-bit synchronous Serial Peripheral Interface (SPI)
- LCD driver with 40 segment outputs, 4 back-plane outputs and selectable duty cycle for up to 160 LCD segments direct driving
- One external not maskable interrupt
- 9 powerful addressing modes
- The accumulator, the X, Y, V & W registers, the port and peripherals data & control registers are addressed in the data space as RAM locations.
- The ST62E42 is the EPROM version, ST62T42 is the OTP version
- Development tool: ST6242-EMU connected via RS232 to an MS-DOS Personal Computer



PQFP64

(Ordering Information at the end of the datasheet)

Figure 1. 64 Pin Quad Flat Pack (QFP) Package Pinout



ST6242 Pin Description

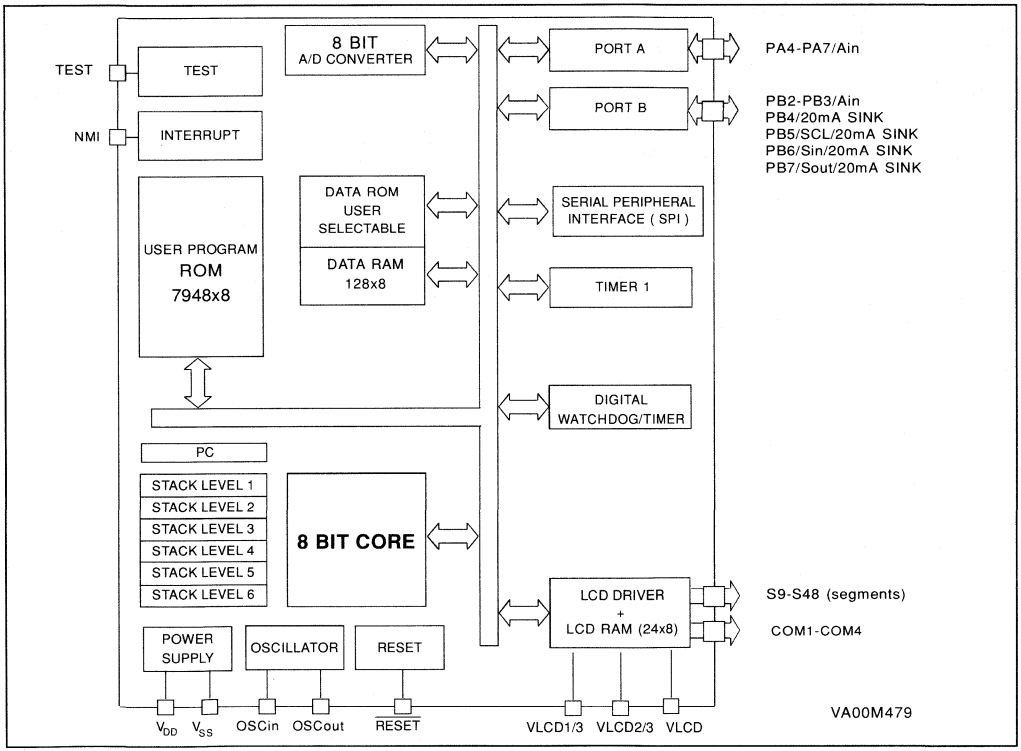
Pin number	Pin name	Pin number	Pin name	Pin number	Pin name	Pin number	Pin name
1	S45	17	V_{DD}	33	S13	49	S29
2	S46	18	V_{SS}	34	S14	50	S30
3	S47	19	$\overline{RESET}$	35	S15	51	S31
4	S48	29	OSCout	36	S16	52	S32
5	COM4	21	OSCin	37	S17	53	S33
6	COM3	22	NMI	38	S18	54	S34
7	COM2	23	PB7/Sout (1)	39	S19	55	S35
8	COM1	24	PB6/Sin (1)	40	S20	56	S36
9	VLCD1/3	25	PB5/SCL (1)	41	S21	57	S37
10	VLCD2/3	26	PB4 (1)	42	S22	58	S38
11	VLCD	27	PB3/Ain	43	S23	59	S39
12	PA7/Ain	28	PB2/Ain	44	S24	60	S40
13	PA6/Ain	29	S9	45	S25	61	S41
14	PA5/Ain	30	S10	46	S26	62	S42
15	PA4/Ain	31	S11	47	S27	63	S43
16	TEST/V_{PP}	32	S12	48	S28	64	S44

Note 1: 20mA SINK

GENERAL DESCRIPTION

The ST6242 microcontroller is a member of the 8-bit HCMOS ST62xx family, a series of devices oriented to low-medium complexity applications. All ST62xx members are based on a building block approach: a common core is associated with a combination of on-chip peripherals (macrocells). The macrocells of the ST6242 are: a high performance LCD controller/driver with 40 segment outputs and 4 backplanes able to drive up to 160 segments. A Timer peripheral including an 8-bit counter with a 7-bit software programmable prescaler (Timer), the digital watchdog timer (DWD), an 8-bit A/D Converter with up to 6 analog inputs, a Power Supply Supervisor and an 8-bit synchronous Serial Peripheral Interface (SPI). Thanks to these peripherals the ST6242 is well suited for general purpose, automotive, security, appliance and industrial applications. The ST62E42 EPROM version is available for prototypes and low-volume production, an OTP version is also available (see separate datasheet).

Figure 2. ST6242 Block Diagram



Note: Ain = Analog Input

PIN DESCRIPTION

V_{DD} and V_{SS}. Power is supplied to the MCU using these two pins. V_{DD} is power and V_{SS} is the ground connection.

OSCin and OSCout. These pins are internally connected with the on-chip oscillator circuit. A quartz crystal or a ceramic resonator can be connected between these two pins in order to allow the correct operation of the MCU with various stability/cost trade-offs. OSCin is the input pin, OSCout is the output pin. An external clock signal can be applied to OSCin.

$\overline{\text{RESET}}$. The active low $\overline{\text{RESET}}$ pin is used to restart the microcontroller at the beginning of its program. The $\overline{\text{RESET}}$ pin is provided with an on-chip pull-up resistor and schmitt trigger input characteristics.

TEST. The TEST must be held at V_{SS} for normal operation (an internal pull-down resistor selects normal operating mode if TEST pin is not connected).

NMI. The NMI pin provides the capability for asynchronous applying an external top priority interrupt to the MCU.This pin is falling edge sensitive. The NMI pin is provided with an on-chip pull-up resistor and schmitt trigger input characteristics.

PA4-PA7. These 4 lines are organized as one I/O port (A). Each line may be configured under software control as an input with or without pull-up resistor, interrupt generating input with pull-up resistor, open-drain or push-pull output or as analog input for the A/D converter. Port A has a 5mA drive capability in output mode.

PB0-PB3,PB4-PB7. These 6 lines are organized as one I/O port (B). Each line may be configured under software control as an input with or without internal pull-up resistor, interrupt generating input with pull-up resistor, open-drain or push-pull output. PB2-PB3 can be programmed as analog inputs for the A/D converter while PB4-PB7 can also sink 20mA for direct LED driving. PB5-PB7 can also be used as respectively Clock, Data in and Data out pins for the on-chip SPI to carry the synchronous serial I/O signals.

COM1-COM4. These 4 pins are the LCD peripheral common outputs. They are the outputs of the on-chip backplane voltage generator which is used for multiplexing the 40 LCD lines allowing up to 160 segments to be driven.

S9-S48. These pins are the 40 LCD peripheral driver outputs of ST6242. Segments S1-S8 are not connected to any pin.

VLCD. Display voltage supply. It determines the high voltage level on COM1-COM4 and S9-S48 pins.

VLCD1/3, VLCD2/3. Resistor network nodes for determining the intermediate display voltage levels on COM1-COM4 and S9-S48 pins during multiplex operation.

ST62xx CORE

The core of the ST62xx Family is implemented independently from the I/O or memory configuration. Consequently, it can be treated as an independent central processor communicating with I/O and memory via internal addresses, data, and control busses. The in-core communication is arranged as shown in Figure 3; the controller being externally linked to both the reset and the oscillator, while the core is linked to the dedicated on-chip macrocells peripherals via the serial data bus and indirectly for interrupt purposes through the control registers.

Registers

The ST62xx Family core has six registers and three pairs of flags available to the programmer. They are shown in Figure 4 and are explained in the following paragraphs.

Accumulator (A). The accumulator is an 8-bit general purpose register used in all arithmetic calculations, logical operations, and data manipulations. The accumulator is addressed in the data space as RAM location at address FFh. Accordingly, the ST62xx instruction set can use the accumulator as any other register of the data space.

Figure 4. ST62xx Core Programming Model

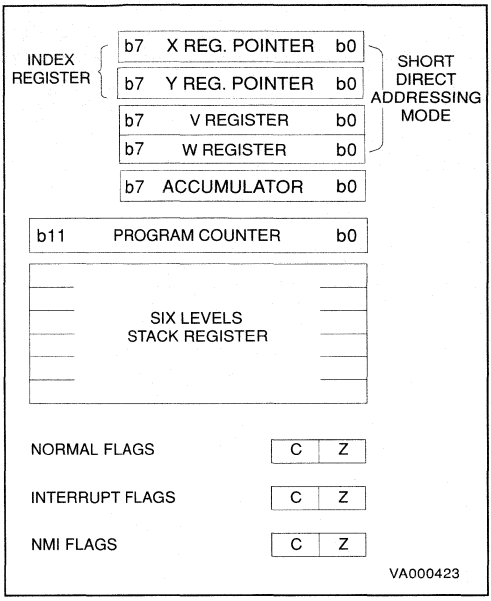
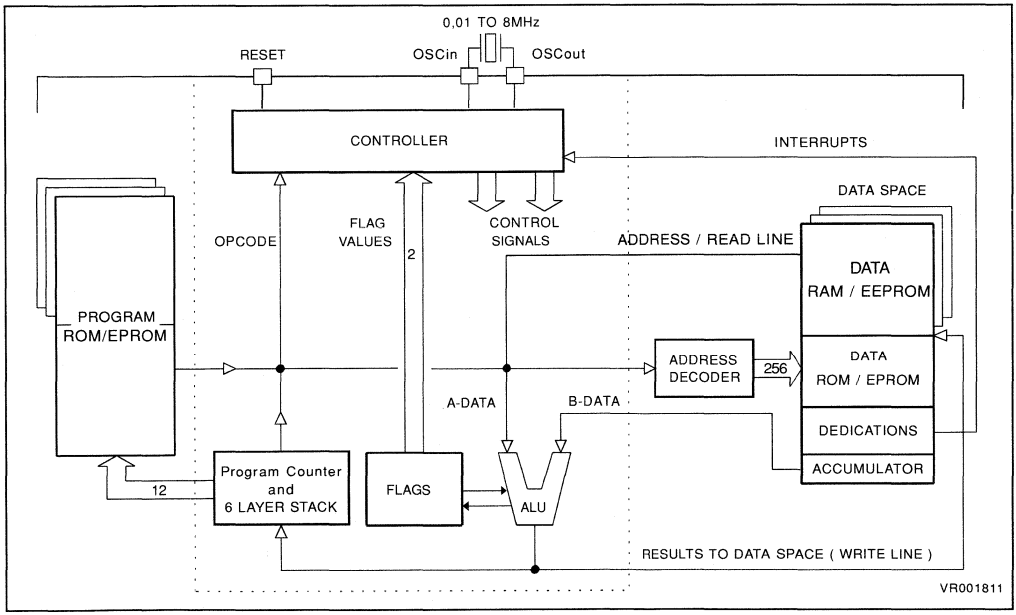


Figure 3.ST62xx Core Block Diagram



ST62xx CORE (Continued)

Indirect Registers (X, Y). These two indirect registers are used as pointers to memory locations in the data space. They are used in the register-indirect addressing mode. These registers can be addressed in the data space as RAM locations at addresses 80h (X) and 81h (Y). They can also be accessed with the direct, short direct, or bit direct addressing modes. Accordingly, the ST62xx instruction set can use the indirect registers as any other register of the data space.

Short Direct Registers (V, W). These two registers are used to save one byte in short direct addressing mode. These registers can be addressed in the data space as RAM locations at addresses 82h (V) and 83h (W). They can also be accessed with the direct and bit direct addressing modes. Accordingly, the ST62xx instruction set can use the short direct registers as any other register of the data space.

Program Counter (PC)

The program counter is a 12-bit register that contains the address of the next ROM location to be processed by the core. This ROM location may be an opcode, an operand, or an address of operand. The 12-bit length allows the direct addressing of 4096 bytes in the program space. Nevertheless, if the program space contains more than 4096 locations, as for the ST6242, the further program space can be addressed by using the Program ROM Page register.
The PC value is incremented after it is read from the address of the current instruction. To execute relative jumps the PC and the offset are shifted through the ALU, where they will be added, and the result is shifted back into the PC. The program counter can be changed in the following ways:

- JP (Jump) instruction . . . PC=Jump address
- CALL instruction . . . . PC=Call address
- Relative Branch instructions . . . . . PC=PC ± offset
- Interrupt . . . . . PC=Interrupt vector
- Reset . . . . . PC=Reset vector
- RET & RETI instructions . PC=Pop (stack)
- Normal instruction . . . . PC=PC+1

Flags (C, Z)

The ST62xx core includes three pairs of flags that correspond to 3 different modes: normal mode, interrupt mode and Non-Maskable Interrupt-Mode. Each pair consists of a CARRY flag and a ZERO flag. One pair (CN, ZN) is used during normal operation, one pair is used during the interrupt mode (CI, ZI) and one is used during the not-maskable interrupt mode (CNMI, ZNMI).

The ST62xx core uses the pair of flags that correspond to the actual mode: as soon as an interrupt (resp. a Non-Maskable-Interrupt) is generated, the ST62xx core uses the interrupt flags (resp. the NMI flags) instead of the normal flags. When the RETI instruction is executed, the normal flags (resp. the interrupt flags) are restored if the MCU was in the normal mode (resp. in the interrupt mode) before the interrupt. It should be observed that each flag set can only be addressed in its own mode (Not-maskable interrupt, normal interrupt or main mode). The flags are not cleared during the context switching and so remain in the state they were at the exit of the last mode switch.

The Carry flag is set when a carry or a borrow occurs during arithmetic operations, otherwise it is cleared. The Carry flag is also set to the value of the bit tested in a bit test instruction, and participates in the rotate left instruction.

The Zero flag is set if the result of the last arithmetic or logical operation was equal to zero, otherwise it is cleared.

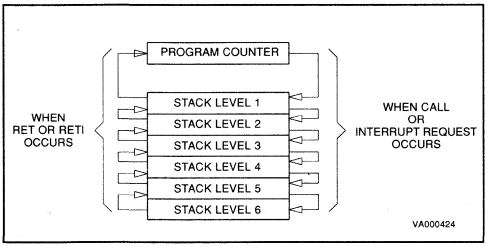
The switching between the three sets of Flags is automatically performed when an NMI, an interrupt or a RETI instruction occurs. As the NMI mode is automatically selected after the reset of the MCU, the ST62xx core uses at first the NMI flags.

ST62xx CORE (Continued)

Stack

The ST62xx core includes a true LIFO hardware stack that eliminates the need for a stack pointer. The stack consists of six separate 12-bit RAM locations that do not belong to the data space RAM area. When a subroutine call (or interrupt request) occurs, the contents of each level is shifted into the next level while the content of the PC is shifted into the first level (the value of the sixth level will be lost). When a subroutine or interrupt return occurs (RET or RETI instructions), the first level register is shifted back into the PC and the value of each level is popped back into the previous level. These two operating modes are described in Figure 5. Since the accumulator, as all other data space registers, is not stored in the stack, the handling of these registers should be performed inside the subroutine. The stack pointer will remain in its deepest position if more than 6 calls or interrupts are executed, so that the last return address will be lost. It will also remain in its highest position if the stack is empty and a RET or RETI is executed. In this case the next instruction will be executed.

Figure 5. Stack Operation



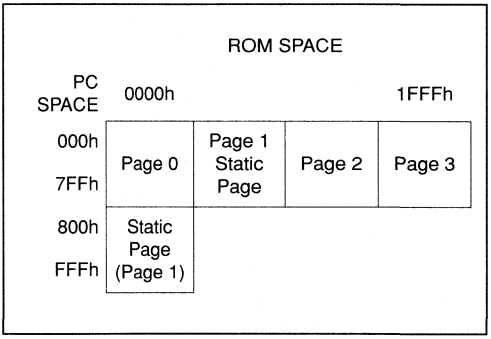
MEMORY SPACES

The MCU operates in three different memory spaces: program space, data space, and stack space. A description of these spaces is shown in the following figures.

Program Space

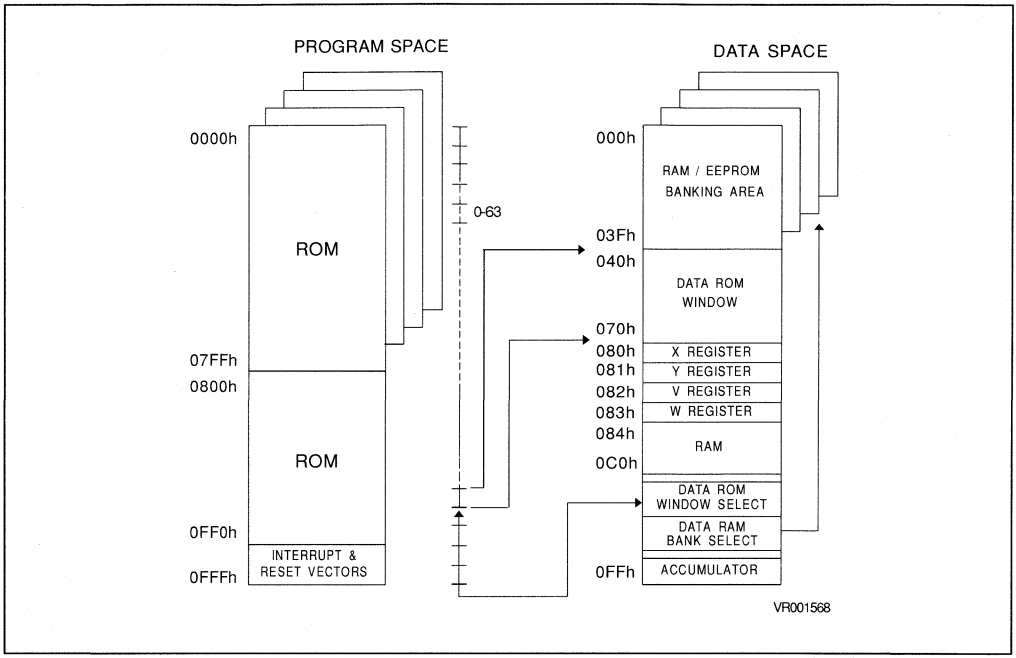
The program space is physically implemented in the ROM memory and includes all the instructions that are to be executed, as well as the data required for the immediate addressing mode instructions, the reserved test area and the user vectors. It is addressed by the 12-bit Program Counter register (PC register) and so the ST62xx core can directly address up to 4K bytes of Program Space. Nevertheless, the Program Space can be extended by the addition of 2Kbyte ROM banks as it is shown in the following figure in which the ST6242 8Kbyte memory is described.

Figure 6. ST6242 8Kbytes Program Space Addressing Description



MEMORY SPACES (Continued)

Figure 7. ST62xx Memory Addressing Description Diagram



These banks are addressed in the 000h-7FFh locations of the Program Space by the Program Counter and by writing the appropriate code in the Program ROM Page Register (PRPR register) located at address CAh of the Data Space. Because interrupts and common subroutines should be available all the time, only the lower 2K byte of the 4K program space are bank switched while the upper 2K byte can be seen as static page. Table 2 gives the different codes that allow the selection of the corresponding banks. Note that, from the memory point of view, Page 1 and the Static Page represent the same physical memory: it is only two different ways of addressing the same locations. On the ST6242 a total of 8192 bytes of ROM have been implemented; 7948 are available as user ROM while 244 are reserved for SGS-THOMSON test purposes.

Table 1. ST6242 Program ROM Memory Map

ROM Page	Device Address	Description
Page 0	0000h-007Fh 0080h-07FFh	Reserved User ROM
Page 1 "STATIC"	0800h-0F9Fh 0FA0h-0FEFh 0FF0h-0FF7h 0FF8h-0FFBh 0FFCh-0FFDh 0FFEh-0FFFh	User ROM Reserved Interrupt Vectors Reserved NMI Vector Reset Vector
Page 2	0000h-000Fh 0010h-07FFh	Reserved User ROM
Page 3	0000h-000Fh 0010h-07FFh	Reserved User ROM

MEMORY SPACES (Continued)

Data Space

The instruction set of the ST62xx core operates on a specific space, named Data Space, that contains all the data necessary for the processing of the program. The Data Space allows the addressing of RAM memory, ST62xx core/peripheral registers, and read-only data such as constants and look-up tables.

Data ROM

All the read-only data is physically implemented in the ROM memory in which the Program Space is also implemented. The ROM memory contains consequently the program to be executed, the constants and the look-up tables needed for the program.

The locations of Data Space in which the different constants and look-up tables are addressed by the ST62xx core can be considered as being a 64-byte window through which it is possible to access to the read-only data stored in the ROM memory .

Data RAM Addressing

The ST6242 offers 128 bytes of data RAM memory. 64 bytes of RAM are directly addressed in data space in the range 084h-0BFh (static space). The additional RAM is addressed using the banks of 64 bytes located between addresses 00h and 3Fh.

Additionally 24 bytes of RAM devoted to LCD dat are available from E0h to F7h and are not banked.

Stack Space

The stack space consists of six 12 bit registers that are used for stacking subroutine and interrupt return addresses plus the current program counter register.

Figure 8. ST6242 Data Memory Space

DATA RAM/EEPROM BANK AREA	000h 03Fh
DATA ROM WINDOW AREA	040h 07Fh
X REGISTER	080h
Y REGISTER	081h
V REGISTER	082h
W REGISTER	083h
DATA RAM 60 BYTES	084h 0BFh
PORT A DATA REGISTER	0C0h
PORT B DATA REGISTER	0C1h
SPI INT. DISABLE REGISTER	0C2h*
RESERVED	0C3h
PORT A DIRECTION REGISTER	0C4h
PORT B DIRECTION REGISTER	0C5h
RESERVED	0C6h
RESERVED	0C7h
INTERRUPT OPTION REGISTER	0C8h*
DATA ROM WINDOW REGISTER	0C9h*
PROGRAM ROM PAGE REGISTER	0CAh*
DATA RAM PAGE REGISTER	0CBh*
PORT A OPTION REGISTER	0CCh
RESERVED	0CDh
PORT B OPTION REGISTER	0CEh
RESERVED	0CFh
A/D DATA REGISTER	0D0h
A/D CONTROL REGISTER	0D1h
TIMER PRESCALER REGISTER	0D2h
TIMER COUNTER REGISTER	0D3h
TIMER STATUS/CONT REGISTER	0D4h
RESERVED	0D5h 0D6h 0D7h
WATCHDOG REGISTER	0D8h
RESERVED	0D9h 0DAh 0DBh
LCD MODE CONTROL REGISTER	0DCh
SPI DATA REGISTER	0DDh
RESERVED	0DEh
WRITE: 40h	0DFh
LCD RAM	0E0h 0F7h
DATA RAM 7 BYTES	0F8h 0FEh
ACCUMULATOR	0FFh

* WRITE ONLY REGISTER

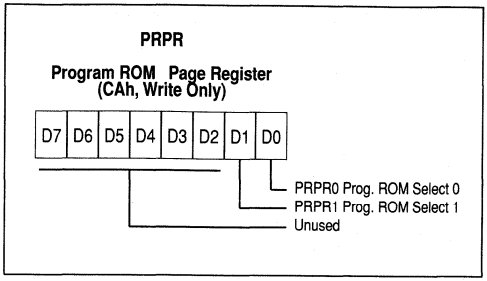
MEMORY SPACES (Continued)

Program ROM Page Register (PRPR)

The PRPR register can be addressed like a RAM location in the Data Space at the address CAh; nevertheless it is a write only register that cannot be accessed with single-bit operations. This register is used to select the 2-Kbyte ROM bank of the Program Space that will be addressed. The number of the page has to be loaded in the PRPR register. Refer to the Program Space description for additional information concerning the use of this register. The PRPR register is not modified when an interrupt or a subroutine occurs.

Care is required when handling the PRPR register as it is write only. For this reason, it is not allowed to change the PRPR contents while executing interrupt service routine, as the service routine cannot save and then restore its previous content. This operation may be necessary if common routines and interrupt service routines take more than 2K bytes; in this case it could be necessary to divide the interrupt service routine into a (minor) part in the static page (start and end) and to a second (major) part in one of the dynamic pages. If it is impossible to avoid the writing of this register in interrupt service routines, an image of this register must be saved in a RAM location, and each time the program writes to the PRPR it must write also to the image register. The image register must be written before PRPR, so if an interrupt occurs between the two instructions the PRPR is not affected.

Figure 9. Program ROM Page Register



D7-D2. These bits are not used.

PRPR1-PRPR0. These are the program ROM banking bits and the value loaded selects the corresponding page to be addressed in the lower part of the 4K program address space as specified in Table 2.

Table 2. ST6242 8Kbytes Program ROM Page Register Coding

PRPR1	PRPR0	PC bit 11	Memory Page
X	X	1	Static Page (Page1)
0	0	0	Page 0
0	1	0	Page 1 (Static Page)
1	0	0	Page 2
1	1	0	Page 3

This register is undefined on reset. Neither read nor single bit instructions may be used to address this register.

Note:

Only the lower part of address space is bank switched because interrupt vectors and common subroutines should be available at all times. The reason of this structure is due to the fact that it is not possible to jump from one dynamic page to another except by jumping back to the static page, changing contents of PRPR, and then jumping to a different dynamic page.

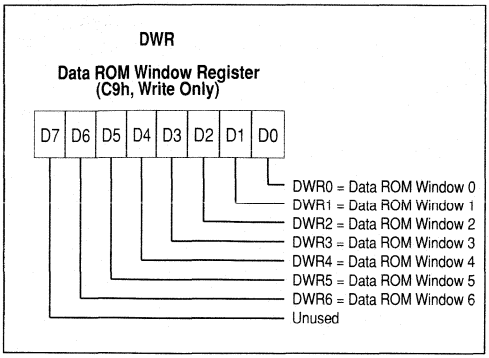
MEMORY SPACES (Continued)

Data Window register (DWR)

The Data ROM window is located from address 040h to address 7Fh in the Data space. It allows the direct reading of 64 consecutive bytes located anywhere in the ROM memory between the addresses 0000h and 1FFFh. All the bytes of the ROM memory can be used to store either instructions or read-only data. Indeed, the window can be moved by step of 64 bytes along the ROM memory in writing the appropriate code in the Write-only Data Window register (DWR register, location C9h).

The DWR register can be addressed like a RAM location in the Data Space at the address C9h, nevertheless it is a write only register that cannot be accessed with single-bit operations. This register is used to move the 64-byte read-only data window (from the 40h address to 7Fh address of the Data Space) up and down the ROM memory of the MCU in steps of 64 bytes. The effective address of the byte to be read as a data in the ROM memory is obtained by the concatenation of the 6 least significant bits of the register address given in the instruction (as least significant bits) and the content of the DWR register (as most significant bits, see Figure 10). So when addressing location 40h of dataspace, and 0 is loaded in the DWR register, the phisycal addressed location in ROM is 00h. The DWR register is not cleared at reset, therefore it must be written to before the first access to the Data ROM window area.

Figure 11. Data ROM Window Register



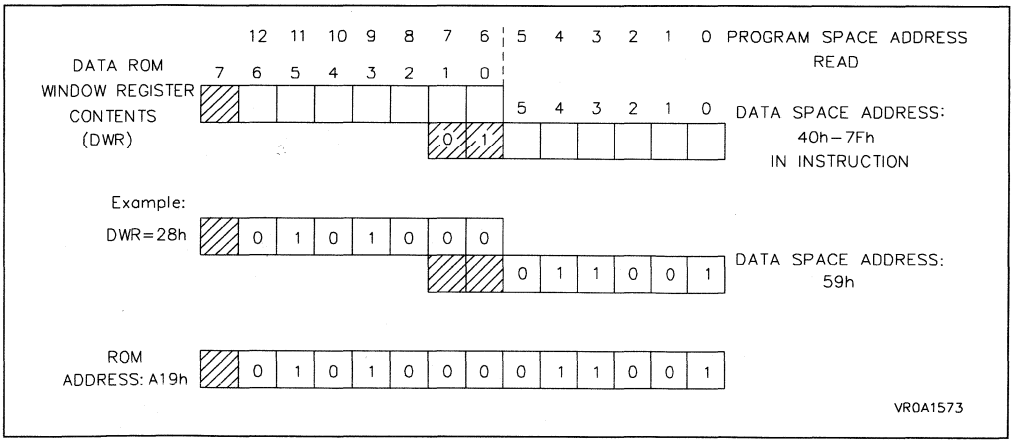
D7. This bit is not used.

DWR6-DWR0. These are the Data ROM Window bits that correspond to the upper bits of the data ROM space.

This register is undefined on reset. Neither read nor single bit instructions may be used to address this register.

Note: Care is required when handling the DWR register as it is write only. For this reason, it is not allowed to change the DWR contents while executing interrupt service routine, as the service routine cannot save and then restore its previous content. If it is impossible to avoid the writing of this register in the interrupt service routine, an image of this register must be saved in a RAM location, and each time the program writes to the DWR it must write also to the image register. The image register must be written first, so if an interrupt occurs between the two instructions the DWR is not affected.

Figure 10. Data ROM Window Memory Addressing



MEMORY SPACES (Continued)

Data RAM Bank Register (DRBR)

The selection of the bank is made by programming the Data RAM Bank Switch register (DRBR register) located at address CBh of the Data Space. The number of the selected bank is equal to the bit content of the DRBR register. In this way each bank of RAM can be selected 64 bytes at a time. No more than one bank should be set at a time.

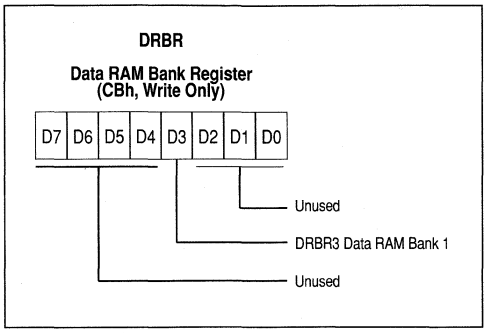
The DRBR register can be addressed like a RAM location in the Data Space at the address CBh; nevertheless it is a write only register that cannot be accessed with single-bit operations. This register is used to select the desired 64-byte RAM bank of the Data Space. The number of the bank has to be loaded in the DRBR register and the instruction has to point to the selected location as if it was in bank 0 (from 00h address to 3Fh address). This register is not cleared during the MCU initialization, therefore it must be written before the first access to the Data Space bank region. Refer to the Data Space description for additional information. The DRBR register is not modified when a interrupt or a subroutine occurs.

The following table 3 summarizes how to set the data RAM bank register in order to select the various banks or pages.

Table 3. Data RAM Register Set-up

DRBR Value	Selection
08h	RAM Page 1

Figure 12. Data RAM Bank Register



D7-D4. These bits are not used.

DRBR3. This bit, when set, will select the RAM page.

D2-D0. These bits are not used.

This register is undefined on reset. Neither read nor single bit instructions may be used to address this register.

Notes:

Care is required when handling the DRBR register as it is write only. For this reason, it is not allowed to change the DRBR contents while executing interrupt service routine, as the service routine cannot save and then restore its previous content. If it is impossible to avoid the writing of this register in interrupt service routine, an image of this register must be saved in a RAM location, and each time the program writes to DRBR it must write also to the image register. The image register must be written first, so if an interrupt occurs between the two instructions the DRBR is not affected.

In DRBR Register, *only 1 bit must be set.* Otherwise two or more pages are enabled in parallel, producing errors.

TEST MODE

For normal operation the TEST pin must be held low. An on-chip 100kΩ pull-down resistor is internally connected to the TEST pin.

INTERRUPTS

The ST62xx core can manage 4 different maskable interrupt sources, plus one non-maskable interrupt source (top priority level interrupt). Each source is associated with a particular interrupt vector that contains a Jump instruction to the related interrupt service routine. Each vector is located in the Program Space at a particular address.
When a source provides an interrupt request, and the request processing is also enabled by the ST62xx core, then the PC register is loaded with the address of the interrupt vector (i.e. of the Jump instruction). Finally, the PC is loaded with the address of the Jump instruction and the interrupt routine is processed.

The ST6242 microcontroller has six different interrupt sources associated to different interrupt vectors as described in Table 4.

Table 4. Interrupt Vectors - Sources Relationship

Interrupt Source	Associated Vector	Vector Address
NMI Pins	Interrupt Vector #0 (NMI)	(FFCh-FFDh)
SPI Peripheral	Interrupt Vector #1	(FF6h-FF7h)
Port A & B Pins	Interrupt Vector #2	(FF4h-FF5h)
TIMER	Interrupt Vector #3	(FF2h-FF3h)
ADC Peripheral	Interrupt Vector #4	(FF0h-FF1h)

Interrupts Vectors Description

The ST62xx core includes 5 different interrupt vectors in order to branch to 5 different interrupt routines in the static page of the Program Space:

- The interrupt vector associated with the non-maskable interrupt source is named interrupt vector #0. It is located at addresses FFCh,FFDh in the Program Space. On ST6242 this vector is associated with the external falling edge sensitive interrupt pin (NMI). An on-chip 100kΩ pull-up resistor is internally connected to the NMI pin.
- The interrupt vector located at the addresses FF6h, FF7h is named interrupt vector #1. It is associated with SPI peripheral and can be programmed by software to generate an interrupt request after the falling edge or low level of the eighth external clock pulse according to the code loaded in the Interrupt Option Register (IOR).
- The interrupt vector located at the addresses FF4h, FF5h is named interrupt vector #2. It is associated with Port A and B pins and can be programmed by software either in the falling edge detection mode or in the rising edge detection mode according to the code loaded in the Interrupt Option Register (IOR).
- The interrupt vector located at the addresses FF2h, FF3h is named interrupt vector #3. It is associated with Timer.
- The interrupt vector located at the addresses FF0h, FF1h is named interrupt vector #4. It is associated with the A/D converter peripheral.

All the on-chip peripherals (refer to their descriptions for further details) have an interrupt request flag bit (TMZ for timer, EOC for A/D, etc.), this bit is set to one when the device wants to generate an interrupt request and a mask bit (ETI for timer, EAI for A/D, etc.) that must be set to one to allow the transfer of the flag bit to the Core.

Interrupt Priority

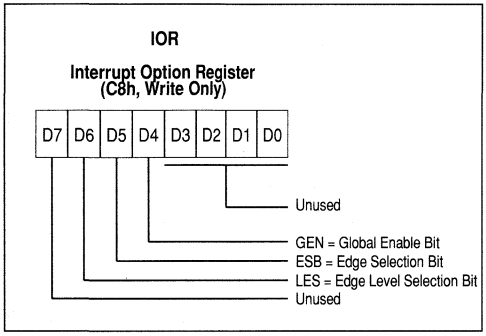
The non-maskable interrupt request has the highest priority and can interrupt any other interrupt routines at any time, nevertheless the four other interrupts cannot interrupt each other. If more than one interrupt request is pending, they are processed by the ST62xx core according to their priority level: vector #1 has the higher priority while vector #4 the lower. The priority of each interrupt source is fixed.

INTERRUPTS (Continued)

Interrupt Option Register

The Interrupt Option Register (IOR register, location C8h) is used to enable/disable the individual interrupt sources and to select the operating mode of the external interrupt inputs. This register can be addressed in the Data Space as RAM location at the address C8h, nevertheless it is a write-only register that cannot be accessed with single-bit operations. The operating modes of the external interrupt inputs associated to interrupt vectors #1 and #2 are selected through bits 5 and 6 of the IOR register.

Figure 13. Interrupt Option Register



D7. This bit is not used.

LES. Level/Edge Selection Bit. When this bit is set to one, the interrupt #1 (SPI) is low level sensitive, when cleared to zero the negative edge sensitive interrupt is selected.

ESB. Edge Selection Bit. When this bit is set to one, the interrupt #2 (Port A & B lines) is positive edge sensitive, when cleared to zero the negative edge sensitive interrupt is selected.

GEN. Global Enable Interrupt. When this bit is set to one, all the interrupts are enabled. When this bit is cleared to zero all the interrupts (including the NMI) are disabled.

This register is cleared on reset.

Table 5. Interrupt Option Register Description

GEN	SET	Enable all interrupts
	CLEARED	Disable all interrupts
ESB	SET	Rising edge mode on interrupt input #2
	CLEARED	Falling edge mode on interrupt input #2
LES	SET	Level-sensitive mode on interrupt input #1
	CLEARED	Falling edge mode on interrupt input #1
OTHERS	NOT USED	

External Interrupts Operating Modes

The NMI interrupt is associated to the NMI pin of the ST6242. The two interrupt requests are "ORed". The highest priority interrupt request will be generated by a falling edge applied to the NMI pin. The NMI interrupt pin signal is latched and is automatically reset by the core at the beginning of the non-maskable interrupt service routine. An on-chip pull-up resistor and a schmitt trigger is available with the NMI pin.

The two interrupt sources associated with the falling/rising edge mode of the external interrupt pins (SPI vector #1, Ports A and B vector #2,) are connected to two internal latches. Each latch is set when a falling/rising edge occurs and is cleared when the associated interrupt routine is started. So, the occurrence of an external interrupt request is stored: a second interrupt, that occurs during the processing of the first one, will be processed as soon as the first one has been finished (if there is not an higher priority interrupt request). If more than one interrupt occurs during the processing of the first one, these other interrupt requests will be lost.

The storage of the interrupt requests is not available in the level sensitive detection mode. To be taken into account, the low level must be present on the interrupt pin when the core samples the line after the execution of the instructions.

During the end of each instruction the core tests the interrupt lines and if there is an interrupt request the next instruction is not executed and the related interrupt routine is executed.

INTERRUPTS (Continued)

Interrupt Procedure. The interrupt procedure is very similar to a call procedure, indeed the user can consider the interrupt as an asynchronous call procedure. As this is an asynchronous event the user does not know about the context and the time at which it occurred. As a result the user should save all the data space registers which will be used inside the interrupt routines. There are separate sets of processor flags for normal, interrupt and non-maskable interrupt modes which are automatically switched and so these do not need to be saved.

The following list summarizes the interrupt procedure:

ST62xx actions

- Interrupt detection
- The flags C and Z of the main routine are exchanged with the flags C and Z of the interrupt routine (or the NMI flags)
- The value of the PC is stored in the first level of the stack
- The normal interrupt lines are inhibited (NMI still active)
- First internal latch is cleared
- The related interrupt vector is loaded in the PC.

User actions

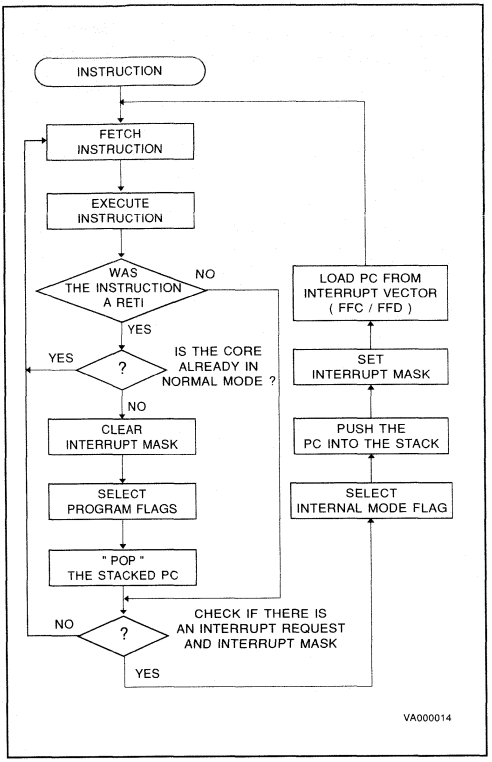
- User selected registers are saved inside the interrupt service routine (normally on a software stack)
- The source of the interrupt is found by polling (if more than one source is associated to the same vector) the interrupt flag of the source.
- Interrupt servicing
- Return from interrupt (RETI)

ST62xx actions

- Automatically the ST62xx core switches back to the normal flags (or the interrupt flags) and pops the previous PC value from the stack

The interrupt routine begins usually by the identification of the device that has generated the interrupt request (by polling). The user should save the registers which are used inside the interrupt routine (that holds relevant data) into a software stack. After the RETI instruction execution, the core carries out the previous actions and the main routine can continue.

Figure 14. Interrupt Processing Flow-Chart



WARNING. *GEN is the global enable for all interrupts except NMI. If this bit is cleared, the NMI interrupt is accepted when the ST62xx core is in the normal RUN Mode.*
If the ST62xx core is in STOP or WAIT Mode, the NMI is not accepted as a restart is disabled. This state can only be finished by a reset (fromthe Watchdog or an external Reset Signal).
As a consequence the NMI can be masked in STOP and WAIT modes, but not in RUN mode.

INTERRUPTS (Continued)

Interrupt request and mask bits

Interrupt Option Register, IOR Location C8h

- GEN. If this bit is set, all the ST62xx interrupts are enabled, if reset all interrupts are disabled (including the NMI).
- ESB. If this bit is set, all the input lines associated to interrupt vector #2 are rising edge sensitive, if reset they are falling edge sensitive.
- LES. If this bit is set, all the inputs lines associated to interrupt vector #1 are low level sensitive, if reset they are falling edge sensitive.

All other bits in this register are not used.

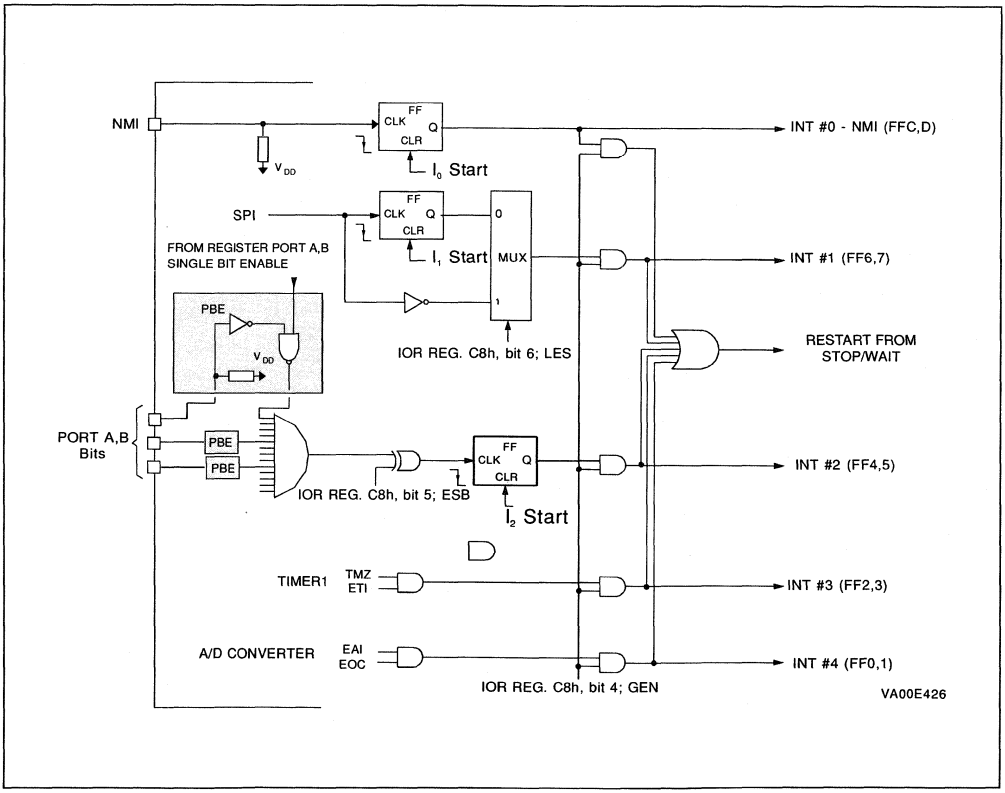
Timer Peripherals, TSCR1 and TSCR2 registers, locations D4h and D7h

- TMZ. A low-to-high transition indicates that the timer count register has decremented to zero. This means that an interrupt request can be generated in relation to the state of ETI bit.
- ETI. This bit, when set, enables the timer interrupt request.

A/D Converter Peripheral, ADCR register location D0h

- EOC. This read only bit indicates when a conversion has been completed, by going to one. An interrupt request can be generated in relation to the state of EAI bit.
- EAI. This bit, when set, enables the A/D converter interrupt request.

Figure 15. ST6242 Interrupt Circuit Diagram



RESET

The ST6242 can be reset in three ways: by the external reset input ($\overline{RESET}$) tied low, by power-on reset and by the digital watchdog/timer peripheral.

RESET Input

The $\overline{RESET}$ pin can be connected to a device of the application board in order to restart the MCU during its operation. The activation of the $\overline{RESET}$ pin may occur in the RUN, WAIT or STOP mode. This input has to be used to reset the MCU internal state and provide a correct start-up procedure. The pin is active low. The internal reset signal is generated by adding a delay to the external signal. Therefore even short pulses at the $\overline{RESET}$ pin will be accepted. This feature is valid providing that V_{DD} has finished its rising phase and the oscillator is correctly running (normal RUN or WAIT modes).

If $\overline{RESET}$ activation occurs in the RUN or Wait mode, the MCU is configured in the Reset mode for as long as the signal of the $\overline{RESET}$ pin is low. The processing of the program is stopped (in RUN mode only) and the Input/Outputs are in the High-impedance state with pull-up resistors switched on. As soon as the level on the $\overline{RESET}$ pin becomes high, the initialization sequence is executed.

If a $\overline{RESET}$ pin activation occurs in the STOP mode, the oscillator starts and all the inputs/outputs are configured in the High-impedance state with pull-up resistors switched on for as long as the level on the $\overline{RESET}$ pin remains low. When the level of the $\overline{RESET}$ pin becomes high, a delay is generated by the ST62xx core to wait that the oscillator becomes completely stabilized. Then, the initialization sequence is started.

Power-On Reset (POR)

The function of the POR consists in waking up the MCU during the power-on sequence. At the beginning of this sequence, the MCU is configured in the Reset state: every Input/Output port is configured in the input mode (High-impedance state with pull-up) and no instruction is executed. When the power supply voltage becomes sufficient, the oscillator starts to operate, nevertheless the ST62xx core generates a delay to allow the oscillator to be completely stabilized before the execution of the first instruction. The initialization sequence is then executed.

Internal circuitry generates a Reset pulse when V_{DD} is switched on. In the case of fast rising V_{DD} (transition time $\leq 100\mu s$), this reset pulse starts the internal reset procedure without the need of external components at the $\overline{RESET}$ pin. In cases of slowly or non monotonously rising V_{DD}, an external reset signal must be provided for a proper reset of the MCU.

For as long as the reset pin is kept at the low level, the processor remains in the reset state. The reset will be released after the voltage at the reset pin reaches the high level.

Note:

To have a correct ST62xx start-up, the user should take care that the reset input does not change to the high level before the V_{DD} level is sufficient to allow MCU operation at the chosen frequency (see recommended operating conditions).

An on-chip counter circuit provides a delay of 2048 oscillator cycles between the detection of the reset high level and the release of the MCU reset.

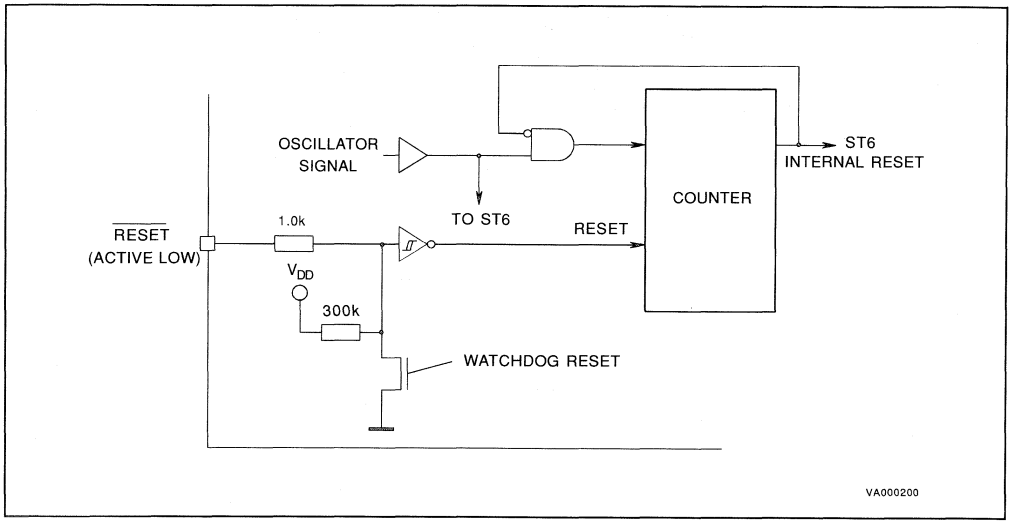
A proper reset signal for slow rising V_{DD}, i.e. the required delay between reaching sufficient operating voltage and the reset input changing to a high level, can be generally provided by an external capacitor connected between the $\overline{RESET}$ pin and V_{SS}.

RESET (Continued)

Watchdog Reset

The ST6242 provides an on-chip watchdog function in order to provide a graceful recovery from a software upset. If the watchdog register is not refreshed, preventing the end-of-count being reached, an internal circuit pulls down the $\overline{\text{RESET}}$ pin. The MCU will enter the reset state as soon as the voltage at $\overline{\text{RESET}}$ pin reaches the related low level. This also resets the watchdog which subsequently turns off the pull-down and activates the pull-up device at the $\overline{\text{RESET}}$ pin. This causes the positive transition at the $\overline{\text{RESET}}$ pin and terminates the reset state.

Figure 16. Reset Circuit

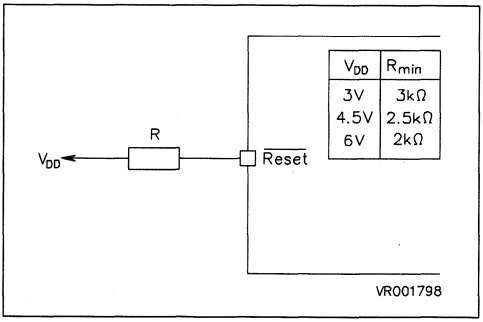


Application Notes

An external resistor between V_{DD} and reset pin is not required because an internal pull-up device is provided. If the user prefers, for any reason, to add an external pull-up resistor its value must comply with the Rmin value defined in Figure 17. If the value is lower than Rmin, the on-chip watchdog pull-down transistor might not be able to pull-down the reset pin resulting in an external deactivation of the watchdog function.

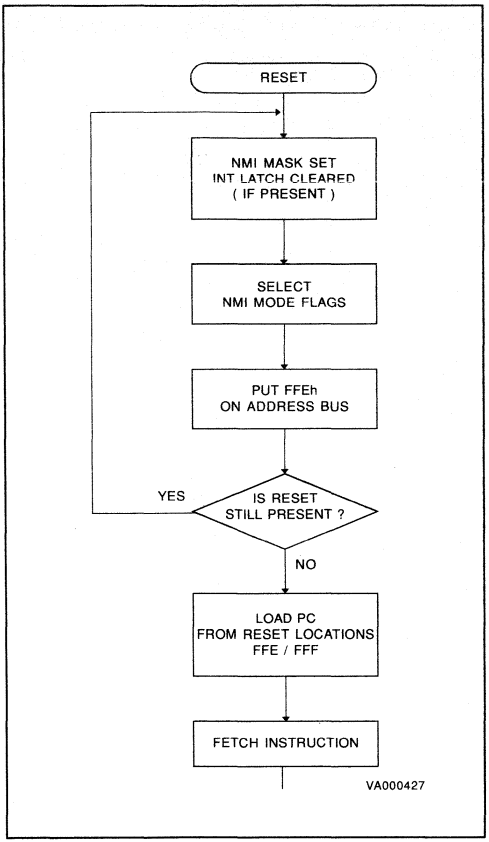
The POR function operates in a dynamic manner in the way that it brings about the initialization of the MCU when it detects a dynamic rising edge of the V_{DD} voltage. The typical detected threshold is about 2 volts, but the actual value of the detected threshold depends on the way in which the V_{DD} voltage rises up. The POR device DOES NOT allow the supervision of a static rising or falling edge of the V_{DD} voltage.

Figure 17. External Reset Resistance



RESET (Continued)

Figure 18. Reset & Interrupt Processing Flow-Chart



MCU Initialization Sequence

When a reset occurs the stack is reset to the program counter, the PC is loaded with the address of the reset vector (located in the program ROM at addresses FFEh & FFFh). A jump instruction to the beginning of the program has to be written into these locations. After a reset the interrupt mask is automatically activated so that the core is in non-maskable interrupt mode to prevent false or ghost interrupts during the restart phase. Therefore the restart routine should be terminated by a RETI instruction to switch to normal mode and enable interrupts. If no pending interrupt is present at the end of the reset routine the ST62xx will continue with the instruction after the RETI; otherwise the pending interrupt will be serviced.

Figure 19. Restart Initialization Program Flow-Chart

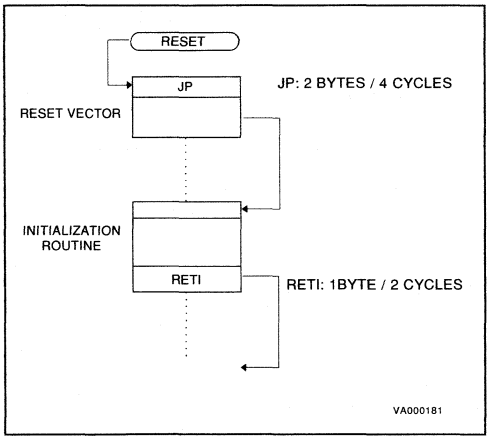


Table 6. Reset Configuration

Input/Output pins	Registers
Input Mode with pull-up and no interrupt	All cleared but A,X,Y,V,W, data RAM, LCD RAM, DWR (C9), PRPR (CA), DRBR (CB). Timers prescaler and TCR are initialized respectively at 7F and FF. Watchdog register DWDR (D8) is set to FEh.

WAIT & STOP MODES

The WAIT and STOP modes have been implemented in the ST62xx core in order to reduce the consumption of the product when the latter has no instruction to execute. These two modes are described in the following paragraphs

WAIT Mode

The configuration of the MCU in the WAIT mode occurs as soon as the WAIT instruction is executed. The microcontroller can also be considered as being in a "software frozen" state where the core stops processing the instructions of the routine, the contents of the RAM locations and peripheral registers are saved as long as the power supply voltage is higher than the RAM retention voltage but where the peripherals are still working.

The WAIT mode is used when the user wants to reduce the consumption of the MCU when it is in idle, while not losing count of time or monitoring of external events. The oscillator is not stopped in order to provide a clock signal to the peripherals. The timer counting may be enabled (writing the PSI bit in TSCR register) and the timer interrupt may be also enabled before entering the WAIT mode; this allows the WAIT mode to be left when timer interrupt occurs. The above explanation related to the timers applies also to the A/D converter.

If the exit from the WAIT mode is performed with a general RESET (either from the activation of the external pin or by watchdog reset) the MCU will enter a normal reset procedure as described in the RESET chapter. If an interrupt is generated during WAIT mode the MCU behavior depends on the state of the ST62xx core before the initialization of the WAIT sequence, but also of the kind of the interrupt request that is generated. This case will be described in the following paragraphs. In any case, the ST62xx core does not generate any delay after the occurrence of the interrupt because the oscillator clock is still available.

STOP Mode

If the Watchdog is disabled the STOP mode is available. When in STOP mode the MCU is placed in the lowest power consumption mode. In this operating mode the microcontroller can be considered as being "frozen", no instruction is executed, the oscillator is stopped, the contents of the RAM locations and peripheral registers are saved as long as the power supply voltage is higher than the RAM retention voltage, and the ST62xx core waits for the occurrence of an external interrupt request or Reset activation to output from the STOP state.

If the exit from the STOP mode is performed with a general RESET (by the activation of the external pin) the MCU will enter a normal reset procedure as described in the RESET chapter. The case of an interrupt depends on the state of the ST62xx core before the initialization of the STOP sequence and also of the kind of the interrupt request that is generated.

This case will be described in the following paragraphs. In any case, the ST62xx core generates a delay after the occurrence of the interrupt request in order to wait the complete stabilization of the oscillator before the execution of the first instruction.

Exit from WAIT and STOP Modes

The following paragraphs describe the output procedure of the ST62xx core from WAIT and STOP modes when an interrupt occurs (not a RESET). It must be noted that the restart sequence depends on the original state of the MCU (normal, interrupt or non-maskable interrupt mode) before the start of the WAIT or STOP sequence, but also of the type of the interrupt request that is generated.

Normal Mode. If the ST62xx core was in the main routine when the WAIT or STOP instruction has been executed, the ST62xx core outputs from the stop or wait mode as soon as any interrupt occurs; the related interrupt routine is executed and at the end of the interrupt service routine the instruction that follows the STOP or the WAIT instruction is executed if no other interrupts are pending.

WAIT & STOP MODES (Continued)

Not Maskable Interrupt Mode. If the STOP or WAIT instruction has been executed during the execution of the non-maskable interrupt routine, the ST62xx core outputs from the stop or wait mode as soon as any interrupt occurs: the instruction that follows the STOP or the WAIT instruction is executed and the ST62xx core is still in the non-maskable interrupt mode even if another interrupt has been generated.

Normal Interrupt Mode. If the ST62xx core was in the interrupt mode before the initialization of the STOP or WAIT sequence, it outputs from the stop or wait mode as soon as any interrupt occurs. Nevertheless, two cases have to be considered:

- If the interrupt is a normal interrupt, the interrupt routine in which the WAIT or STOP was entered will be completed with the execution of the instruction that follows the STOP or the WAIT and the ST62xx core is still in the interrupt mode. At the end of this routine pending interrupts will be serviced in accordance to their priority.
- If the interrupt is a non-maskable interrupt, the non-maskable routine is processed at first. Then the routine in which the WAIT or STOP was entered will be completed with the execution of the instruction that follows the STOP or the WAIT and the ST62xx core remains in the normal interrupt mode.

Notes:

To reach the lowest power consumption the user software must take care of:

- placing the A/D converter in its power down mode by clearing the PDS bit in the A/D control register before entering the STOP instruction.
- switching off the 32kHz oscillator by clearing the oscillator start/stop bit in the 32kHz oscillator control register.
- putting the EEPROM on-chip memory in stand-by mode by writing 40h in EEPROM Control Register (address DFh).

The LCD Driver peripheral is automatically switched-off by the STOP instruction when the 32kHz oscillator operation is not selected.

When the watchdog has been enabled, the STOP instruction is deactivated and any attempt to execute the STOP instruction will cause an execution of a WAIT instruction.

If all the interrupt sources are disabled (including NMI if GEN="0"), the restart of the MCU can only be done by a RESET activation. The WAIT and STOP instructions are not executed if an enabled interrupt request is pending.

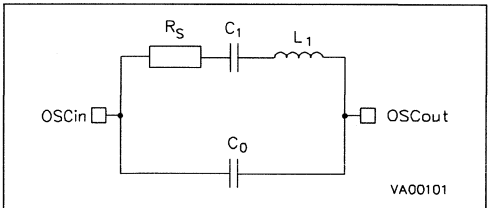
ON-CHIP CLOCK OSCILLATOR

The internal oscillator circuit is designed to require a minimum of external components. A crystal, a ceramic resonator, or an external signal (provided to the OSCin pin) may be used to generate a system clock with various stability/cost tradeoffs. The different clock generator options connection methods are shown in Figure 21.

One machine cycle takes 13 oscillator pulses; 12 clock pulses are needed to increment the PC while and additional 13th pulse is needed to stabilize the internal latches during memory addressing. This means that with a clock frequency of 8MHz the machine cycle is 1.625µs.

The crystal oscillator start-up time is a function of many variables: crystal parameters (especially R_S), oscillator load capacitance (CL), IC parameters, ambient temperature, supply voltage. It must be observed that the crystal or ceramic leads and circuit connections must be as short as possible.Typical values for CL1, CL2 are 15-22pF for a 4/8MHz crystal. The oscillator output frequency is internally divided by 13 to produce the machine cycle and by 12 to produce the Timer, the Watchdog and the A/D peripheral clock. A machine cycle is the smallest unit needed to execute any operation (i.e., increment the program counter). An instruction may need two, four, or five machine cycles to be executed.

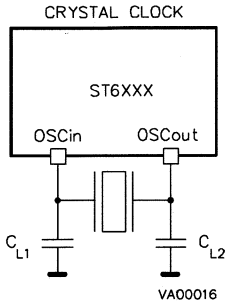
Figure 20. Crystal Parameters

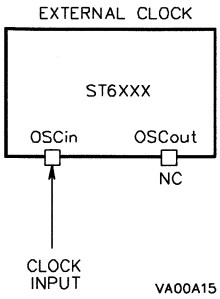


Crystal parameters: AT-Cut Parallel Resonance Crystal
C0 = Parallel Resonance Capacitance

ON-CHIP CLOCK OSCILLATOR (Continued)

Figure 21. Oscillator Connection





CL1 = CL2 = 12 to 22pF for a 4/8MHz crystal (typical)

INPUT/OUTPUT PORTS

The ST6242 microcontroller has 10 Input/Output lines that can be individually programmed either in the input mode or the output mode with the following options that can be selected by software:

- Input without pull-up and without interrupt
- Input with pull-up and with interrupt
- Input with pull-up without interrupt
- Analog inputs (PA4-PA7, PB2-PB3)
- SPI control signals (PB5-PB7)
- Push-pull output
- Standard Open drain output
- 20mA Open drain output (PB4-PB7)

The lines are organized in two ports (port A,B).

Each port occupies 3 registers in the data space. Each bit of these registers is associated with a particular line (for instance, the bits 0 of the Port A Data, Direction and Option registers are associated with the PA0 line of Port A).

The two DATA registers (DRA, DRB), are used to read the voltage level values of the lines programmed in the input mode, or to write the logic value of the signal to be output on the lines configured in the output mode. The port data registers can be read to get the effective logic levels of the pins, but they can be also written by the user software, in conjunction with the related option registers, to select the different input mode options.

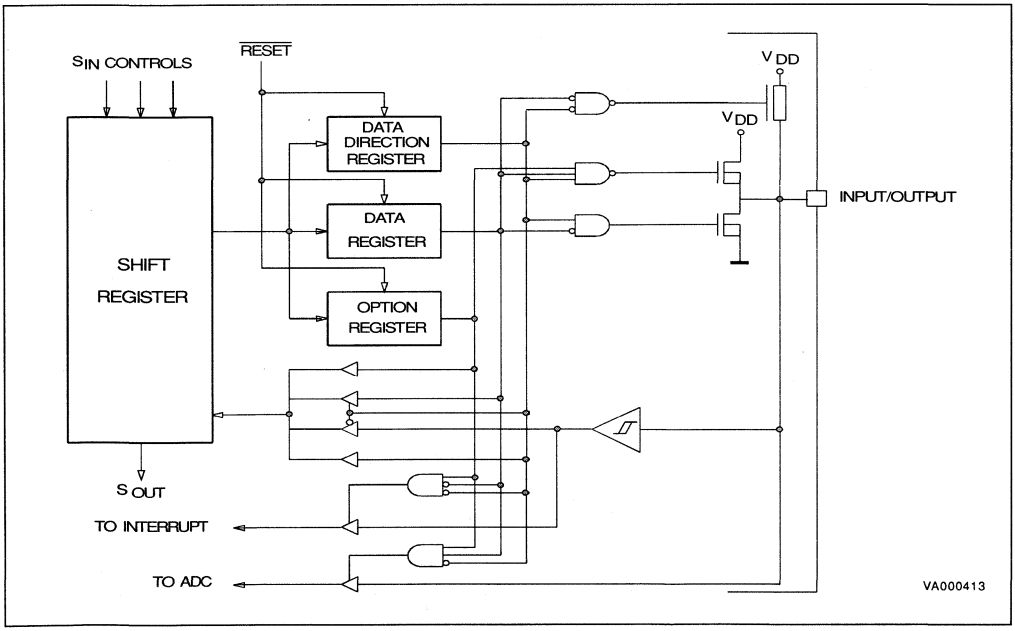
Single-bit operations on I/O registers are possible but care is necessary because reading in input mode is done from I/O pins while writing will directly affect the Port data register causing an undesired change of the input configuration.

The two Data Direction registers (DDRA, DDRB) allow the selection of the data direction of each pin (input or output).

The two Option registers (ORPA, ORPB) are used to select the different port options available both in input and in output mode.

All the I/O registers can be read or written as any other RAM location of the data space, so no extra RAM cell is needed for port data storing and manipulation. During the initialization of the MCU, all the I/O registers are cleared and the input mode with pull-up/no-interrupt is selected on all the pins, thus avoiding pin conflicts.

Figure 22. I/O Port Block Diagram



INPUT/OUTPUT PORTS (Continued)

I/O Pin Programming

Each pin can be individually programmed as input or output with different input and output configurations.

This is achieved by writing to the relevant bit in the data (DR), data direction register (DDR) and option registers (OR). Table 7 shows all the port configurations that can be selected by user software.

Input Option Description

Pull-up, High Impedance Option. All the input lines can be individually programmed with or without an internal pull-up according to the codes programmed in the OR and DR registers . If the pull-up option is not selected, the input pin is in the high-impedance state.

Interrupt Option. All the input lines can be individually connected by software to the interrupt lines of the ST62xx core according to the codes programmed in the OR and DR registers. The pins of Port A and B are "ORed" and are connected to the interrupt associated to the vector #2. The interrupt modes (falling edge sensitive, rising edge sensitive) can be selected by software for each port by programming the IOR register.

Analog Input Option. The six PA4-PA7, PB2-PB3 pins can be configured to be analog inputs according to the codes programmed in the OR and DR registers. These analog inputs are connected to the on-chip 8-bit Analog to Digital Converter. *ONLY ONE* pin should be programmed as analog input at a time, otherwise the selected inputs will be shorted.

Figure 23. I/O Port Data Registers

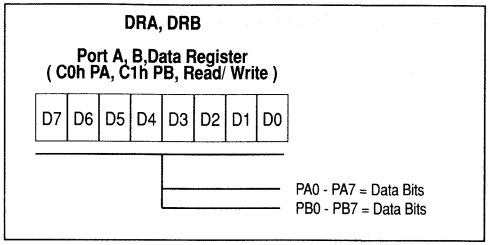


Figure 24. I/O Port Data Direction Registers

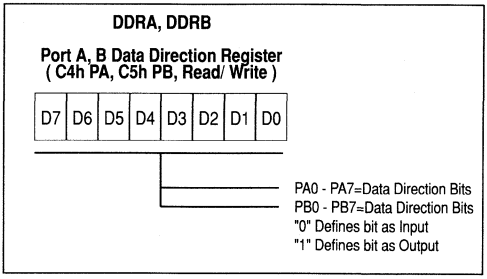
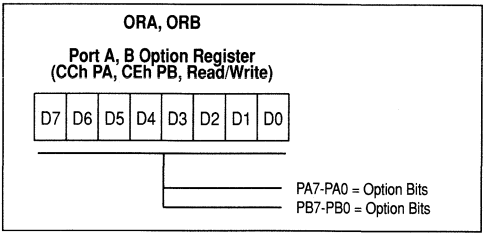


Figure 25. I/O Port Option Registers



Note: For complete coding explanation refer to Table 8.

Table 8. I/O Port Options Selection

DDR	OR	DR	MODE	OPTION
0	0	0	Input	With pull-up, no interrupt (Reset state)
0	0	1	Input	No pull-up, no interrupt
0	1	0	Input	With pull-up, with interrupt
0	1	1	Input	No pull-up, no interrupt (for PB4-PB7).
			Input	Analog input (for PA4-PA7, PB2-PB3)
1	0	X	Output	Open-drain output (20mA sink current for PB4-PB7)
1	1	X	Output	Push-pull output (20mA sink current for PB4-PB7)

Note: X. Means don't care.

INPUT/OUTPUT PORTS (Continued)

SPI alternate function Option. The I/O pins PB5-PB7 are also used by serial peripheral interface SPI. PB5 is connected with the SPI clock input SCL, PB6 is connected with the SPI data input SIN and PB7 is connected with the SPI data output SOUT.

For serial input operation PB5 and PB6 have to be programmed as inputs. For serial output operation PB7 has to be programmed as open-drain output (DDR = "1", OPR = "0"). In this operating mode the output of the SPI shift register instead of the port data register is connected to the port buffer. When PB7 is programmed as push-pull output (DDR = "1", OPR = "1"), the port data register is connected to the port buffer. When the SPI peripheral is not used PB5-PB7 can be used as general purpose I/O lines (provided that PB7 is not selected to be open-drain in output mode).

Notes:

Switching the I/O ports from one state to another should be done in a way that no unwanted side effects can happen. The recommended safe transitions are shown below. All other transitions are risky and should be avoided during change of operation mode as it is most likely that there will be an unwanted side-effect such as interrupt generation or two pins shorted together by the analog input lines.

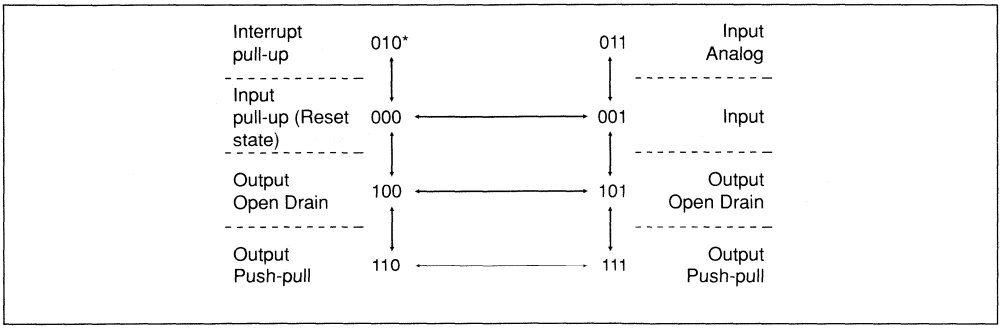
Single bit SET and RES instructions should be used very carefully with Port A and B data registers because these instructions make an implicit read and write back of the whole addressed register byte. In port input mode however data register address reads from input pins, not from data register latches and data register information in input mode is used to set characteristics of the input pin (interrupt, pull-up, analog input), therefore these characteristics may be unintentionally reprogrammed depending on the state of input pins. As general rule is better to use SET and RES instructions on data register only when the whole port is in output mode. If input or mixed configuration is needed it is recommended to keep a copy of the data register in RAM. On this copy it is possible to use single bit instructions, then the copy register could be written into the port data register.

```
SET     bit, datacopy
LD      a, datacopy
LD      DRA, a
```

The WAIT and STOP instructions allow the ST62xx to be used in situations where low power consumption is needed. The lowest power consumption is achieved by configuring I/Os in input mode with well-defined logic levels.

The user has to take care not to switch outputs with heavy loads during the conversion of one of the analog inputs in order to avoid any disturbance in the measurement.

Figure 26. I/O Port State Transition Diagram for Safe



Note *. xxx = DDR, OR, DR Bits respectively

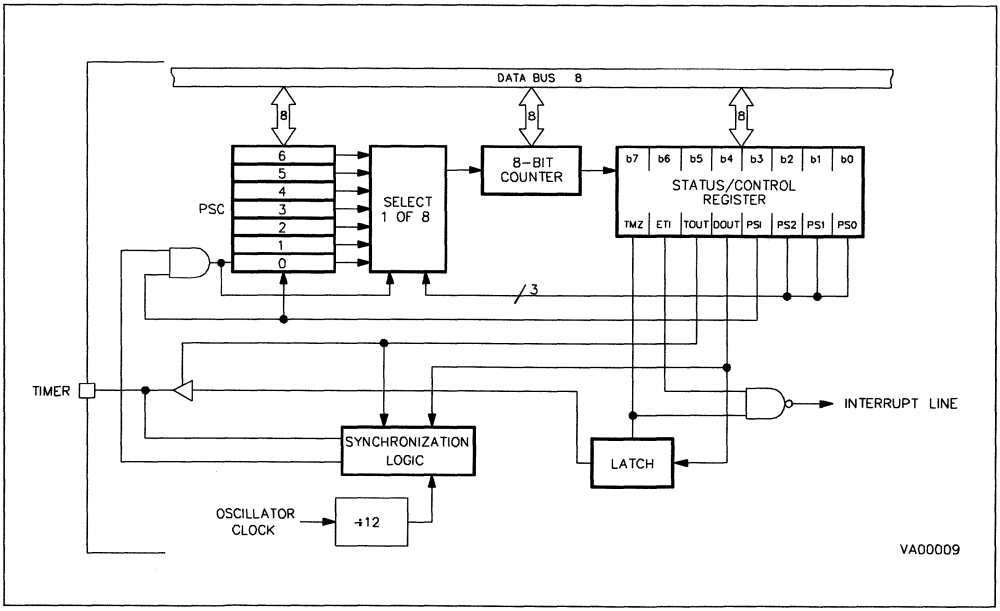
TIMERS

The ST6242 offers an on-chip Timer peripheral consisting of an 8-bit counter with a 7-bit programmable prescaler. This Timer gives a maximum count of 2^{15}, and contains a control logic that allows the configuration of the peripheral in three operating modes. Figure 27 shows the Timer block diagram. The content of the 8-bit counter can be read/written in the Timer/Counter register TCR which is addressed in the data space as a RAM location at addresses D3h. The state of the 7-bit prescaler can be read in the PSC register at address D2h. The control logic device is managed in the TSCR register (address D4h) as described in the following paragraphs.

The 8-bit counter is decrement by the output (rising edge) coming from the 7-bit prescaler and can be loaded and read under program control. When it decrements to zero then the TMZ (Timer Zero) bit in the TSCR is set to one. If the ETI (Enable Timer Interrupt) bit in the TSCR is also set to one an interrupt request, associated to interrupt vector #3, is generated. The Timer interrupt can be used to exit the MCU from the WAIT mode.

The prescaler input is the oscillator frequency divided by 12. The prescaler input decrements on the rising edge. Depending on the division factor programmed by PS2, PS1 and PS0 bits in the TSCR, the clock input of the timer/counter register is multiplexed to different sources. On division factor 1, the clock input of the prescaler is also that of timer/counter; on factor 2, bit 0 of prescaler register is connected to the clock input of TCR. This bit changes its state with the half frequency of prescaler clock input. On factor 4, bit 1 of PSC is connected to clock input of TCR, and so on. On division factor 128, the MSB bit 6 of PSC is connected to the clock input of TCR. The prescaler initialize bit (PSI) in the TSCR register must be set to one to allow the prescaler (and hence the counter) to start. If it is cleared to zero then all of the prescaler bits are set to one and the counter is inhibited from counting. The prescaler can be given any value between 0 and 7Fh by writing to address D2h, if bit PSI in the TSCR register is set to one. The tap of the prescaler is selected using the PS2,PS1,PS0 bits in the control register. Figure 27 shows the Timer working principle.

Figure 27. Timer Peripheral Block Diagram



TIMERS (Continued)

Timer Operating Modes

The Timer has one operating mode. This mode is selected with TOUT= "1" in the Timer status control register TSCR (D4h).

The prescaler is decremented by the timer clock (OSC/12). The user can select the desired prescaler division ratio through the PS2, PS1, PS0 bits.

When the TCR count reach 0, it sets the TMZ bit in the TSCR.

Timer Interrupt

When the counter register decrements to zero and the software controlled ETI (Enable Timer Interrupt) bit is set to one then an interrupt request associated to interrupt vector #3 is generated. When the counter decrements to zero also the TMZ bit in the TSCR register is set to one.

Notes:

TMZ is set when the counter reaches 00h; however, it may be set by writing 00h in the TCR register or setting bit 7 of the TSCR register. TMZ bit must be cleared by user software when servicing the timer interrupt to avoid undesired interrupts when leaving the interrupt service routine. After reset, the 8-bit counter register is loaded to FFh while the 7-bit prescaler is loaded to 7Fh , and the TSCR register is cleared which means that timer is stopped (PSI="0") and the timer interrupt is disabled.

If the Timer is programmed in output mode, DOUT bit is transferred to the TIMER pin when TMZ is set to one (by software or due to counter decrement). When TMZ is high, the latch is transparent and DOUT is copied to the timer pin. When TMZ goes low, DOUT is latched.

A write to the TCR register will predominate over the 8-bit counter decrement to 00h function, i.e. if a write and a TCR register decrement to 00h occur simultaneously, the write will take precedence, and the TMZ bit is not set until the 8-bit counter reaches 00h again. The values of the TCR and the PSC registers can be read accurately at any time.

Figure 29. Timer Status Control Register

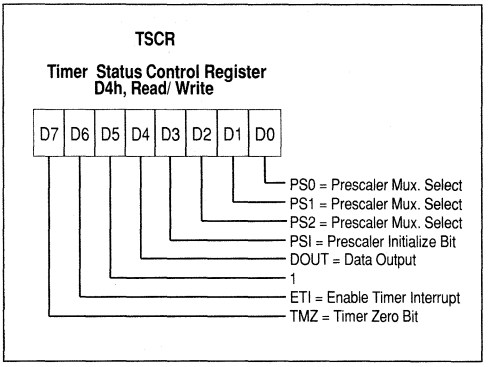
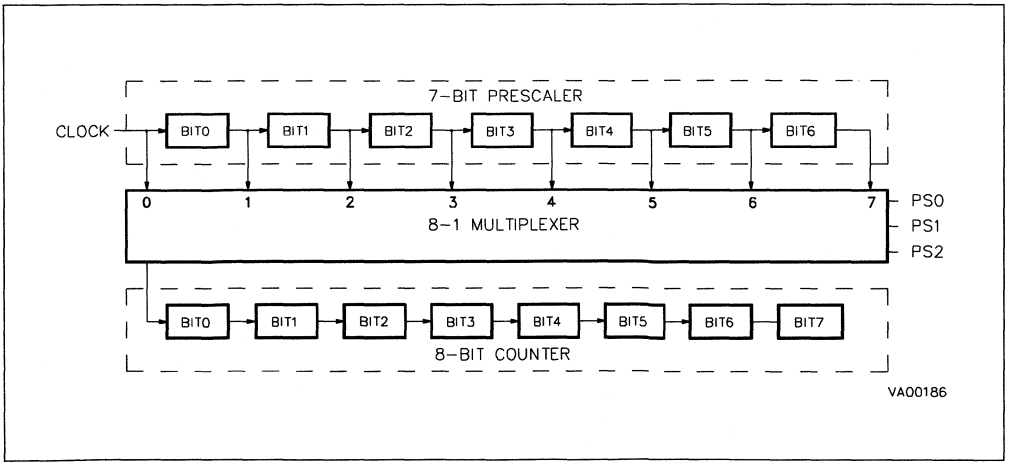


Figure 28. Timer Working Principle



TIMERS (Continued)

TMZ. Low-to-high transition indicates that the timer count register has decremented to zero. This bit must be cleared by user software before starting with a new count.

ETI. This bit, when set, enables the timer interrupt request (vector #3). If ETI="0" the timer interrupt is disabled. If ETI="1" and TMZ="1" an interrupt request is generated.

TOUT. This bit must be set high.

DOUT. Data sent to the timer output when TMZ is set high.

PSI. Used to initialize the prescaler and inhibit its counting. When PSI="0" the prescaler is set to 7Fh and the counter is inhibited. When PSI="1" the prescaler is enabled to count downwards. As long as PSI="0" both counter and prescaler are not running.

PS2, PS1, PS0. These bits select the division ratio of the prescaler register (see Table 9).

Table 9. Prescaler Division Factors

PS2	PS1	PS0	Divided by	PS2	PS1	PS0	Divided by
0	0	0	1	1	0	0	16
0	0	1	2	1	0	1	32
0	1	0	4	1	1	0	64

Figure 30. Timer Counter Register

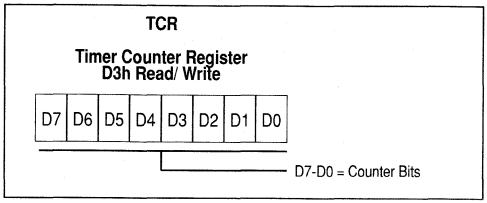
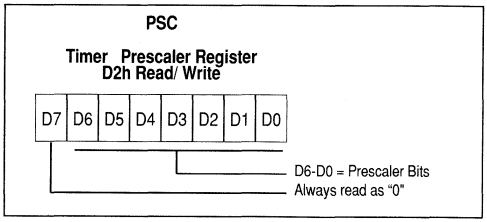


Figure 31. Prescaler Register



DIGITAL WATCHDOG

The ST6242 watchdog is a software activated watchdog.

Figure 32 shows the watchdog block diagram while Figure 34 shows its working principle.

The software activated digital watchdog consists of a down counter that can be used to provide a controlled recovery from a software upset or as a simple 7-bit timer for general purpose counting. The watchdog uses one data space register (DWDR location D8h). The watchdog register is set to FEh after reset and the watchdog function is disabled. The watchdog time can be programmed using the 6 Most Significant Bits in the Watchdog register. The check time can be set differently for different routines within the general program.

After a reset the software Watchdog is in the off-state. The watchdog should be activated inside the Reset restart routine by writing a "1" in watchdog timer register bit 0. Bit one of this register must be set to one before programming bit zero as otherwise a reset will be immediately generated when bit 0 is set. This allows the user to generate a reset by software (bit 0 = "1", bit 1 = "0"). Once bit 0 is set, it can not be cleared by software without generating a Reset. The delay time is defined by programming bits 2-7 of the watchdog register. Bit 7 is the Least Significant Bit while bit 2 is the MSB. This gives the possibility to generate a reset in a time between 3072 to 196608 clock cycles in 64 possible steps: (With a clock frequency of 8MHz this means from 384µs to 24.576ms). The reset is prevented if the register is reloaded with the desired value before bits 2-7 decrement from all zeros to all ones. If the watchdog is active the STOP instruction is deactivated and a WAIT instruction is automatically executed instead of a STOP. If bit 0 of the watchdog register is never set to one then bits 1-7 of the register can be used as a simple 7-bit counter which is decrement every 3072 clock cycles.

Figure 33. Watchdog Register

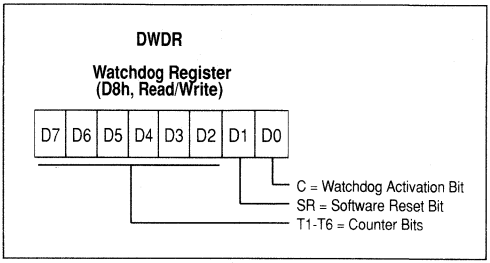
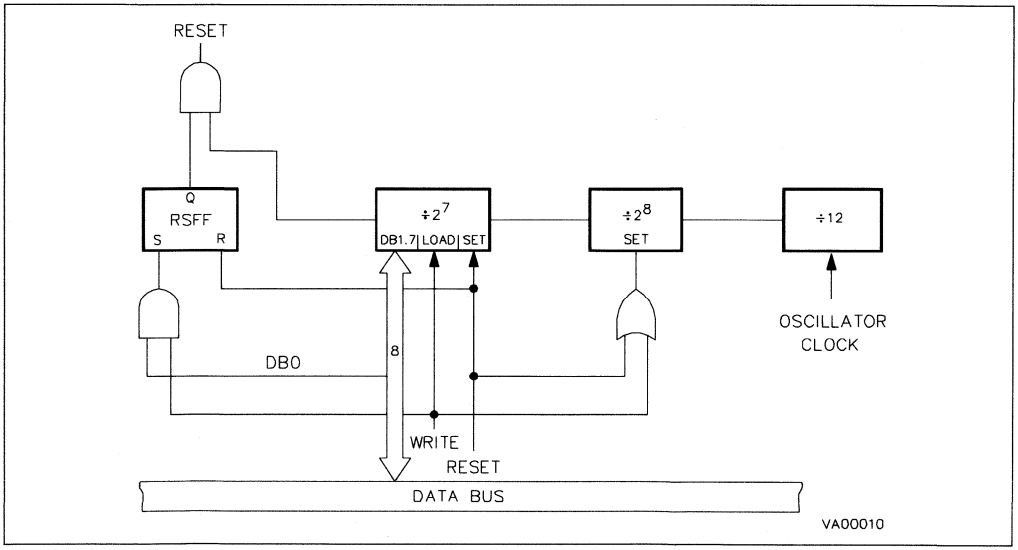


Figure 32. Digital Watchdog Block Diagram



DIGITAL WATCHDOG (Continued)

Watchdog Register

C. This is the watchdog activation bit, that, if set to one, will activate the watchdog function. When cleared to zero it allows the use of the counter as a 7-bit timer. This bit is cleared on reset.

SR. This bit is set to one during the reset and will generate a software reset if cleared to zero. When C = "0" (watchdog disabled software option) it is the MSB of the 7-bit timer.

T1-T6. These are the watchdog counter bits. It should be noted that D7 (T1) is the LSB of the counter and D2 (T6) is the MSB of the counter. These bits are in the opposite order to normal.

Application note:

If the Watchdog is not used during power-on reset external noise may cause the undesired activation of the Watchdog with a generation of an unexpected reset. To avoid this risk, two additional instructions, that check the state of the watchdog and eventually reset the chip are needed within the first 27 instructions, after the reset. These instructions are:

```
jrx 0, WD, #+3
ldi WD, 0FDH
```

These instructions should be executed at the very beginning of the customer program.

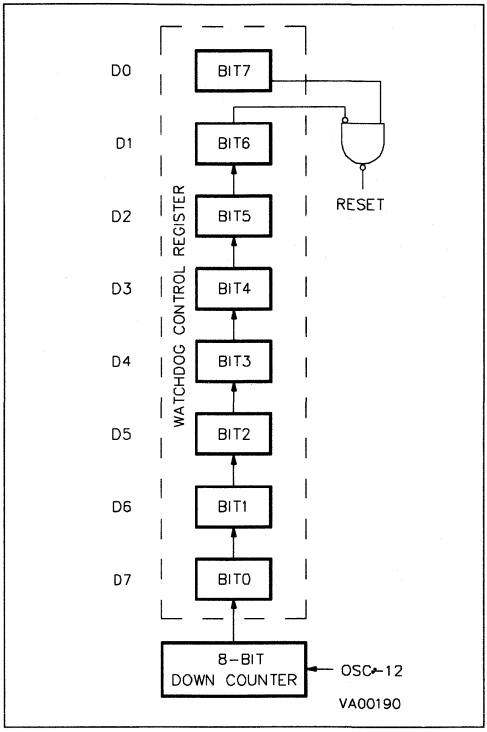
If the Watchdog is used, during power-on reset the Watchdog register may be set to a low value, that could give a reset after 28 instructions earliest. To avoid undesired resets, the Watchdog must be set to the desired value within the first 27 instructions, the best is to put at the very beginning.

Alternatively the normal legal state can be checked with the following short routine:

```
ldi a, 0FEH
and a, WD
cpi a, 0FEH
jrz #+3
ldi WD, 0FDH
```

This sequence is recommended for security applications, where possible stack confusion error loops must be avoided and the Watchdog must only be refreshed after extensive checks.

Figure 34. Watchdog Working Principle



8-BIT A/D C ONVERTER

The A/D converter of ST6242 is an 8-bit analog to digital converter with 6 analog inputs (as alternate functions of I/O lines PA4-PA7, PB2-PB3) offering 8-bit resolution with total accuracy ±2 LSB and a typical conversion time of 70µs (clock frequency of 8MHz).

The A/D peripheral converts the input voltage by a process of successive approximations using a clock frequency derived from the oscillator with a division factor of twelve. With an oscillator clock frequency less than 1.2MHz, the A/D converter accuracy is decreased.

The selection of the pin signal that has to be converted is done by configuring the related I/O line as analog input through the I/O ports option and data registers (refer to I/O ports description for additional information). Only one I/O line must be configured as analog input at a time. The user must avoid the situation in which more than one I/O pin is selected to be analog input to avoid malfunction of the ST62xx.

The ADC uses two registers in the data space: the ADC data conversion register which stores the conversion result and the ADC control register used to program the ADC functions.

A conversion is started by writing a "1" to the Start bit (STA) in the ADC control register. This automatically clears (resets to "0") the End Of Conversion Bit (EOC). When a conversion has been finished this EOC bit is automatically set to "1" in order to flag that conversion is complete and that the data in the ADC data conversion register is valid. Each conversion has to be separately initiated by writing to the STA bit.

The STA bit is continually being scanned so that if the user sets it to "1" while a previous conversion is in progress then a new conversion is started before the previous one has been completed.The start bit (STA) is a write only bit, any attempt to read it will show a logical "0".

The A/D converter has a maskable interrupt associated to the end of conversion. This interrupt is associated to the interrupt vector #4 and occurs when the EOC bit is set, i.e. when a conversion is completed. The interrupt is masked using the EAI (interrupt mask) bit in the control register.

The power consumption of the device can be reduced by turning off the ADC peripheral. That is achieved when the PDS bit in the ADC control register is cleared to "0". If PDS="1", the A/D is supplied and enabled for conversion. This bit must be set at least one instruction before the beginning of the conversion to allow the stabilization of the A/D converter.*This action is needed also before entering the STOP instruction as the A/D comparator is not automatically disabled by the STOP mode*

Figure 35. A/D Converter Block Diagram

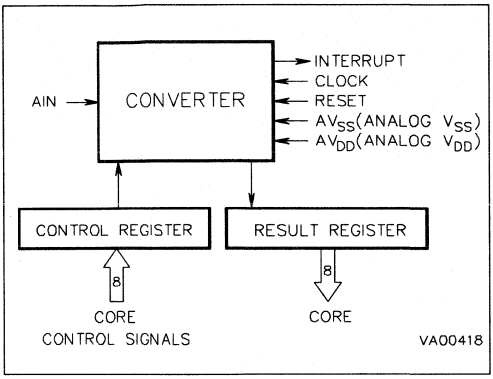


Figure 36. A/D Converter Control Register

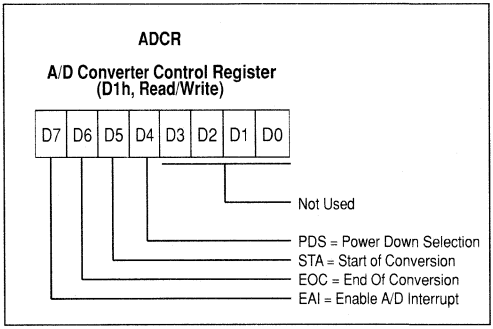
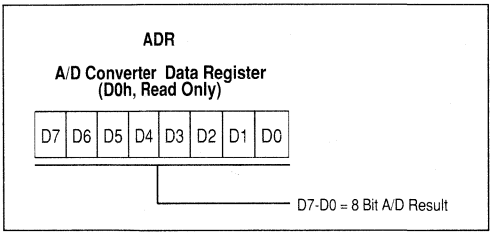


Figure 37. A/D Converter Data Register



8-BIT A/D CONVERTER(Continued)

During reset any conversion in progress is stopped, the control register is reset to all zeros and the A/D interrupt is masked (EAI=0).

A/D Converter Control Register

EAI. If this bit is set to one the A/D interrupt (vector #4) is enabled, when EAI=0 the interrupt is disabled.

EOC. *Read Only*;This read only bit indicates when a conversion has been completed. This bit is automatically reset to zero when the STA bit is written. If the user is using the interrupt option then this bit can be used as an interrupt pending bit. Data in the data conversion register are valid only when this bit is set to one.

STA. *Write Only*; Writing a "1" in this bit will start a conversion on the selected channel and automatically reset to zero the EOC bit. If the bit is set again when a conversion is in progress, the present conversion is stopped and a new one will take place. This bit is write only, any attempt to read it will show a logical zero.

PDS. This bit activates the A/D converter if set to 1. Writing a zero into this bit will put the ADC in power down mode (idle mode).

D3-D0. Not used

A/D Converter Data Register

D7-D0. *Read Only*;These are the conversion result bits; the register is read only and stores the result of the last conversion. The contents of this register are valid only when EOC bit in the ADCR register is set to one (end-of-conversion).

Notes:

The ST62xx A/D converter does not feature a sample and hold. The analog voltage to be measured should therefore be stable during the conversion time. Variation should not exceed ±1/2 LSB for the best accuracy in measurement.

Since the ADC is on the same chip as the microprocessor the user should not switch heavily loaded output signals during conversion if high precision is needed. This is because such switching will affect the supply voltages which are used for comparisons.

A low pass filter can be used at the analog input pins to reduce input voltage variation during the conversion. For true 8 bit conversions the impedance of the analog voltage sources should be less than 30kΩ while the impedance of the reference voltage should not exceed 2kΩ.

The accuracy of the conversion depends on the quality of the power supply voltages (V_{DD} and V_{SS}). The user must specially take care of applying regulated reference voltage on the V_{DD} and V_{SS} pins (the variation of the power supply voltage must be inferior to 5V/ms).

The converter can resolve the input voltage with an resolution of:

$$\frac{V_{DD} - V_{SS}}{256}$$

So if operating with a supply voltage of 5V the resolution is about 20mV.

The Input voltage (Ain) which has to be converted must be constant for 1µs before conversion and remain constant during the conversion.

The resolution of the conversion can be improved if the power supply voltage (V_{DD}) of the microcontroller becomes lower.

In order to optimize the resolution of the conversion, the user can configure the microcontroller in the WAIT mode because this mode allows the minimization of the noise disturbances and the variations of the power supply voltages due to the switching of the outputs. Nevertheless, it must be take care of executing the WAIT instruction as soon as possible after the beginning of the conversion because the execution of the WAIT instruction may provide a small variation of the V_{DD} voltage (the negative effect of this variation is minimized at the beginning of the conversion because the latter is less sensitive than the end of the conversion when the less significant bits are determined).

The best configuration from a accuracy point of view is the WAIT mode with the Timer and LCD driver stopped. Indeed, only the ADC peripheral and the oscillator are still working. The MCU has to be wake-up from the WAIT mode by the interrupt of the ADC peripheral at the end of the conversion. It must be noticed that the wake-up of the microcontroller could be done also with the interrupt of the TIMER, but in this case, the Timer is working and some noise could disturb the converter in terms of accuracy.

SERIAL PERIPHERAL INTERFACE (SPI)

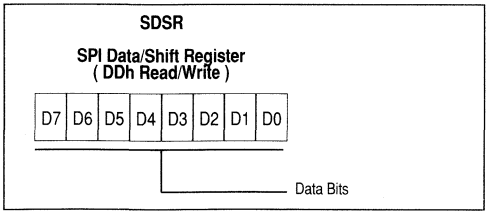
The ST6242 SPI is an optimized serial synchronous interface that supports a wide range of industry standard SPI specifications. The ST6242 SPI is controlled by small and simple user software to perform serial data exchange. The serial shift clock can be implemented either by software (using the bit-set and bit-reset instructions), with the on-chip Timer 1 by externally connecting the SPI clock pin to the timer pin or by directly applying an external clock to the SPI.

The peripheral is composed by an 8-bit Data/shift Register (address DDh) and a 4-bit binary counter. The SCL, Sin and Sout SPI data and clock signals are connected to the PB5, PB6 and PB7 I/O lines. With the 3 I/O pins, the SPI can operate in the following operating modes: Software SPI, S-BUS, I^2C-bus and as a standard serial I/O (clock, data, enable). An interrupt request can be generated after eight clock pulses. Figure 39 shows the SPI block diagram.

The PB5/SCL line clocks, on the falling edge, the shift register and the counter. To allow SPI operation the PB5/SCL must be programmed as input, an external clock supplied to this pin will drive the SPI peripheral (slave mode).

If PB5/SCL is programmed as output, a clock signal can be generated by software, setting and resetting the port line by software (master mode).

Figure 38. SPI Data/Shift Register



The SCL clock signal is the shift clock for the SPI data/shift register. The PB6/Sin pin is the serial shift input and PB7/Sout is the serial shift output. These two lines can be tied together to implement two wires protocols (I^2C-bus, etc). When data is serialized, the MSB is the first bit. PB6/Sin has to be programmed as input. For serial output operation PB7/Sout has to be programmed as open-drain output.

After 8 clock pulses (D7..D0) the output $\overline{Q4}$ of the 4-bit binary counter becomes low, disabling the clock from the counter and the data/shift register. Q4 enables the clock to generate an interrupt on the 8th clock falling edge as long as no reset of the counter (processor write into the 8-bit data/shift register) takes place. After a processor reset the interrupt is disabled. The interrupt is active when writing data in the shift register (DDh) and desactivated when writing any data in the register SPI Interrupt Disable (C2h).

The generation of an interrupt to the Core provides information that new data is available (input mode) or that transmission is completed (output mode), allowing the Core to generate an acknowledge on the 9th clock pulse (I^2C-bus).

Since the SPI interrupt is connected to interrupt #1, the falling edge interrupt option should be selected by clearing to zero bit 6 of the Interrupt Option Register (IOR, C8h).

After power on reset, or after writing the data/shift register, the counter is reset to zero and the clock is enabled. In this condition the data shift register is ready for reception. No start condition has to be detected. Through the user software the Core may pull down the Sin line (Acknowledge) and slow down the SCL, as long as it is needed to carry out data from the shift register.

SERIAL PERIPHERAL INTERFACE (Continued)

I²C-bus Master-Slave, Receiver-Transmitter

When pins Sin and Sout are externally connected together it is possible to use the SPI as a receiver as well as a transmitter. With a simple software routine (by using bit-set and bit-reset on I/O line) a clock can be generated allowing I²C-bus to work in master mode.

When implementing an I²C-bus protocol, the start condition can be detected by setting the processor into a "wait for start" condition by simply enabling the interrupt of the PA6/Sin I/O port. This frees the processor from polling the Sin and SCL lines. After the transmission/reception the processor has to poll for the STOP condition.

In slave mode the user software can slow down the SCL clock frequency by simply putting the SCL I/O line in output open-drain mode and writing a zero into the corresponding data register bit.

As it is possible to directly read the Sin pin directly through the port register, the software can detect a difference between internal data and external data (master mode). Similar condition can be applied to the clock.

The typical speed of transmission in I²C master or slave mode is in the range of 10kHz.

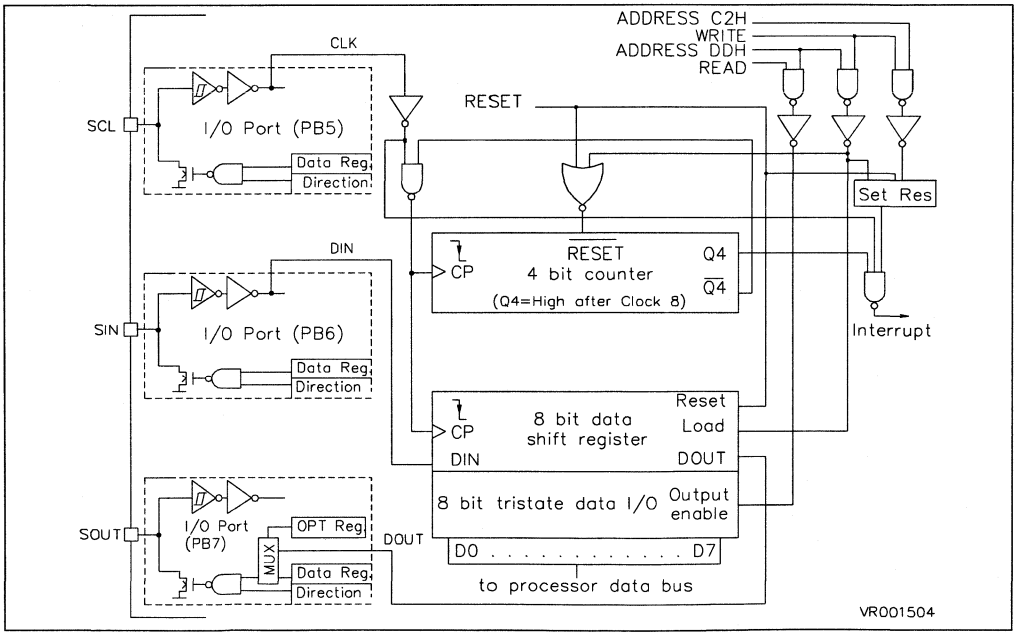
Three (Four) Wire Serial Bus

It is possible to use a single general purpose I/O pin (with the corresponding interrupt enabled) as a "chip enable" pin. SCL acts as active or passive clock pin, Sin as data in and Sout as data out (four wire bus). Sin and Sout can be connected together externally to implement three wire bus.

Note:

When the SPI is not used, the three I/O lines (Sin, SCL, Sout) can be used as normal I/O, with the following limitation: bit Sout cannot be used in open drain mode as this enables the shift register output to the port.
It is recommended, in order to avoid spurious interrupts from the SPI, to disable the SPI interrupt (the default state after reset) i.e. no write must be made to the 8-bit shift register (DDh). An explicit interrupt disable may be made in software by a dummy write to address C2h.

Figure 39. SPI Block Diagram



LCD CONTROLLER-DRIVER

The ST6242 LCD driver consists of a LCD control logic, a programmable prescaler, a 24 bytes wide dedicated LCD RAM, 40 segment and 4 common outputs. This allows a direct driving of up to 160 LCD segments.

The LCD driver is managed by the LCD Mode/Control register located at data RAM address DCh. Different display modes (1/1 duty, 1/2 duty, 1/3 duty and 1/4 duty) are available to cover a wide range of application requirements. The multiplexing display modes are software selectable by programming bits 6 and 7 of the LCD control register. Bits 0-5 are used to select the LCD drive and frame frequency (in relation to the system clock) and to switch off all segments.

According to the data in the LCD RAM, the segment and the common drivers generate the segment and common signals which can directly drive an LCD panel.

The LCD control logic reads automatically the data from the LCD RAM independently and without interruption of the processor. The part of the LCD RAM that is not used for displaying can be used as normal data memory.

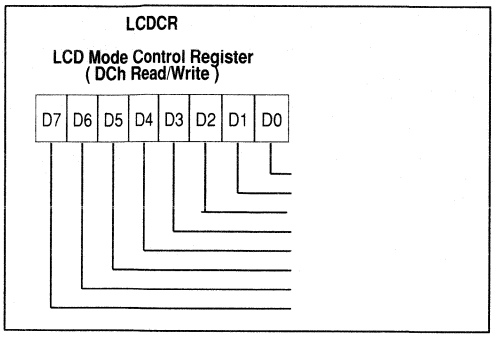
The scale factor of the clock prescaler can be fixed by software, therefore different frame frequencies can be defined.

The ST6242 oscillator should operate with a 1.0486, 2.0972, 4.1943, 8.3886MHz frequency quartz crystal. This allows the associated division rates to achieve an internal reference frequency of 32.768kHz. The different division rates can be achieved by programming bits 3, 4, 5 in the LCD control register (see Table 14). It is not recommended to select an internal frequency lower than 32.768kHz as the clock supervisor circuit may switch off the LCD peripheral if the lower frequency is detected.

When the display is turned off, all segment and common outputs are switched to ground, causing all the segments to be switched off regardless of the contents of the LCD RAM.

To avoid incomplete frames of the LCD, the mode control bits do not immediately influence the LCD controller when the LCD control register is written. They are stored in a temporary register and change the LCD function only at the end of the frame. Different LCD frame frequencies for each display mode are selected by bits in the LCD control register (see Table 12).

Figure 40. LCD Mode Control Register



DS0, DS1. Duty cycle select bits. These bits select the number of common backplanes used by the LCD control. This allows different multiplexing conditions.

HF0, HF1, HF2. These bits allow the LCD controller to be supplied with the correct frequency when different high main oscillator frequencies are selected as system clock. Table 11 shows the set-up for different clock crystals.

LF0, LF1, LF2. These bits control the LCD base operational frequency of the LCD common lines. Table 12 shows the set-up to select the different frequencies while Table 13 shows the corresponding frame values with the different multiplexing conditions.

Table 10. Duty Cycle Selection

DS1	DS0	Display Mode	Active Blackplanes	Max. Number of Segments Driven
0	0	1/4 duty	COM1, 2, 3, 4	180
0	1	1/1 duty	COM1	45
1	0	1/2 duty	COM1, 2	90
1	1	1/3 duty	COM1, 2, 3	135

LCD CONTROLLER-DRIVER (Continued)

Table 11. High Frequency Select Bits

HF2	HF1	HF0	Function	fosc
0	0	0	Display off	
0	0	1	for stand-by Oscillator	32.768kHz
0	1	0	NOT TO BE USED	
0	1	1	÷ 32 for main oscillator	1.048MHz
1	0	0	÷ 64 for main oscillator	2.097MHz
1	0	1	÷ 128 for main oscillator	4.194MHz
1	1	0	÷ 256 for main oscillator	8.388MHz
1	1	1	NOT TO BE USED	

Notes :

1. The usage f_{osc} values different from those defined in this table cause the LCD to operate at a reference frequency different from 32.768kHz.
2. It is not recommended to select an internal frequency lower than 32.768kHz as the clock supervisor circuit may switch off the LCD peripheral if lower frequency is detected.

Table 12. LCD Frequency Select Bits

LF2	LF1	LF0	f_{LCD} (Hz)
0	0	0	64
0	0	1	85
0	1	0	128
0	1	1	171
1	0	0	256
1	0	1	341
1	1	0	512
1	1	1	Not to be Used

Table 13. Available Frame Frequencies for LCD

f_{LCD} (Hz)	Frame Frequency f_F (Hz)			
	1/1 duty	1/2 duty	1/3 duty	1/4 duty
512	512	256	171	128
341	341	171	114	85
256	256	128	85	64
171	171	85	57	43
128	128	64	43	32
85	85	43	28	21
64	64	32	21	16

According to the selected LCD drive frequency f_{LCD} the frame frequencies come out as shown in Table 13.

The Figure 47 illustrates the waveforms of the different duty signals.•

The value of the VLCD voltage can be chosen independently from V_{DD} according to the display requirements. The intermediate VLCD levels 2/3 VLCD, 1/3 VLCD and 1/2 VLCD are generated by an internal resistor network as shown in Figures 45 and 46. The half VLCD level for 1/2 duty cycle is obtained by the external connection of VLCD1/3 and VLCD2/3 pins. All intermediate VLCD levels are connected to pins to enable external capacitive buffering or resistive shunting.

LCD CONTROLLER-DRIVER (Continued)

The internal resistive divider network is realized with two parallel dividers. One has high resistivity, the other one low resistivity. The high resistive divider (R_H) is permanently switched on during the LCD operation. The low resistive divider (R_L) is only switched on for a short period of time when the levels of common lines and segment lines are changed. This method combines low source impedance for fast switching of the LCD pixels with high source impedance for low power consumption. Figure 41 shows the typical current into V_{LCD} pin in dependency of the display voltage V_{LCD}.

When the display is switched off (by program or reset) the internal resistor network is also switched off to achieve minimum power consumption. The low resistivity divider is active at each edge of f_{LCD} during 8 clock cycles of F_{32kHz}.

The internal resistor network is implemented with resistive transistor elements to achieve high precision. For display voltages $V_{LCD} < 4.5V$ the resistivity of the divider may be too high for some applications (especially using 1/3 or 1/4 duty display mode). In that case an external resistive divider must be used to achieve the desired resistivity.

Figure 41. Typical Current Consumption on VLCD Pin (25°C, no load, fLCD=512 Hz, mux=1/3-1/4)

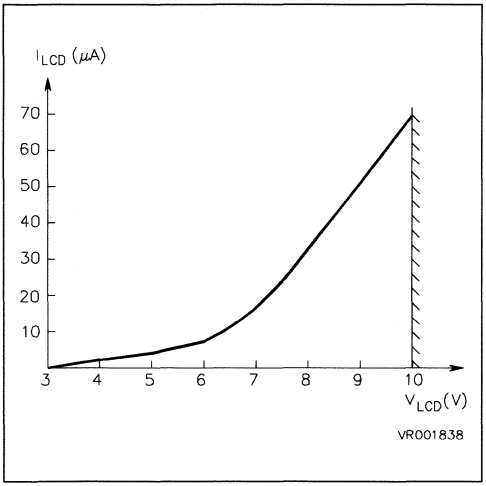
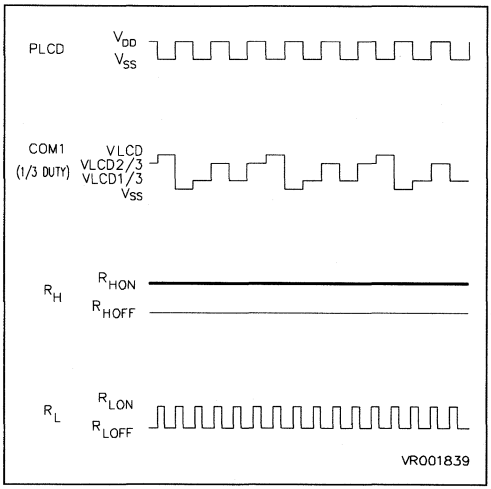


Figure 42. Typical Chronogram of Activation of the V_{LCD} Divider Network



LCD CONTROLLER-DRIVER (Continued)

Typical External resistances values are in the range of 100 kΩ to 150 kΩ. External capacitances in the range of 10 to 47 nF can be added to V_{LCD} 2/3 and V_{LCD} 1/3 pins and to V_{LCD} if the V_{LCD} connection is highly impedant.

When the program is switched off (by program or reset) the internal resistor network is also switched off to achieve minimum power consumption.

Figure 43. Typical Network to connect to V_{LCD} pins if $V_{LCD} \leq 4.5V$

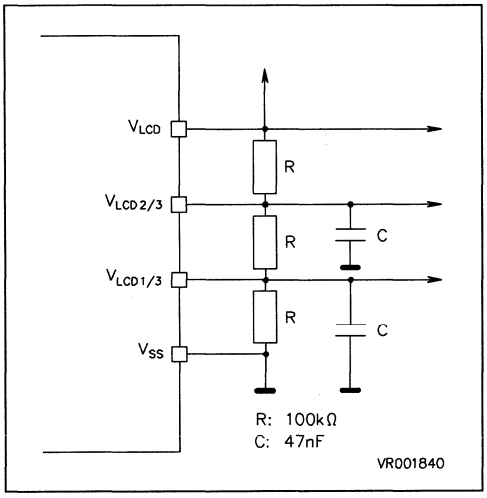


Figure 44. Generation of the 32kHz clock

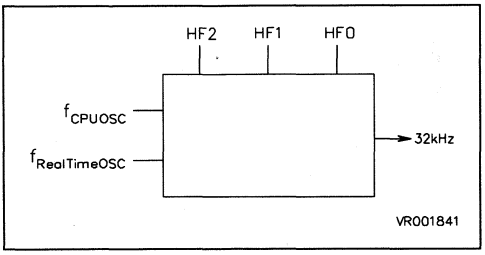


Figure 45. Bias Configuration for 1/2 Duty

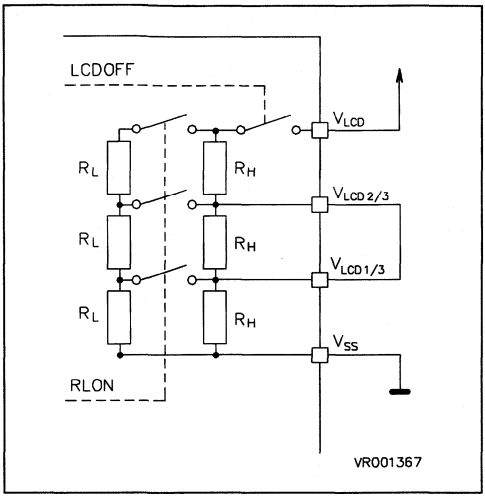
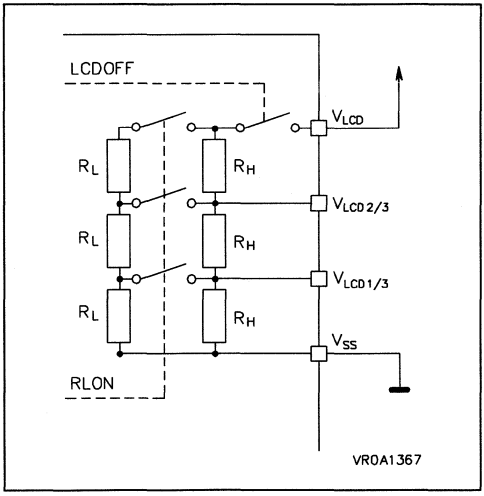


Figure 46. Bias Configuration for 1/1, 1/3 and 1/4 Duty Operation of LCD



LCD CONTROLLER-DRIVER (Continued)

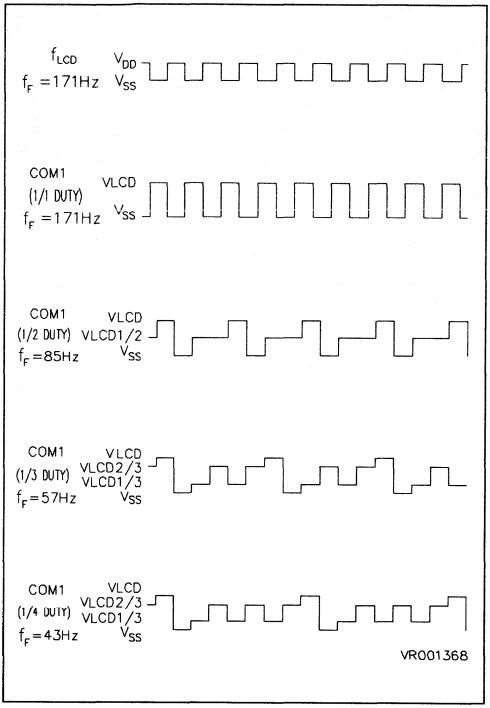
Address Mapping of the Display Segments.

The LCD RAM is located in the ST6242 data space from addresses E0h to F7h. The LCD forms a matrix of 45 segment lines (rows) and up to 4 common lines (columns). Each bit of the LCD RAM is mapped to one element of the LCD matrix, as described in Figure 48. If a bit is set, the corresponding LCD segment is switched on; if it is reset, the segment is switched off. The segments outputs S1, S2 and S3 are not connected to any pin.

When multiplex rates lower than 1/4 are selected, the unused LCD RAM is free for general use. In the 1/2 duty mode, for instance, half of the LCD RAM is available for storing general purpose data. The address range from F8h to FEh can be used as general purpose data RAM, but not for displaying data (it is reserved for future LCD expansion).

After a reset, the LCD RAM is not initializated and contains arbitrary information. As the LCD control register is reset, the LCD is completely switched off.

Figure 47. Common Signal Waveforms



LCD CONTROLLER-DRIVER (Continued)

Figure 48. Addressing Map of the LCD RAM

Data RAM Address	MSB							LSB	
E0 *									
E1	S16	S15	S14	S13	S12	S11	S10	S9	
E2	S24	S23	S22	S21	S20	S19	S18	S17	COM1
E3	S32	S31	S30	S29	S28	S27	S26	S25	
E4	S40	S39	S38	S37	S36	S35	S34	S33	
E5	S48	S47	S46	S45	S44	S43	S42	S41	
E6 *									
E7	S16	S15	S14	S13	S12	S11	S10	S9	
E8	S24	S23	S22	S21	S20	S19	S18	S17	COM2
E9	S32	S31	S30	S29	S28	S27	S26	S25	
EA	S40	S39	S38	S37	S36	S35	S34	S33	
EB	S48	S47	S46	S45	S44	S43	S42	S41	
EC *									
ED	S16	S15	S14	S13	S12	S11	S10	S9	
EE	S24	S23	S22	S21	S20	S19	S18	S17	COM3
EF	S32	S31	S30	S29	S28	S27	S26	S25	
F0	S40	S39	S38	S37	S36	S35	S34	S33	
F1	S48	S47	S46	S45	S44	S43	S42	S41	
F2 *									
F3	S16	S15	S14	S13	S12	S11	S10	S9	
F4	S24	S23	S22	S21	S20	S19	S18	S17	COM4
F5	S32	S31	S30	S29	S28	S27	S26	S25	
F6	S40	S39	S38	S37	S36	S35	S34	S33	
F7	S48	S47	S46	S45	S44	S43	S42	S41	
F8 - FE *									

Note *. Row to be used as general purpose RAM (not for display data)

Notes:

In STOP mode no clock is available for the LCD controller from the main oscillator. If the 32kHz oscillator is activated the LCD can also operate in STOP mode. If the stand-by oscillator is not active, the LCD controller is switched off when STOP instruction is executed; this mode has to be selected to reach the lowest power consumption.

A missing LCD clock (no oscillator active, broken crystal, etc.) is detected by a clock supervisor circuit that switches all the segments and common lines to ground to avoid destructive DC levels at the LCD.

The LCD function change is only effective at the end of a frame. For this reason special care has to be taken when entering the STOP mode. After switching the LCD clock source from the main oscillator to the 32kHz stand-by oscillator it must be guaranteed that enough clock pulses are delivered to complete the current frame before entering the STOP mode. Otherwise the LCD function will not be changed and the LCD is switched off after entering the STOP mode.

The RAM addresses E6-EC-F2-F8/FE are not used for LCD display purposes. So they are available as 10 additional Data RAM registers.

SOFTWARE DESCRIPTION

The ST62xx software has been designed to fully use the hardware in the most efficient way possible while keeping byte usage to a minimum, in short to provide byte efficient programming capability. The ST62xx core has the ability to set or clear any register or RAM location bit of the Data space with a single instruction. Furthermore, the program may branch to a selected address depending on the status of any bit of the Data space. The carry bit is stored with the value of the bit when the SET or RES instruction is processed.

Addressing Modes

The ST62xx core has nine addressing modes which are described in the following paragraphs. The ST62xx core uses three different address spaces: Program space, Data space, and Stack space. Program space contains the instructions which are to be executed, plus the data for immediate mode instructions. Data space contains the Accumulator, the X,Y,V and W registers, peripheral and Input/Output registers, the RAM locations and Data ROM locations (for storage of tables and constants). Stack space contains six 12-bit RAM cells used to stack the return addresses for subroutines and interrupts.

Immediate. In the immediate addressing mode, the operand of the instruction follows the opcode location. As the operand is a ROM byte, the immediate addressing mode is used to access constants which do not change during program execution (e.g., a constant used to initialize a loop counter).

Direct. In the direct addressing mode, the address of the byte that is processed by the instruction is stored in the location that follows the opcode. Direct addressing allows the user to directly address the 256 bytes in Data Space memory with a single two-byte instruction.

Short Direct. The core can address the four RAM registers X,Y,V,W (locations 80h, 81h, 82h, 83h) in the short-direct addressing mode . In this case, the instruction is only one byte and the selection of the location to be processed is contained in the opcode. Short direct addressing is a subset of the direct addressing mode. (Note that 80h and 81h are also indirect registers).

Extended. In the extended addressing mode, the 12-bit address needed to define the instruction is obtained by concatenating the four less significant bits of the opcode with the byte following the opcode. The instructions (JP, CALL) that use the extended addressing mode are able to branch to any address of the 4K bytes Program space.

An extended addressing mode instruction is two-byte long.

Program Counter Relative. The relative addressing mode is only used in conditional branch instructions. The instruction is used to perform a test and, if the condition is true, a branch with a span of -15 to +16 locations around the address of the relative instruction. If the condition is not true, the instruction that follows the relative instruction is executed. The relative addressing mode instruction is one-byte long. The opcode is obtained in adding the three most significant bits that characterize the kind of the test, one bit that determines whether the branch is a forward (when it is 0) or backward (when it is 1) branch and the four less significant bits that give the span of the branch (0h to Fh) that must be added or subtracted to the address of the relative instruction to obtain the address of the branch.

Bit Direct. In the bit direct addressing mode, the bit to be set or cleared is part of the opcode, and the byte following the opcode points to the address of the byte in which the specified bit must be set or cleared. Thus, any bit in the 256 locations of Data space memory can be set or cleared.

Bit Test & Branch. The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit test and branch instruction is three-byte long. The bit identification and the tested condition are included in the opcode byte. The address of the byte to be tested follows immediately the opcode in the Program space. The third byte is the jump displacement, which is in the range of -126 to +129. This displacement can be determined using a label, which is converted by the assembler.

Indirect. In the indirect addressing mode, the byte processed by the register-indirect instruction is at the address pointed by the content of one of the indirect registers, X or Y (80h,81h). The indirect register is selected by the bit 4 of the opcode. A register indirect instruction is one byte long.

Inherent. In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. These instructions are one byte long.

SOFTWARE DESCRIPTION (Continued)

Instruction Set

The ST62xx core has a set of 40 basic instructions. When these instructions are combined with nine addressing modes, 244 usable opcodes can be obtained. They can be divided into six different types:load/store, arithmetic/logic, conditional branch, control instructions, jump/call, bit manipulation. The following paragraphs describe the different types.

All the instructions within a given type are presented in individual tables.

Load & Store. These instructions use one,two or three bytes in relation with the addressing mode. One operand is the Accumulator for LOAD and the other operand is obtained from data memory using one of the addressing modes.

For Load Immediate one operand can be any of the 256 data space bytes while the other is always immediate data.

Table 14. Load & Store Instructions

Instruction	Addressing Mode	Bytes	Cycles	Flags	
				Z	C
LD A, X	Short Direct	1	4	Δ	*
LD A, Y	Short Direct	1	4	Δ	*
LD A, V	Short Direct	1	4	Δ	*
LD A, W	Short Direct	1	4	Δ	*
LD X, A	Short Direct	1	4	Δ	*
LD Y, A	Short Direct	1	4	Δ	*
LD V, A	Short Direct	1	4	Δ	*
LD W, A	Short Direct	1	4	Δ	*
LD A, rr	Direct	2	4	Δ	*
LD rr, A	Direct	2	4	Δ	*
LD A, (X)	Indirect	1	4	Δ	*
LD A, (Y)	Indirect	1	4	Δ	*
LD (X), A	Indirect	1	4	Δ	*
LD (Y), A	Indirect	1	4	Δ	*
LDI A, #N	Immediate	2	4	Δ	*
LDI rr, #N	Immediate	3	4	*	*

Notes:
X,Y. Indirect Register Pointers, V & W Short Direct Registers
#. Immediate data (stored in ROM memory)
rr. Data space register
Δ. Affected
*. Not Affected

SOFTWARE DESCRIPTION (Continued)

Arithmetic and Logic. These instructions are used to perform the arithmetic calculations and logic operations. In AND, ADD, CP, SUB instructions one operand is always the accumulator while the other can be either a data space memory content or an immediate value in relation with the addressing mode. In CLR, DEC, INC instructions the operand can be any of the 256 data space addresses. In COM, RLC, SLA the operand is always the accumulator.

Table 15. Arithmetic & Logic Instructions

Instruction	Addressing Mode	Bytes	Cycles	Flags	
				Z	C
ADD A, (X)	Indirect	1	4	Δ	Δ
ADD A, (Y)	Indirect	1	4	Δ	Δ
ADD A, rr	Direct	2	4	Δ	Δ
ADDI A, #N	Immediate	2	4	Δ	Δ
AND A, (X)	Indirect	1	4	Δ	*
AND A, (Y)	Indirect	1	4	Δ	*
AND A, rr	Direct	2	4	Δ	*
ANDI A, #N	Immediate	2	4	Δ	*
CLR A	Short Direct	2	4	Δ	Δ
CLR rr	Direct	3	4	*	*
COM A	Inherent	1	4	Δ	Δ
CP A, (X)	Indirect	1	4	Δ	Δ
CP A, (Y)	Indirect	1	4	Δ	Δ
CP A, rr	Direct	2	4	Δ	Δ
CPI A, #N	Immediate	2	4	Δ	Δ
DEC X	Short Direct	1	4	Δ	*
DEC Y	Short Direct	1	4	Δ	*
DEC V	Short Direct	1	4	Δ	*
DEC W	Short Direct	1	4	Δ	*
DEC A	Direct	2	4	Δ	*
DEC rr	Direct	2	4	Δ	*
DEC (X)	Indirect	1	4	Δ	*
DEC (Y)	Indirect	1	4	Δ	*
INC X	Short Direct	1	4	Δ	*
INC Y	Short Direct	1	4	Δ	*
INC V	Short Direct	1	4	Δ	*
INC W	Short Direct	1	4	Δ	*
INC A	Direct	2	4	Δ	*
INC rr	Direct	2	4	Δ	*
INC (X)	Indirect	1	4	Δ	*
INC (Y)	Indirect	1	4	Δ	*
RLC A	Inherent	1	4	Δ	Δ
SLA A	Inherent	2	4	Δ	Δ
SUB A, (X)	Indirect	1	4	Δ	Δ
SUB A, (Y)	Indirect	1	4	Δ	Δ
SUB A, rr	Direct	2	4	Δ	Δ
SUBI A, #N	Immediate	2	4	Δ	Δ

Notes:
X,Y. Indirect Register Pointers, V & W Short Direct Registers
#. Immediate data (stored in ROM memory)
rr. Data space register
Δ. Affected
*. Not Affected

SOFTWARE DESCRIPTION (Continued)

Conditional Branch. The branch instructions achieve a branch in the program when the selected condition is met.

Bit Manipulation Instructions. These instructions can handle any bit in data space memory. One group either sets or clears. The other group (see Conditional Branch) performs the bit test branch operations.

Control Instructions. The control instructions control the MCU operations during program execution.

Jump and Call. These two instructions are used to perform long (12-bit) jumps or subroutines call inside the whole program space.

Table 16. Conditional Branch Instructions

Instruction	Branch If	Bytes	Cycles	Flags	
				Z	C
JRC e	C = 1	1	2	*	*
JRNC e	C = 0	1	2	*	*
JRZ e	Z = 1	1	2	*	*
JRNZ e	Z = 0	1	2	*	*
JRR b, rr, ee	Bit = 0	3	5	*	Δ
JRS b, rr, ee	Bit = 1	3	5	*	Δ

Notes:
b. 3-bit address
e. 5 bit signed displacement in the range -15 to +16
ee. 8 bit signed displacement in the range -126 to +129
rr. Data space register
Δ. Affected
*. Not Affected

Table 17. Bit Manipulation Instructions

Instruction	Addressing Mode	Bytes	Cycles	Flags	
				Z	C
SET b,rr	Bit Direct	2	4	*	*
RES b,rr	Bit Direct	2	4	*	*

Notes:
b. 3-bit address;
rr. Data space register;
*. Not Affected

Table 18. Control Instructions

Instruction	Addressing Mode	Bytes	Cycles	Flags	
				Z	C
NOP	Inherent	1	2	*	*
RET	Inherent	1	2	*	*
RETI	Inherent	1	2	Δ	Δ
STOP (1)	Inherent	1	2	*	*
WAIT	Inherent	1	2	*	*

Notes:
1. This instruction is deactivated and a WAIT is automatically executed instead of a STOP if the Watchdog function is selected.
Δ. Affected
*. Not Affected

Table 19. Jump & Call Instructions

Instruction	Addressing Mode	Bytes	Cycles	Flags	
				Z	C
CALL abc	Extended	2	4	*	*
JP abc	Extended	2	4	*	*

Notes:
abc. 12-bit address;
*. Not Affected

SOFTWARE DESCRIPTION (Continued)

Opcode Map Summary. The following table contains an opcode map for the instructions used on the MCU.

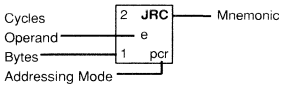
LOW / HI	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111	8 1000	9 1001	A 1010	B 1011	C 1100	D 1101	E 1110	F 1111	LOW / HI
0 0000	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b0,rr,ee 3 bt	2 JRZ e 1 pcr	# 1 pcr	2 JRC e 1 prc	4 LD a,(x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b0,rr 2 b.d	2 JRZ e 1 pcr	4 LDI rr,nn 3 imm	2 JRC e 1 pcr	4 LD a,(y) 1 ind	0 0000
1 0001	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b0,rr,ee 3 bt	2 JRZ e 1 pcr	4 INC x 1 sd	2 JRC e 1 prc	4 LDI a,nn 2 imm	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b0,rr 2 b.d	2 JRZ e 1 pcr	4 DEC x 1 sd	2 JRC e 1 pcr	4 LD a,rr 2 dir	1 0001
2 0010	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b4,rr,ee 3 bt	2 JRZ e 1 pcr	# 1 pcr	2 JRC e 1 prc	4 CP a,(x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b4,rr 2 b.d	2 JRZ e 1 pcr	4 COM a 1 inh	2 JRC e 1 pcr	4 CP a,(y) 1 ind	2 0010
3 0011	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b4,rr,ee 3 bt	2 JRZ e 1 pcr	4 LD a,x 1 sd	2 JRC e 1 prc	4 CPI a,nn 2 imm	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b4,rr 2 b.d.	2 JRZ e 1 pcr	4 LD x,a 1 sd	2 JRC e 1 pcr	4 CP a,rr 2 dir	3 0011
4 0100	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b2,rr,ee 3 bt	2 JRZ e 1 pcr	# 1 pcr	2 JRC e 1 prc	4 ADD a,(x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b2,rr 2 b.d	2 JRZ e 1 pcr	2 RETI 1 inh	2 JRC e 1 pcr	4 ADD a,(y) 1 ind	4 0100
5 0101	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b2,rr,ee 3 bt	2 JRZ e 1 pcr	4 INC y 1 sd	2 JRC e 1 prc	4 ADDI a,nn 2 imm	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b2,rr 2 b.d	2 JRZ e 1 pcr	4 DEC y 1 sd	2 JRC e 1 pcr	4 ADD a,rr 2 dir	5 0101
6 0110	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b6,rr,ee 3 bt	2 JRZ e 1 pcr	# 1 pcr	2 JRC e 1 prc	4 INC (x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b6,rr 2 b.d	2 JRZ e 1 pcr	2 STOP 1 inh	2 JRC e 1 pcr	4 INC (y) 1 ind	6 0110
7 0111	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b6,rr,ee 3 bt	2 JRZ e 1 pcr	4 LD a,y 1 sd	2 JRC e 1 prc	# 	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b6,rr 2 b.d	2 JRZ e 1 pcr	4 LD y,a 1 sd	2 JRC e 1 pcr	4 INC rr 2 dir	7 0111
8 1000	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b1,rr,ee 3 bt	2 JRZ e 1 pcr	# 1 pcr	2 JRC e 1 prc	4 LD (x),a 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b1,rr 2 b.d	2 JRZ e 1 pcr	# 	2 JRC e 1 pcr	4 LD (y),a 1 ind	8 1000
9 1001	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b1,rr,ee 3 bt	2 JRZ e 1 pcr	4 INC v 1 sd	2 JRC e 1 prc	# 	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b1,rr 2 b.d	2 JRZ e 1 pcr	4 DEC v 1 sd	2 JRC e 1 pcr	4 LD rr,a 2 dir	9 1001
A 1010	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b5,rr,ee 3 bt	2 JRZ e 1 pcr	# 1 pcr	2 JRC e 1 prc	4 AND a,(x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b5,rr 2 b.d	2 JRZ e 1 pcr	4 RLC a 1 inh	2 JRC e 1 pcr	4 AND a,(y) 1 ind	A 1010
B 1011	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b5,rr,ee 3 bt	2 JRZ e 1 pcr	4 LD a,v 1 sd	2 JRC e 1 prc	4 ANDI a,nn 2 imm	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b5,rr 2 b.d	2 JRZ e 1 pcr	4 LD v,a 1 sd	2 JRC e 1 pcr	4 AND a,rr 2 dir	B 1011
C 1100	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b3,rr,ee 3 bt	2 JRZ e 1 pcr	# 1 pcr	2 JRC e 1 prc	4 SUB a,(x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b3,rr 2 b.d	2 JRZ e 1 pcr	2 RET 1 inh	2 JRC e 1 pcr	4 SUB a,(y) 1 ind	C 1100
D 1101	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b3,rr,ee 3 bt	2 JRZ e 1 pcr	4 INC w 1 sd	2 JRC e 1 prc	4 SUBI a,nn 2 imm	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b3,rr 2 b.d	2 JRZ e 1 pcr	4 DEC w 1 sd	2 JRC e 1 pcr	4 SUB a,rr 2 dir	D 1101
E 1110	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b7,rr,ee 3 bt	2 JRZ e 1 pcr	# 1 pcr	2 JRC e 1 prc	4 DEC (x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b7,rr 2 b.d	2 JRZ e 1 pcr	2 WAIT 1 inh	2 JRC e 1 pcr	4 DEC (y) 1 ind	E 1110
F 1111	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b7,rr,ee 3 bt	2 JRZ e 1 pcr	4 LD a,w 1 sd	2 JRC e 1 prc	# 	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b7,rr 2 b.d	2 JRZ e 1 pcr	4 LD w,a 1 sd	2 JRC e 1 pcr	4 DEC rr 2 dir	F 1111

Abbreviations for Addressing Modes:

dir	Direct
sd	Short Direct
imm	Immediate
inh	Inherent
ext	Extended
b.d	Bit Direct
bt	Bit Test
pcr	Program Counter Relative
ind	Indirect

Legend:

#	Indicates Illegal Instructions
e	5 Bit Displacement
b	3 Bit Address
rr	1byte dataspace address
nn	1 byte immediate data
abc	12 bit address
ee	8 bit Displacement



ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings

This product contains devices to protect the inputs against damage due to high static voltages, however it is advised to take normal precaution to avoid application of any voltage higher than maximum rated voltages.

For proper operation it is recommended that V_I and V_O must be higher than V_{SS} and smaller than V_{DD}. Reliability is enhanced if unused inputs are connected to an appropriated logic voltage level (V_{DD} or V_{SS}).

Power Considerations. The average chip-junction temperature, Tj, in Celsius can be obtained from:

$$Tj = T_A + PD \times RthJA$$

Where : T_A = Ambient Temperature.

RthJA = Package thermal resistance (junction-to ambient).

PD = Pint + Pport.

Pint = $I_{DD} \times V_{DD}$ (chip internal power).

Pport = Port power dissipation (determinated by the user).

Symbol	Parameter	Value	Unit
V_{DD}	Supply Voltage	-0.3 to 7.0	V
V_{LCD}	Display Voltage	-0.3 to 11.0	V
V_I	Input Voltage	V_{SS} - 0.3 to V_{DD} + 0.3	V
V_O	Output Voltage	V_{SS} - 0.3 to V_{DD} + 0.3	V
I_O	Current Drain per Pin Excluding V_{DD} & V_{SS}	± 10	mA
IV_{DD}	Total Current into V_{DD} (source)	50	mA
IV_{SS}	Total Current out of V_{SS} (sink)	50	mA
Tj	Junction Temperature	150	°C
T_{STG}	Storage Temperature	-60 to 150	°C

Note : Stresses above those listed as "absolute maximum ratings" may cause permanent damage to the device . This is a stress rating only and functional operation of the device at these conditions is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

THERMAL CHARACTERISTIC

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
RthJA	Thermal Resistance	PQFP64		70		°C/W

RECOMMENDED OPERATING CONDITIONS

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
T_A	Operating Temperature	1 Suffix Version 6 Suffix Version	0 -40		70 85	°C
V_{DD}	Operating Supply Voltage		3		6	V
V_{LCD}	Display Voltage		3		10	V
V_{DD}	RAM Retention Voltage		2			V

RECOMMENDED OPERATING CONDITIONS (Continued)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
f_{OSC}	Oscillator Frequency (1)(4)	$V_{DD} \geq 4.5V$ $V_{DD} \geq 3V$	0.01 0.01		8.388 2	MHz
I_{INJ+}	Pin Injection Current (positive) Digital Input (2) Analog Input (3)	V_{DD} = 4.5 to 5.5V			+5	mA
I_{INJ-}	Pin Injection Current (negative) Digital Input (2) Analog Input	V_{DD} = 4.5 to 5.5V			-5	mA

Notes :
1. An oscillator frequency above 1MHz is recommanded for reliable A/D results.
2. A current of ± 5mA can be forced on each pin of the digital section without affecting the functional behaviour of the device. For a positive current injected into one pin, a part of this current (~ 10%) can be expected to flow from the neighbouring pins. A current of -5mA can be forced on one input of the analog section at a time (or -2.5mA for all inputs at a time) without affecting the conversion.
3. If a total current of +1mA is flowing into the single analog channel or if the total current flowing into all the analog inputs is of 1mA, all the conversion is resulting shifted of +1LSB. If a total positive current of +5mA is flowing into the single analog channel or if the total current flowing into all the analog inputs is of 5mA, all the conversion is resulting shifted of +2LSB.
4. Operation below 0.01 MHz is possible but requires increased supply current.

EEPROM INFORMATION

The ST62xx EEPROM single poly process has been specially developed to achieve 300.000 Write/Erase cycles and a 10 years data retention.

DC ELECTRICAL CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
V_{IL}	Input Low Level Voltage	RESET, NMI, TIMER, WDON Pin			$0.3V_{DD}$	V
V_{IH}	Input High Level Voltage	TIMER	$0.80V_{DD}$			V
		RESET, NMI, WDON Pin	$0.70V_{DD}$			V
I_{IL} I_{IH}	Input Leakage Current	RESET Pin V_{DD} =5V $V_{IN} = V_{DD}$ (1) $V_{IN} = V_{DD}$ (2) $V_{IN} = V_{SS}$ (5)			10 1 50	µA mA µA
V_{OL}	Low Level Output Voltage	TIMER, I_{OL} = 5.0mA			$0.2V_{DD}$	V
V_{OH}	High Level Output Voltage	TIMER, I_{OL} = -5.0mA	$0.65V_{DD}$			V
R_{PU}	Pull-up Resistor	V_{IN}=0V V_{DD}=5V WDON - NMI	40	100	200	kΩ
		RESET	200	300	500	kΩ

Notes on next page

DC ELECTRICAL CHARACTERISTICS (Continued)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
I_{IL} I_{IH}	Input Leakage Current	TIMER $V_{IN} = V_{DD}$ or V_{SS}		0.1	1.0	µA
I_{IL} I_{IH}	Input Leakage Current	NMI VDD = 5V $V_{IN} = V_{SS}$ (5) $V_{IN} = V_{DD}$			 100 1.0	µA
I_{IL} I_{IH}	Input Leakage Current	WDON VDD = 5V $V_{IN} = V_{SS}$ (5) $V_{IN} = V_{DD}$			 100 1.0	µA
I_{DD}	Supply Current RUN Mode	f_{OSC} = 8MHz, I_{LOAD} = 0mA V_{DD} = 5.5V		4	7	mA
	Supply Current WAIT Mode (4)	f_{OSC} = 8MHz, I_{LOAD} = 0mA V_{DD} = 5.0V		1	3	mA
	Supply Current RESET Mode	f_{OSC} = 8MHz, $V_{RESET} = V_{SS}$		1	7	mA
	Supply Current STOP Mode (3)	I_{LOAD} = 0mA V_{DD} = 5.5V		1	10	µA

Notes :
1. No Watchdog Reset activated.
2. Reset generated by Watchdog.
3. When the watchdog function is actvated the STOP instruction is deactivated. WAIT instruction is automatically executed.
4. All on-chip peripherals in OFF state
5. Pull-up resistor

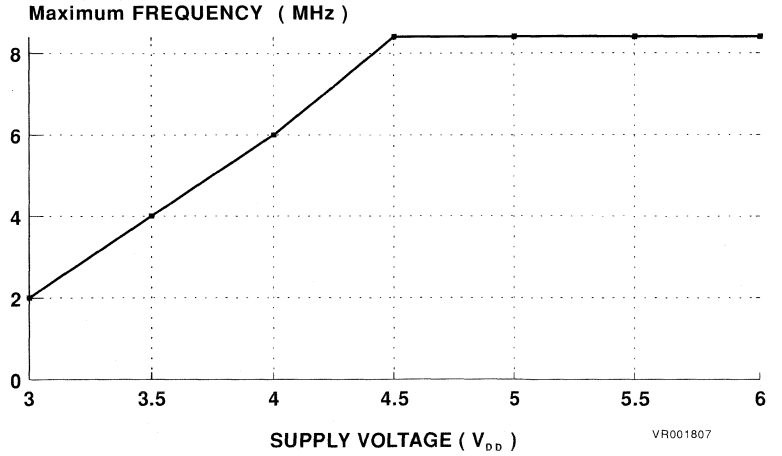
AC ELECTRICAL CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
f_{OSC}	Oscillator Frequency (2)	$V_{DD} \geq 4.5V$ $V_{DD} \geq 3V$	0.01		8.388 2	MHz
t_{SU}	Oscillator Start-up Time	$C_{L1} = C_{L2}$ = 22pF - crystal		5	10	ms
t_{SR}	Supply Rise Time	10% to 90%	0.01		100	
t_{REC}	Supply Recovery Time (1)		100			
T_W	Minimum Pulse Width	NMI Pin V_{DD} = 5V	100			ns
		RESET Pin	100			ns
T_{WEE}	EEPROM Write Time	T_A = 25°C One Byte T_A = 85°C One Byte		5 15	10 25	ms
Endurance	EEPROM WRITE/ERASE Cycles	Q_A L_{OT} Acceptance Criteria	300.000	> 1 million		cycles
Retention	EEPROM Data Retention	T_A = 55°C	10			years
C_{IN}	Input Capacitance	All Inputs Pins			10	pF

Notes:
1. Period for which V_{DD} has to be connected or at 0V to allow internal Reset function at next power-up.
2. Operation below 0.01 MHz is possible but requires increased supply current.

Maximum Operating FREQUENCY (Fmax) Versus SUPPLY VOLTAGE (V_{DD})



I/O PORTS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
C_{OUT}	Output Capacitance	All Outputs Pins			10	pF
V_{IL}	Input Low Level Voltage	I/O Pins			$0.3V_{DD}$	V
V_{IH}	Input High Level Voltage	I/O Pins	$0.7V_{DD}$			V
V_{OL}	Low Level Output Voltage	I/O Pins, I_O = 10µA (sink)			0.1	V
		I/O Pins, $I_{OL} = V_{DD}$x1mA V_{DD} = 4.5 to 6V			$0.16xV_{DD}$	V
		I/O Pins, I_{OL} = 1.6mA V_{DD} = 3V			0.4	V
	Low Level Output Voltage, PB4-PB7 Only	I/O Pins, $I_{OL} = V_{DD}$x2mA V_{DD} = 4.5 to 6V			$0.16xV_{DD}$	V
		I/O Pins, I_{OL} = 3.2mA V_{DD} = 3V			0.4	V
		I/O Pins, $I_{OL} = V_{DD}$x4mA V_{DD} = 4.5 to 6V			$0.26xV_{DD}$	V
		I/O Pins, I_{OL} = 6.4mA V_{DD} = 3V			0.8	V
V_{OH}	High Level Output Voltage	I/O Pins, I_O = -10µA (source)	V_{DD}-0.1			V
		I/O Pins, $I_{OL} = -V_{DD}$x1mA V_{DD} = 5.0V	$0.6xV_{DD}$			V
I_{IL} I_{IH}	Input Leakage Current	I/O Pins,(1)		0.1	1.0	µA
R_{PU}	Pull-up Resistor	I/O Pins V_{IN} = 0V, V_{DD} = 5.0V	40	100	200	KΩ

Note 1. Pull-up resistor off

SPI ELECTRICAL CHARACTERISTICS

(T_A = -40 to +85°C unless otherwhise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
F_{CL}	Clock Frequency	applied on PB5/SCL			500	kHz
t_{SU}	Set-up Time	applied on PB6/Sin		50		ns
t_h	Hold Time	applied on PB6/Sin		100		ns

A/D CONVERTER CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
Res	Resolution (3)			8		Bit
A_{TOT}	Total Accuracy (3)	f_{OSC} > 1.2 MHz f_{OSC} > 32kHz			± 2 ±4	LSB
t_C(1)	Conversion Time	f_{OSC} = 8MHz		70		µs
V_{AN}	Conversion Range		V_{SS}		V_{DD}	V
ZIR	Zero Input Reading	Conversion result when $V_{IN} = V_{SS}$	00			Hex
FSR	Full Scale Reading	Conversion result when $V_{IN} = V_{DD}$			FF	Hex
AD_I	Analog Input Current During Conversion	V_{DD}= 4.5V			1.0	µA
AC_{IN}(2)	Analog Input Capacitance			2	5	pF
ASI	Analog Source Impedance				30	kΩ
SSI	Analog Reference Supply Impedence				2	kΩ

Notes:
1. With oscillator frequencies less than 1MHz, the A/D Converter accuracy is decreased.
2. Excluding Pad Capacitance
3. Noise at V_{DD},$V_{SS} \leq$ 10mV

TIMER CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
t_{RES}	Resolution		$\frac{12}{fosc}$			second
f_{IN}	Input Frequency on TIMER Pin				$\frac{fosc}{8}$	MHz
t_W	Pulse Width at TIMER Pin	$V_{DD} \geq 3V$ $V_{DD} \geq 4.5V$	1 125			µs ns

LCD ELECTRICAL CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

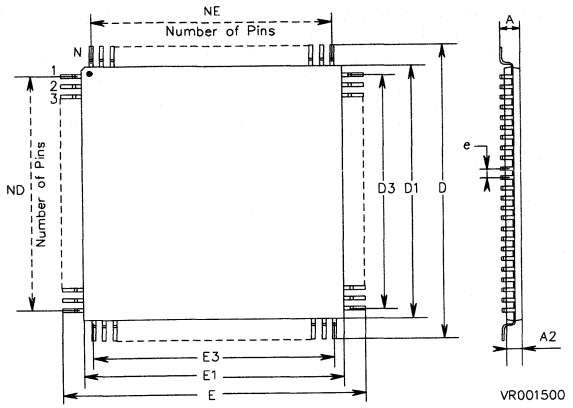
Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
f_{FR}	Frame Frequency	1/4 Duty f_{OSC} = 1, 2, 4, 8MHz	16		128	Hz
V_{OS}	DC Offset Voltage(1)	$V_{LCD} = V_{DD}$, no load			50	mV
V_{OH}	COM High Level, Output Voltage	I = 100µA, V_{LCD} = 5V	4.5V			V
V_{OL}	COM Low Level, Output Voltage	I = 100µA, V_{LCD} = 5V			0.5V	V
V_{OH}	SEG High Level, Output Voltage	I = 50µA, V_{LCD} = 5V	4.5V			V
V_{OL}	SEG Low Level, Output Voltage	I = 50µA, V_{LCD} = 5V			0.5V	V
V_{LCD}	Display Voltage	Note 2	3		10	V

Notes :

1. The DC offset voltage refers to all segment and common outputs. It is the difference between the measured voltage value and nominal value for every voltage level. Ri of voltage meter must be greater than or equal to 100MΩ.
2. An external resistances network is required when $V_{LCD} \leq 4.5V$.

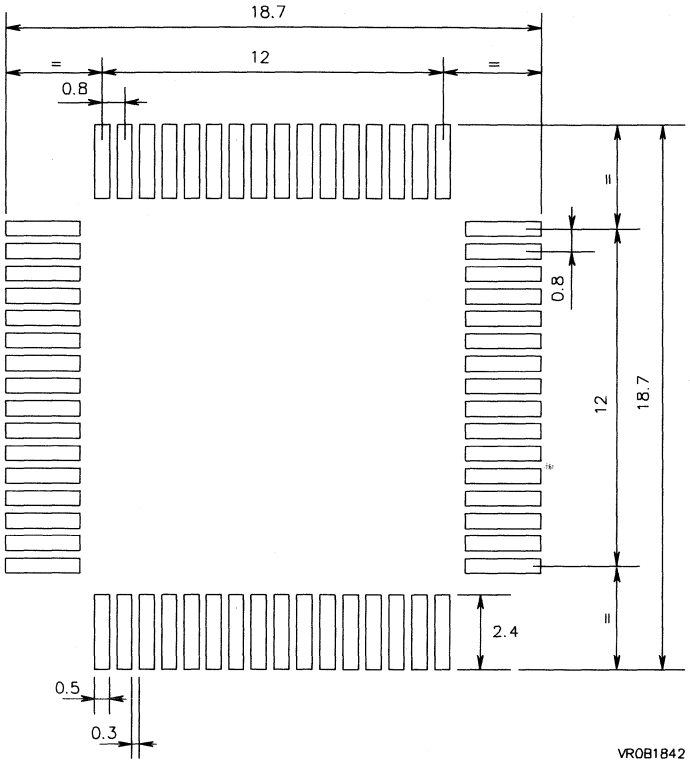
PACKAGE MECHANICAL DATA

Figure 49. ST6242 64 Pin Plastic Quad Flat Pack Package



Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A			3.40			0.134
A2	2.55	2.80	3.05	0.100	0.110	0.120
D	16.95	17.20	17.45	0.667	0.677	0.687
D1	13.90	14.00	14.10	0.547	0.551	0.555
D3		12.00			0.472	
E	16.95	17.20	17.45	0.667	0.677	0.687
E1	13.90	14.00	14.10	0.547	0.551	0.555
E3		12.00			0.472	
e		0.80			0.032	
Number of Pins						
ND	16					
NE	16					
N	64					

Recommanded Solder Pad Footprint For QFP64 (in mm)



ORDERING INFORMATION

The following chapter deals with the procedure for transfer the Program/Data ROM codes to SGS-THOMSON.

Communication of the ROM Codes. To communicate the contents of Program/Data ROM memories to SGS-THOMSON, the customer has to send :

- one file in INTEL INTELLEC 8/MDS FORMAT (in an MS-DOS 5" diskette) for the PROGRAM Memory
- one file in INTEL INTELLEC 8/MDS FORMAT (in a MS-DOS 5" diskette) for the EEPROM initial content (this file is optional)
- a filled Option List form as described in the OPTION LIST paragraph.

The program ROM should respect the ROM Memory Map as in Table 20.

The ROM code must be generated with ST6 assembler. Before programming the EPROM, the buffer of the EPROM programmer must be filled with FFh.

Table 20. ROM Memory Map

ROM Page	Device Address	Description
Page 0	0000h-007Fh 0080h-07FFh	Reserved User ROM
Page 1 "STATIC"	0800h-0F9Fh 0FA0h-0FEFh 0FF0h-0FF7h 0FF8h-0FFBh 0FFCh-0FFDh 0FFEh-0FFFh	User ROM Reserved Interrupt Vectors Reserved NMI Vector Reset Vector
Page 2	0000h-000Fh 0010h-07FFh	Reserved User ROM
Page 3	0000h-000Fh 0010h-07FFh	Reserved User ROM

Note : EPROM addresses are related to the ROM file to be processed.

Customer EEPROM Initial Contents : Format

a. The content should be written into an INTEL INTELLEC format file.

b. In the case of 128 bytes of EEPROM, the starting address in 000h and the end in 7Fh.

c. Undefined or don't care bytes should have the content FFh.

Listing Generation & Verification. When SGS-THOMSON receives the Codes, they are compared and a computer listing is generated from them. This listing refers exactly to the mask that will be used to produce the microcontroller. Then the listing is returned to the customer that must thoroughly check, complete, sign and return it to SGS-THOMSON. The signed list constitutes a part of the contractual agreement for the creation of the customer mask. SGS-THOMSON sales organization will provide detailed information on contractual points.

ORDERING INFORMATION TABLE

Sales Types	Temperature Range	Package
ST6242Q1/XX	0 to + 70°C	PQFP64
ST6242Q6/XX	-40 to + 85°C	PQFP64

Note : "XX" is the ROM code identifier allocated by SGS-THOMSON after receipt of all required options and the related ROM file.

ST6242 MICROCONTROLLER OPTION LIST

Customer . . . . . .

Address . . . . . .

Contact . . . . . .

Phone No . . . . . .

Reference . . . . . .

SGS-THOMSON Microelectronics references

Device	[ ] ST6242	
Package	[ ] Plastic Quad Flat Package	
Temperature Range	[ ] 0°C to + 70°C	[ ] -40°C to + 85°C

Special Marking [ ] No
[ ] Yes "_ _ _ _ _ _ _ _ _ _"

Authorized characters are Letters, digits, '.', '–', '/' and spaces only.
For marking one line with 10 characters maximum is possible.

Comments :
- Number of LCD segments used :
- Number of LCD backplanes used :

Note :

Signature

Date

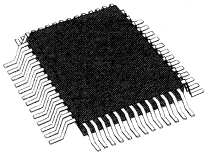
SGS-THOMSON MICROELECTRONICS

ST62E42
ST62T42

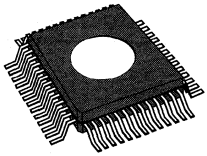
8-BIT EPROM HCMOS MCU WITH LCD DRIVER, AND A/D CONVERTER

PRELIMINARY DATA

- 3 to 6V supply operating range
- 8.4MHz Maximum Clock Frequency
- Run, Wait & Stop Modes
- 5 different interrupt vectors
- Look-up table capability in EPROM
- User EPROM: 7948 bytes
  Data RAM: 128 bytes
  LCD RAM: 24 bytes
- PQFP64 and CQFP64-W packages
- 10 fully software programmable I/O as:
  - Input with/without pull-up resistor
  - Input with interrupt generation
  - Open-Drain or Push-pull outputs
  - Analog Inputs (6 pins)
- 4 I/O lines can sink up to 20mA for direct LED or TRIAC driving and have SPI alternate functions
- 8-bit counters and 7-bit programmable prescalers
- Software activated digital watchdog
- 8-bit A/D converter with up to 6 analog inputs
- 8-bit synchronous serial peripheral interface (SPI)
- LCD driver with 40 segment outputs, 4 backplane outputs and selectable duty cycle for up to 160 LCD segments direct driving
- One external not maskable interrupt
- 9 powerful addressing modes
- The accumulator, the X, Y, V & W registers, the port and peripherals data & control registers are addressed in the data space as RAM locations.
- The ST62E42 is the EPROM version, ST62T42 is the OTP version, fully compatible with ST6242 ROM version.



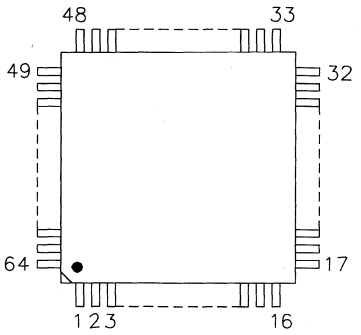
PQFP64



CQFP64-W

(Ordering Information at the end of the datasheet)

Figure 1.Pin Quad Flat Pack (QFP) Package Pinout



ST62E42/T42 Pin Description

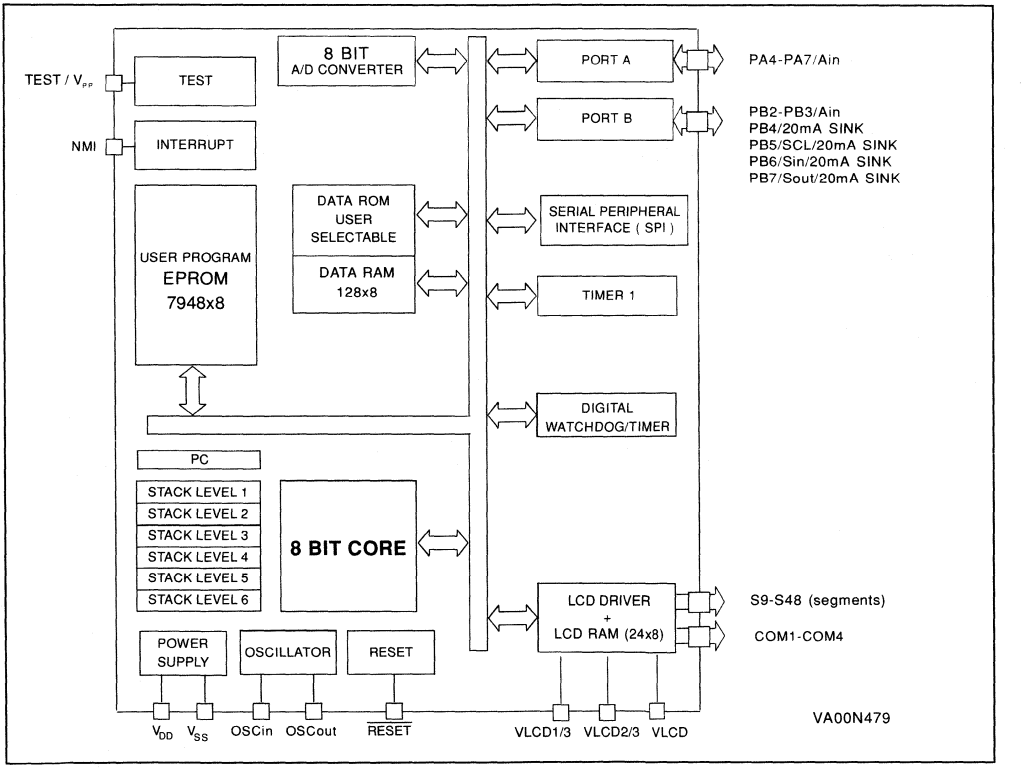
Pin number	Pin name	Pin number	Pin name	Pin number	Pin name	Pin number	Pin name
1	S45	17	V_{DD}	33	S13	49	S29
2	S46	18	V_{SS}	34	S14	50	S30
3	S47	19	$\overline{RESET}$	35	S15	51	S31
4	S48	29	OSCout	36	S16	52	S32
5	COM4	21	OSCin	37	S17	53	S33
6	COM3	22	NMI	38	S18	54	S34
7	COM2	23	PB7/Sout (1)	39	S19	55	S35
8	COM1	24	PB6/Sin (1)	40	S20	56	S36
9	VLCD1/3	25	PB5/SCL (1)	41	S21	57	S37
10	VLCD2/3	26	PB4 (1)	42	S22	58	S38
11	VLCD	27	PB3/Ain	43	S23	59	S39
12	PA7/Ain	28	PB2/Ain	44	S24	60	S40
13	PA6/Ain	29	S9	45	S25	61	S41
14	PA5/Ain	30	S10	46	S26	62	S42
15	PA4/Ain	31	S11	47	S27	63	S43
16	TEST/V_{PP}	32	S12	48	S28	64	S44

Note 1: 20mA SINK

GENERAL DESCRIPTION

The ST62E42,T42 microcontrollers are members of the 8-bit HCMOS ST62xx family, a series of devices oriented to low-medium complexity applications. They are the EPROM/OTP versions of the ST6242 ROM device and are suitable for product prototyping and low volume production. All ST62xx members are based on a building block approach: a common core is associated with a combination of on-chip peripherals (macrocells). The macrocells of the ST6242 family are: a high performance LCD controller/driver with 40 segment outputs and 4 backplanes able to drive up to 160 segments, a Timer peripheral including an 8-bit counter with a 7-bit software programmable prescaler, a digital watchdog timer (DWD), an 8-bit A/D Converter with up to 6 analog inputs and an 8-bit synchronous Serial Peripheral Interface (SPI). Thanks to these peripherals the ST6242 family is well suited for general purpose, automotive, security, appliance and industrial applications.

Figure 2. ST62E42 Block Diagram



Note: Ain = Analog Input

PIN DESCRIPTION

V_{DD} and V_{SS}. Power is supplied to the ST62E42 using these two pins. V_{DD} is power and V_{SS} is the ground connection.

OSCin and OSCout. These pins are internally connected with the on-chip oscillator circuit. A quartz crystal or a ceramic resonator can be connected between these two pins in order to allow the correct operation of the MCU with various stability/cost trade-offs. OSCin is the input pin, OSCout is the output pin. An external clock signal can be applied to OSCin.

$\overline{\text{RESET}}$. The active low $\overline{\text{RESET}}$ pin is used to restart the microcontroller at the beginning of its program. The $\overline{\text{RESET}}$ pin is provided with an on-chip pull-up resistor and schmitt trigger input characteristics.

TEST/V_{PP}. The TEST pin is used to place the MCU into special operating mode. TEST must be held at V_{SS} for normal operation (an internal pull-down resistor is present to select normal operating mode if TEST pin is not connected). If this pin is connected to a +12.5V level during the reset phase, the EPROM programming mode is entered.

NMI. The NMI pin provides the capability for asynchronous applying an external top priority interrupt to the ST62E42.This pin is falling edge sensitive. The NMI pin is provided with an on-chip pull-up resistor and schmitt trigger input characteristics.

PA4-PA7. These 4 lines are organized as one I/O port (A). Each line may be configured under software control as an input with or without pull-up resistor, interrupt generating input with pull-up resistor, open-drain or push-pull output or as analog input for the A/D converter. Port A has a 5mA drive capability in output mode.

PB2-PB3,PB4-PB7. These 6 lines are organized as one I/O port (B). Each line may be configured under software control as an input with or without internal pull-up resistor, interrupt generating input with pull-up resistor, open-drain or push-pull output. PB2-PB3 can be programmed as analog inputs for the A/D converter while PB4-PB7 can also sink 20mA for direct LED driving. PB5-PB7 can also be used as respectively Clock, Data in and Data out pins for the on-chip SPI to carry the synchronous serial I/O signals.

COM1-COM4. These four pins are the LCD peripheral common outputs. They are the outputs of the on-chip backplane voltage generator which is used for multiplexing the 40 LCD lines allowing up to 160 segments to be driven.

S9-S48. These pins are the 40 LCD peripheral driver outputs of ST62E42. Segments S1-S8 are not connected to any pin.

VLCD. Display voltage supply. It determines the high voltage level on COM1-COM4 and S9-S48 pins.

VLCD1/3, VLCD2/3. Display supply voltage inputs for determining the display voltage levels on COM1-COM4 and S9-S48 pins during multiplex operation.

ST62E42,T42 EPROM/OTP DESCRIPTION

The ST62E42 is the EPROM version of the ST6242 ROM product. It is intended for use during the development of an application, and for pre-production and small volume production. The ST62T42 OTP has the same characteristics. Both include EPROM memory instead of the ROM memory of the ST6242, and so the program and constants of the program can be easily modified by the user with the ST62E42 EPROM programming board from SGS-THOMSON.

From a user point of view (with the following exception) the ST62E42,T42 products have exactly the same software and hardware features of the ROM version. An additional mode is used to configure the part for programming of the EPROM, this is set by a +12.5V voltage applied to the TEST/V_{PP} pin. The programming of the ST62E42,T42 is described in the User Manual of the EPROM Programming board.

On the ST62E42, all the 8192 bytes of PROGRAM memory are available for the user, as all the EPROM memory can be erased by exposure to UV light. On the ST62T42 (OTP) device) a reserved area for test purposes exists, as for the ST6242 ROM device. In order to avoid any discrepancy between program functionality when using the EPROM, OTP and ROM it is recommended not to use these reserved areas, even when using the ST62E42.

Notes on programming:

In order to emulate exactly the ST6242 features with the ST62E42 and ST6242, some software precautions have to be taken:

1. Data RAM: The data entered in the Data RAM bank register (CBh) must be 08h.

2. I/O: To prevent floating input or uncontrolled I/O interrrupt on the EPROM/OTP devices, the port bits PA0-PA3, PB0, PB1 must be programmed as push-pull outputs.

3. Timer: The bit "TOUT" of the Timer status control register (D4h) must be set "1" (timer in output mode).

4. Data Memory Space: Write 40h at the address DFh of the Data Memory Space (desabled EE).

5. When programming for the EPROM/OTP parts, it is suggested that the conditional assembly technique is used for controlling the I/O ports in order to disable the appropriate code for the ROM device.

6. Do not access data space locations D5h, D6h, D7h, DAh, DBh.

Other than this exception, the ST62E42,T42 parts are fully compatible with the ROM ST6242 equivalent, this datasheet thus provides only information specific to the EPROM based devices.

THE READER IS ASKED TO REFER TO THE DATASHEET OF THE ST6242 ROM-BASED DEVICE FOR FURTHER DETAILS.

EPROM ERASING

The EPROM of the windowed package of the ST62E42 may be erased by exposure to Ultra Violet light.

The erasure characteristic of the ST62E42 EPROM is such that erasure begins when the memory is exposed to light with wave lengths shorter than approximately 4000Å. It should be noted that sunlight and some types of fluorescent lamps have wavelengths in the range 3000-4000Å. It is thus recommended that the window of the ST62E42 package be covered by an opaque label to prevent unintentional erasure problems when testing the application in such an environment.
The recommended erasure procedure of the ST62E42 EPROM is exposure to short wave ultraviolet light which has wavelength 2537Å. The integrated dose (i.e. UV intensity x exposure time) for erasure should be a minimum of 15 W-sec/cm^2. The erasure time with this dosage is approximately 15 to 20 minutes using an ultraviolet lamp with 12000µW/cm^2 power rating. The ST62E42 should be placed within 2.5 cm (1 inch) of the lamp tubes during erasure.

ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings

This product contains devices to protect the inputs against damage due to high static voltages, however it is advised to take normal precaution to avoid application of any voltage higher than maximum rated voltages.

For proper operation it is recommended that V_I and V_O must be higher than V_{SS} and smaller than V_{DD}. Reliability is enhanced if unused inputs are connected to an appropriated logic voltage level (V_{DD} or V_{SS}).

Power Considerations. The average chip-junction temperature, Tj, in Celsius can be obtained from:

$Tj = T_A + PD \times RthJA$

Where : T_A = Ambient Temperature.

RthJA = Package thermal resistance (junction-to ambient).

PD = Pint + Pport.

Pint = $I_{DD} \times V_{DD}$ (chip internal power).

Pport = Port power dissipation (determinated by the user).

Symbol	Parameter	Value	Unit
V_{DD}	Supply Voltage	-0.3 to 7.0	V
V_{LCD}	Display Voltage	-0.3 to 11.0	V
V_I	Input Voltage	V_{SS} - 0.3 to V_{DD} + 0.3	V
V_O	Output Voltage	V_{SS} - 0.3 to V_{DD} + 0.3	V
I_O	Current Drain per Pin Excluding V_{DD} & V_{SS}	± 10	mA
IV_{DD}	Total Current into V_{DD} (source)	50	mA
IV_{SS}	Total Current out of V_{SS} (sink)	50	mA
Tj	Junction Temperature	150	°C
T_{STG}	Storage Temperature	-60 to 150	°C

THERMAL CHARACTERISTIC

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
RthJA	Thermal Resistance	PQFP64 CQFP64-W		70 70		°C/W

RECOMMENDED OPERATING CONDITIONS

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
T_A	Operating Temperature	1 Suffix Version 6 Suffix Version	0 -40		70 85	°C
V_{DD}	Operating Supply Voltage		3		6	V
V_{PP}	EPROM Prog. Voltage		12	12.5	13	V
V_{LCD}	Display Voltage		3		10	V
V_{RM}	RAM Retention Voltage		2			V

Note: Refer to ordering information at end of the datasheet.

RECOMMENDED OPERATING CONDITIONS (Continued)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
f_{OSC}	Oscillator Frequency (1)(4)	$V_{DD} \geq 4.5V$ $V_{DD} \geq 3V$	0.01 0.01		8.388 2	MHz
I_{INJ+}	Pin Injection Current (positive) Digital Input (2) Analog Input (3)	V_{DD} = 4.5 to 5.5V			+5	mA
I_{INJ-}	Pin Injection Current (negative) Digital Input (2) Analog Input	V_{DD} = 4.5 to 5.5V			-5	mA

Notes :

1. An oscillator frequency above 1MHz is recommanded for reliable A/D results.
2. A current of ± 5mA can be forced on each pin of the digital section without affecting the functional behaviour of the device. For a positive current injected into one pin, a part of this current (~ 10%) can be expected to flow from the neighbouring pins. A current of -5mA can be forced on one input of the analog section at a time (or -2.5mA for all inputs at a time) without affecting the conversion.
3. If a total current of +1mA is flowing into the single analog channel or if the total current flowing into all the analog inputs is of 1mA, all the conversion is resulting shifted of +1LSB. If a total positive current of +5mA is flowing into the single analog channel or if the total current flowing into all the analog inputs is of 5mA, all the conversion is resulting shifted of +2LSB.
4. Operation below 0.01 MHz is possible but requires increased supply current.

EEPROM INFORMATION

The ST62xx EEPROM single poly process has been specially developed to achieve 300.000 Write/Erase cycles and a 10 years data retention.

DC ELECTRICAL CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
V_{IL}	Input Low Level Voltage	RESET, NMI, TIMER, WDON Pin			$0.3V_{DD}$	V
V_{IH}	Input High Level Voltage	TIMER	$0.80V_{DD}$			V
		RESET, NMI, WDON Pin	$0.70V_{DD}$			V
I_{IL} I_{IH}	Input Leakage Current	RESET Pin V_{DD} =5V $V_{IN} = V_{DD}$ (1) $V_{IN} = V_{DD}$ (2) $V_{IN} = V_{SS}$ (5)			10 1 50	µA mA µA
V_{OL}	Low Level Output Voltage	TIMER, I_{OL} = 5.0mA			$0.2V_{DD}$	V
V_{OH}	High Level Output Voltage	TIMER, I_{OL} = -5.0mA	$0.65V_{DD}$			V

Notes on next page

DC ELECTRICAL CHARACTERISTICS (Continued)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
R_{PU}	Pull-up Resistor	V_{IN}=0V V_{DD}=5V WDON - NMI	40	100	200	kΩ
		RESET	200	300	500	kΩ
I_{IL} I_{IH}	Input Leakage Current	TIMER $V_{IN} = V_{DD}$ or V_{SS}		0.1	1.0	µA
I_{IL} I_{IH}	Input Leakage Current	NMI VDD = 5V $V_{IN} = V_{SS}$ (5) $V_{IN} = V_{DD}$			 100 1.0	µA
I_{IL} I_{IH}	Input Leakage Current	WDON VDD = 5V $V_{IN} = V_{SS}$ (5) $V_{IN} = V_{DD}$			 100 1.0	µA
I_{DD}	Supply Current RUN Mode	f_{OSC} = 8MHz, I_{LOAD} = 0mA V_{DD} = 5.5V		4	7	mA
	Supply Current WAIT Mode (4)	f_{OSC} = 8MHz, I_{LOAD} = 0mA V_{DD} = 5.0V		1	3	mA
	Supply Current RESET Mode	f_{OSC} = 8MHz, $V_{RESET} = V_{SS}$		1	7	mA
	Supply Current STOP Mode (3)	I_{LOAD} = 0mA V_{DD} = 5.5V		1	10	µA

Notes :

1. No Watchdog Reset activated.
2. Reset generated by Watchdog.
3. When the watchdog function is actvated the STOP instruction is deactivated. WAIT instruction is automatically executed.
4. All on-chip peripherals in OFF state
5. Pull-up resistor

AC ELECTRICAL CHARACTERISTICS

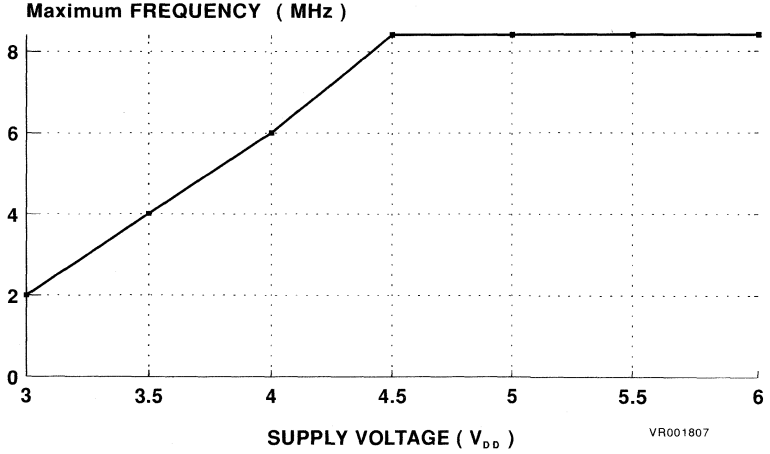
(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
f_{OSC}	Oscillator Frequency (2)	$V_{DD} \geq 4.5V$ $V_{DD} \geq 3V$	0.01		8.388 2	MHz
t_{SU}	Oscillator Start-up Time	$C_{L1} = C_{L2}$ = 22pF - crystal		5	10	ms
t_{SR}	Supply Rise Time	10% to 90%	0.01		100	
t_{REC}	Supply Recovery Time (1)		100			
T_W	Minimum Pulse Width	NMI Pin V_{DD} = 5V	100			ns
		RESET Pin	100			ns
T_{WEE}	EEPROM Write Time	T_A = 25°C One Byte T_A = 85°C One Byte		5 15	10 25	ms
Endurance	EEPROM WRITE/ERASE Cycles	Q_A L_{OT} Acceptance Criteria	300.000	> 1 million		cycles
Retention	EEPROM Data Retention	T_A = 55°C	10			years
C_{IN}	Input Capacitance	All Inputs Pins			10	pF

Notes:

1. Period for which V_{DD} has to be connected or at 0V to allow internal Reset function at next power-up.
2. Operation below 0.01 MHz is possible but requires increased supply current.

Maximum Operating FREQUENCY (Fmax) Versus SUPPLY VOLTAGE (V_{DD})



I/O PORTS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
C_{OUT}	Output Capacitance	All Outputs Pins			10	pF
V_{IL}	Input Low Level Voltage	I/O Pins			$0.3V_{DD}$	V
V_{IH}	Input High Level Voltage	I/O Pins	$0.7V_{DD}$			V
V_{OL}	Low Level Output Voltage	I/O Pins, I_O = 10µA (sink)			0.1	V
		I/O Pins, $I_{OL} = V_{DD}$x1mA V_{DD} = 4.5 to 6V			$0.16xV_{DD}$	V
		I/O Pins, I_{OL} = 1.6mA V_{DD} = 3V			0.4	V
	Low Level Output Voltage, PB4-PB7 Only	I/O Pins, $I_{OL} = V_{DD}$x2mA V_{DD} = 4.5 to 6V			$0.16xV_{DD}$	V
		I/O Pins, I_{OL} = 3.2mA V_{DD} = 3V			0.4	V
		I/O Pins, $I_{OL} = V_{DD}$x4mA V_{DD} = 4.5 to 6V			$0.26xV_{DD}$	V
		I/O Pins, I_{OL} = 6.4mA V_{DD} = 3V			0.8	V
V_{OH}	High Level Output Voltage	I/O Pins, I_O = -10µA (source)	V_{DD}-0.1			V
		I/O Pins, $I_{OL} = -V_{DD}$x1mA V_{DD} = 5.0V	$0.6xV_{DD}$			V
I_{IL} I_{IH}	Input Leakage Current	I/O Pins,(1)		0.1	1.0	µA
R_{PU}	Pull-up Resistor	I/O Pins V_{IN} = 0V, V_{DD} = 5.0V	40	100	200	kΩ

Note 1. Pull-up resistor off

SPI ELECTRICAL CHARACTERISTICS

(T_A = -40 to +85°C unless otherwhise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
F_{CL}	Clock Frequency	applied on PB5/SCL			500	kHz
t_{SU}	Set-up Time	applied on PB6/Sin		50		ns
t_h	Hold Time	applied on PB6/Sin		100		ns

A/D CONVERTER CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
Res	Resolution (3)			8		Bit
A_{TOT}	Total Accuracy (3)	f_{OSC} > 1.2 MHz f_{OSC} > 32kHz			± 2 ±4	LSB
t_C (1)	Conversion Time	f_{OSC} = 8MHz		70		µs
V_{AN}	Conversion Range		V_{SS}		V_{DD}	V
ZIR	Zero Input Reading	Conversion result when $V_{IN} = V_{SS}$	00			Hex
FSR	Full Scale Reading	Conversion result when $V_{IN} = V_{DD}$			FF	Hex
AD_I	Analog Input Current During Conversion	V_{DD}= 4.5V			1.0	µA
AC_{IN} (2)	Analog Input Capacitance			2	5	pF
ASI	Analog Source Impedance				30	kΩ
SSI	Analog Reference Supply Impedence				2	kΩ

Notes:

1. With oscillator frequencies less than 1MHz, the A/D Converter accuracy is decreased.
2. Excluding Pad Capacitance
3. Noise at $V_{DD}, V_{SS} \leq 10mV$

TIMER CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
t_{RES}	Resolution		$\frac{12}{f_{OSC}}$			second
f_{IN}	Input Frequency on TIMER Pin				$\frac{f_{OSC}}{8}$	MHz
t_W	Pulse Width at TIMER Pin	$V_{DD} \geq 3V$ $V_{DD} \geq 4.5V$	1 125			µs ns

LCD ELECTRICAL CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

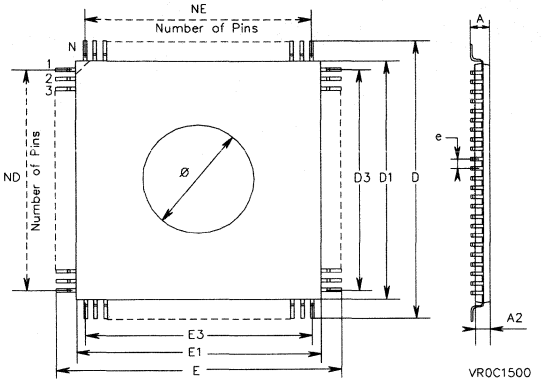
Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
f_{FR}	Frame Frequency	1/4 Duty f_{OSC} = 1, 2, 4, 8MHz	16		128	Hz
V_{OS}	DC Offset Voltage(1)	$V_{LCD} = V_{DD}$, no load			50	mV
V_{OH}	COM High Level, Output Voltage	I = 100µA V_{LCD} = 5V	4.5V			V
V_{OL}	COM Low Level, Output Voltage	I = 100µA V_{LCD} = 5V			0.5V	V
V_{OH}	SEG High Level, Output Voltage	I = 50µA V_{LCD} = 5V	4.5V			V
V_{OL}	SEG Low Level, Output Voltage	I = 50µA V_{LCD} = 5V			0.5V	V
V_{LCD}	Display Voltage	Note 2	3		10	V

Notes :

1. The DC offset voltage refers to all segment and common outputs. It is the difference between the measured voltage value and nominal value for every voltage level. Ri of voltage meter must be greater than or equal to 100MΩ.
2. An external resistances network is required when $V_{LCD} \leq 4.5V$.

PACKAGE MECHANICAL DATA

Figure 3. ST62E42 64 Pin Ceramic Quad Flat Package with Window



Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A		3.55			0.14	
A2		3.40			0.133	
D		17.90			0.705	
D1		14.00			0.551	
D3		12.00			0.472	
E		17.90			0.705	
E1		14.00			0.551	
E3		12.00			0.472	
Ø		7.62			0.3	
e		0.80			0.032	
Number of Pins						
ND	16					
NE	16					
N	64					

ORDERING INFORMATION TABLE

Sales Types	Memory Type	Temperature Range	Package
ST62E42G1	8K EPROM	0 to + 70°C	CQFP64-W
ST62T42Q6	8K EPROM	-40 to + 85°C	PQFP64

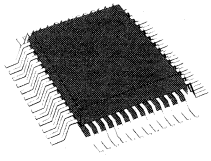
SGS-THOMSON MICROELECTRONICS

ST6245

8-BIT HCMOS MCU WITH LCD DRIVER, EEPROM AND A/D CONVERTER

PRELIMINARY DATA

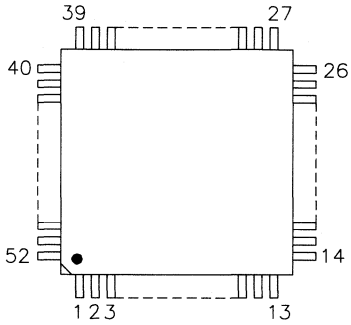
- 3 to 6V supply operating range
- 8.4MHz Maximum Clock Frequency
- -40 to +85°C Operating Temperature Range
- Run, Wait & Stop Modes
- 5 different interrupt vectors
- Look-up table capability in ROM
- User ROM: 3884 bytes
  Data RAM: 128 bytes
  LCD RAM: 12 bytes
  EEPROM: 64 bytes
- PQFP52 package
- 11 fully software programmable I/O as:
  - Input with/without pull-up resistor
  - Input with interrupt generation
  - Open-Drain or Push-pull outputs
  - Analog Inputs (7 pins)
- 4 I/O lines can sink up to 20mA for direct LED or TRIAC driving, and have SPI alternate functions
- Two 8-bit counters with 7-bit programmable prescalers (Timers 1 and 2)
- Software activated digital watchdog
- 8-bit A/D converter with up to 7 analog inputs
- 8-bit synchronous Serial Peripheral Interface (SPI)
- LCD driver with 24 segment outputs, 4 back-plane outputs and selectable duty cycle for up to 96 LCD segments direct driving
- 32kHz oscillator for stand-by LCD operation
- One external not maskable interrupt
- 9 powerful addressing modes
- The accumulator, the X, Y, V & W registers, the port and peripherals data & control registers are addressed in the data space as RAM locations.
- The ST62E45 is the EPROM version, ST62T45 is the OTP version
- Development tool: ST6245-EMU connected via RS232 to an MS-DOS Personal Computer



PQFP52

(Ordering Information at the end of the datasheet)

Figure 1. 52 Pin Quad Flat Pack (QFP) Package Pinout



ST6245 Pin Description

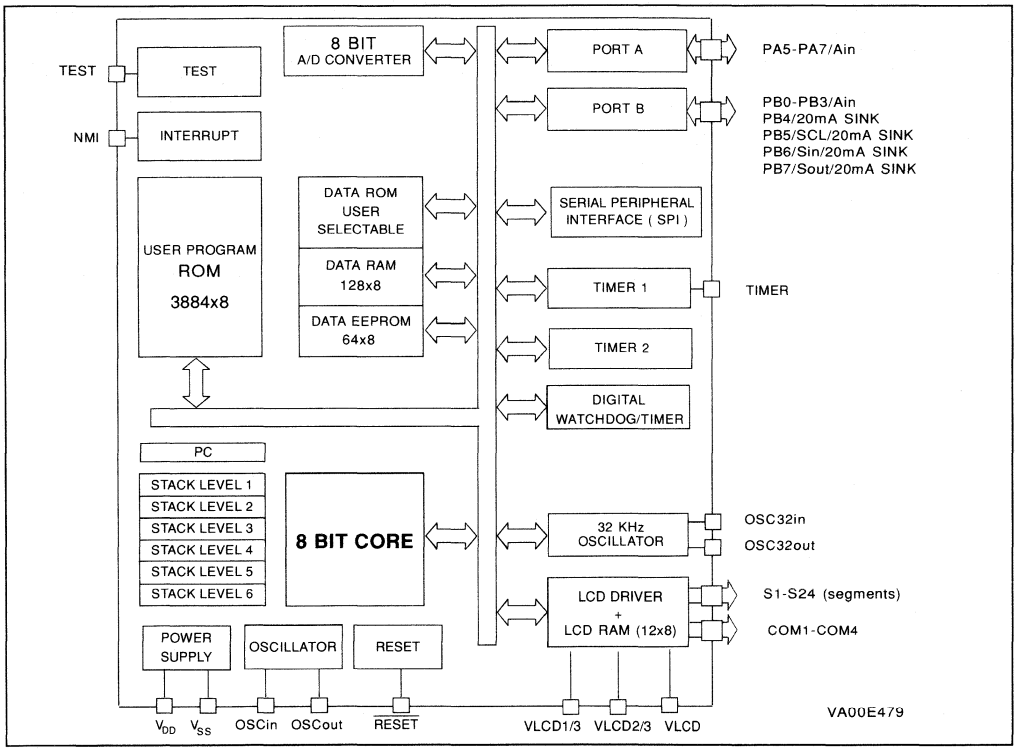
Pin number	Pin name	Pin number	Pin name	Pin number	Pin name	Pin number	Pin name
1	COM4	14	$\overline{\text{RESET}}$	27	OSC32out	40	S12
2	COM3	15	OSCout	28	OSC32in	41	S13
3	COM2	16	OSCin	29	S1	42	S14
4	COM1	17	NMI	30	S2	43	S15
5	VLCD1/3	18	TIMER	31	S3	44	S16
6	VLCD2/3	19	PB7/Sout (1)	32	S4	45	S17
7	VLCD	20	PB6/Sin (1)	33	S5	46	S18
8	PA7/Ain	21	PB5/SCL (1)	34	S6	47	S19
9	PA6/Ain	22	PB4 (1)	35	S7	48	S20
10	PA5/Ain	23	PB3/Ain	36	S8	49	S21
11	TEST	24	PB2/Ain	37	S9	50	S22
12	V_{DD}	25	PB1/Ain	38	S10	51	S23
13	V_{SS}	26	PB0/Ain	39	S11	52	S24

Note 1: 20mA SINK

GENERAL DESCRIPTION

The ST6245 microcontroller is a member of the 8-bit HCMOS ST62xx family, a series of devices oriented to low-medium complexity applications. All ST62xx members are based on a building block approach: a common core is associated with a combination of on-chip peripherals (macrocells). The macrocells of the ST6245 are: a high performance LCD controller/driver with 24 segment outputs and 4 backplanes able to drive up to 96 segments, two Timer peripherals each including an 8-bit counter with a 7-bit software programmable prescaler (Timer), the digital watchdog (DWD), an 8-bit A/D Converter with up to 7 analog inputs and an 8-bit synchronous Serial Peripheral Interface (SPI). In addition these devices offer 64 bytes of EEPROM for storage of non volatile data. Thanks to these peripherals the ST6245 is well suited for general purpose, automotive, security, appliance and industrial applications. The ST62E45 EPROM version is available for prototypes and low-volume production, an OTP version is also available (see separate datasheet).

Figure 2. ST6245 Block Diagram



Note: Ain = Analog Input

PIN DESCRIPTION

V_{DD} and V_{SS}. Power is supplied to the MCU using these two pins. V_{DD} is power and V_{SS} is the ground connection.

OSCin and OSCout. These pins are internally connected with the on-chip oscillator circuit. A quartz crystal or a ceramic resonator can be connected between these two pins in order to allow the correct operation of the MCU with various stability/cost trade-offs. OSCin is the input pin, OSCout is the output pin. An external clock signal can be applied to OSCin.

$\overline{\text{RESET}}$. The active low $\overline{\text{RESET}}$ pin is used to restart the microcontroller at the beginning of its program. The $\overline{\text{RESET}}$ pin is provided with an on-chip pull-up resistor and schmitt trigger input characteristics.

TEST. The TEST must be held at V_{SS} for normal operation (an internal pull-down resistor selects normal operating mode if TEST pin is not connected).

NMI. The NMI pin provides the capability for asynchronous applying an external top priority interrupt to the MCU.This pin is falling edge sensitive. The NMI pin is provided with an on-chip pull-up resistor and schmitt trigger input characteristics.

TIMER. This is the TIMER 1 I/O pin. In input mode it is connected to the prescaler and acts as external timer clock or as control gate for the internal timer clock. In the output mode the TIMER pin outputs the data bit when a time-out occurs.

PA5-PA7. These 3 lines are organized as one I/O port (A). Each line may be configured under software control as an input with or without pull-up resistor, interrupt generating input with pull-up resistor, open-drain or push-pull output or as analog input for the A/D converter. Port A has a 5mA drive capability in output mode.

PB0-PB3,PB4-PB7. These 8 lines are organized as one I/O port (B). Each line may be configured under software control as an input with or without internal pull-up resistor, interrupt generating input with pull-up resistor, open-drain or push-pull output. PB0-PB3 can be programmed as analog inputs for the A/D converter while PB4-PB7 can also sink 20mA for direct LED driving. PB5-PB7 can also be used as respectively Clock, Data in and Data out pins for the on-chip SPI to carry the synchronous serial I/O signals.

COM1-COM4. These four pins are the LCD peripheral common outputs. They are the outputs of the on-chip backplane voltage generator which is used for multiplexing the 24 LCD lines allowing up to 96 segments to be driven.

S1-S24. These pins are the 24 LCD peripheral driver outputs of ST6245. Segments S1-S3 are not connected to any pin.

VLCD. Display voltage supply. It determines the high voltage level on COM1-COM4 and S1-S24 pins.

VLCD1/3, VLCD2/3. Display supply voltage inputs for determining the display voltage levels on COM1-COM4 and S1-S24 pins during multiplex operation.

OSC32in and OSC32out. These pins are internally connected with the on-chip 32kHz oscillator circuit. A 32.768kHz quartz crystal can be connected between these two pins if it is necessary to provide the LCD stand-by clock and real time interrupt. OSC32in is the input pin, OSC32out is the output pin.

ST62xx CORE

The core of the ST62xx Family is implemented independently from the I/O or memory configuration. Consequently, it can be treated as an independent central processor communicating with I/O and memory via internal addresses, data, and control busses. The in-core communication is arranged as shown in Figure 3; the controller being externally linked to both the reset and the oscillator, while the core is linked to the dedicated on-chip macrocells peripherals via the serial data bus and indirectly for interrupt purposes through the control registers.

Registers

The ST62xx Family core has six registers and three pairs of flags available to the programmer. They are shown in Figure 4 and are explained in the following paragraphs.

Accumulator (A). The accumulator is an 8-bit general purpose register used in all arithmetic calculations, logical operations, and data manipulations. The accumulator is addressed in the data space as RAM location at address FFh. Accordingly, the ST62xx instruction set can use the accumulator as any other register of the data space.

Figure 4. ST62xx Core Programming Model

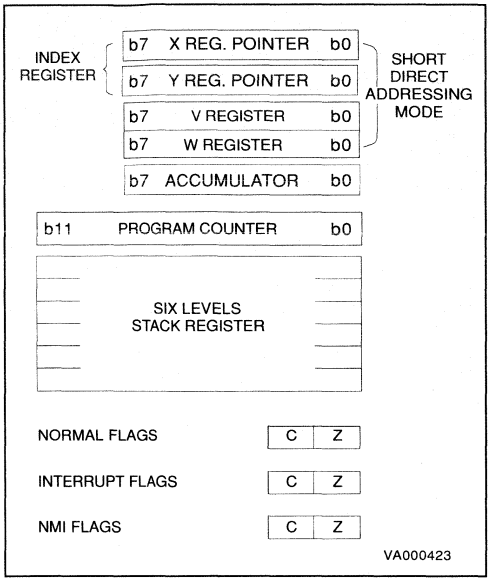
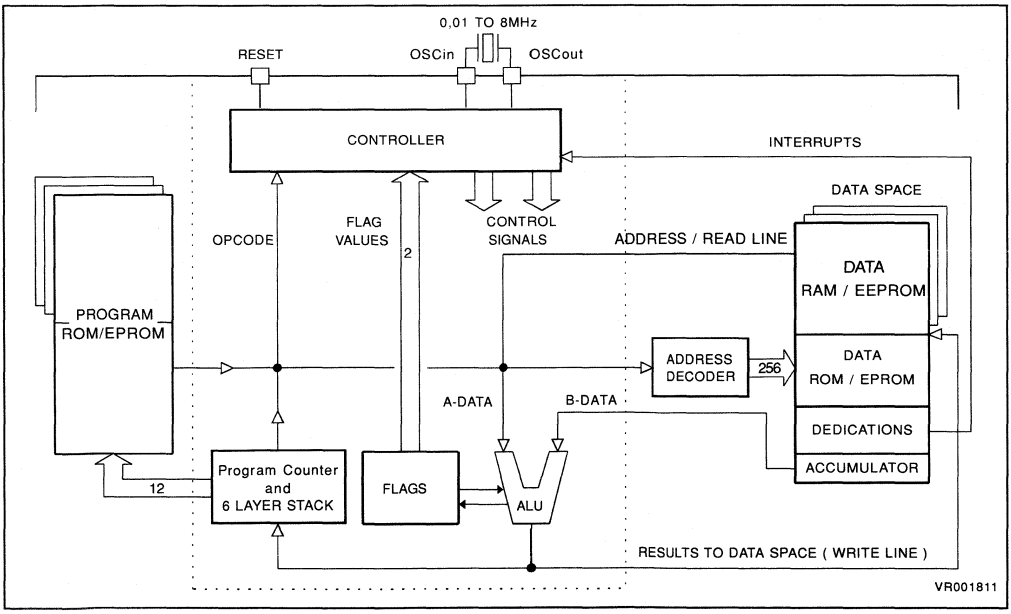


Figure 3. ST62xx Core Block Diagram



ST62xx CORE (Continued)

Indirect Registers (X, Y). These two indirect registers are used as pointers to memory locations in the data space. They are used in the register-indirect addressing mode. These registers can be addressed in the data space as RAM locations at addresses 80h (X) and 81h (Y). They can also be accessed with the direct, short direct, or bit direct addressing modes. Accordingly, the ST62xx instruction set can use the indirect registers as any other register of the data space.

Short Direct Registers (V, W). These two registers are used to save one byte in short direct addressing mode. These registers can be addressed in the data space as RAM locations at addresses 82h (V) and 83h (W). They can also be accessed with the direct and bit direct addressing modes. Accordingly, the ST62xx instruction set can use the short direct registers as any other register of the data space.

Program Counter (PC)

The program counter is a 12-bit register that contains the address of the next ROM location to be processed by the core. This ROM location may be an opcode, an operand, or an address of operand. The 12-bit length allows the direct addressing of 4096 bytes in the program space.

The PC value is incremented after it is read from the address of the current instruction. To execute relative jumps the PC and the offset are shifted through the ALU, where they will be added, and the result is shifted back into the PC. The program counter can be changed in the following ways:

- JP (Jump) instruction . . . PC=Jump address
- CALL instruction . . . . PC=Call address
- Relative Branch instructions . . . . . PC=PC ± offset
- Interrupt . . . . . PC=Interrupt vector
- Reset . . . . . PC=Reset vector
- RET & RETI instructions . PC=Pop (stack)
- Normal instruction . . . . PC=PC+1

Flags (C, Z)

The ST62xx core includes three pairs of flags that correspond to 3 different modes: normal mode, interrupt mode and Non-Maskable Interrupt-Mode. Each pair consists of a CARRY flag and a ZERO flag. One pair (CN, ZN) is used during normal operation, one pair is used during the interrupt mode (CI, ZI) and one is used during the not-maskable interrupt mode (CNMI, ZNMI).

The ST62xx core uses the pair of flags that correspond to the actual mode: as soon as an interrupt (resp. a Non-Maskable-Interrupt) is generated, the ST62xx core uses the interrupt flags (resp. the NMI flags) instead of the normal flags. When the RETI instruction is executed, the normal flags (resp. the interrupt flags) are restored if the MCU was in the normal mode (resp. in the interrupt mode) before the interrupt. It should be observed that each flag set can only be addressed in its own mode (Not-maskable interrupt, normal interrupt or main mode). The flags are not cleared during the context switching and so remain in the state they were at the exit of the last mode switch.

The Carry flag is set when a carry or a borrow occurs during arithmetic operations, otherwise it is cleared. The Carry flag is also set to the value of the bit tested in a bit test instruction, and participates in the rotate left instruction.

The Zero flag is set if the result of the last arithmetic or logical operation was equal to zero, otherwise it is cleared.

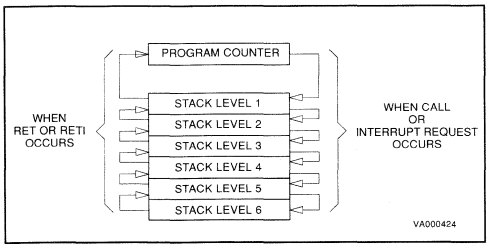
The switching between the three sets of Flags is automatically performed when an NMI, an interrupt or a RETI instruction occurs. As the NMI mode is automatically selected after the reset of the MCU, the ST62xx core uses at first the NMI flags.

ST62xx CORE (Continued)

Stack

The ST62xx core includes a true LIFO hardware stack that eliminates the need for a stack pointer. The stack consists of six separate 12-bit RAM locations that do not belong to the data space RAM area. When a subroutine call (or interrupt request) occurs, the contents of each level is shifted into the next level while the content of the PC is shifted into the first level (the value of the sixth level will be lost). When a subroutine or interrupt return occurs (RET or RETI instructions), the first level register is shifted back into the PC and the value of each level is popped back into the previous level. These two operating modes are described in Figure 5. Since the accumulator, as all other data space registers, is not stored in the stack, the handling of these registers should be performed inside the subroutine. The stack pointer will remain in its deepest position if more than 6 calls or interrupts are executed, so that the last return address will be lost. It will also remain in its highest position if the stack is empty and a RET or RETI is executed. In this case the next instruction will be executed.

Figure 5. Stack Operation



MEMORY SPACES

The MCU operates in three different memory spaces: program space, data space, and stack space. A description of these spaces is shown in the following figures.

Program Space

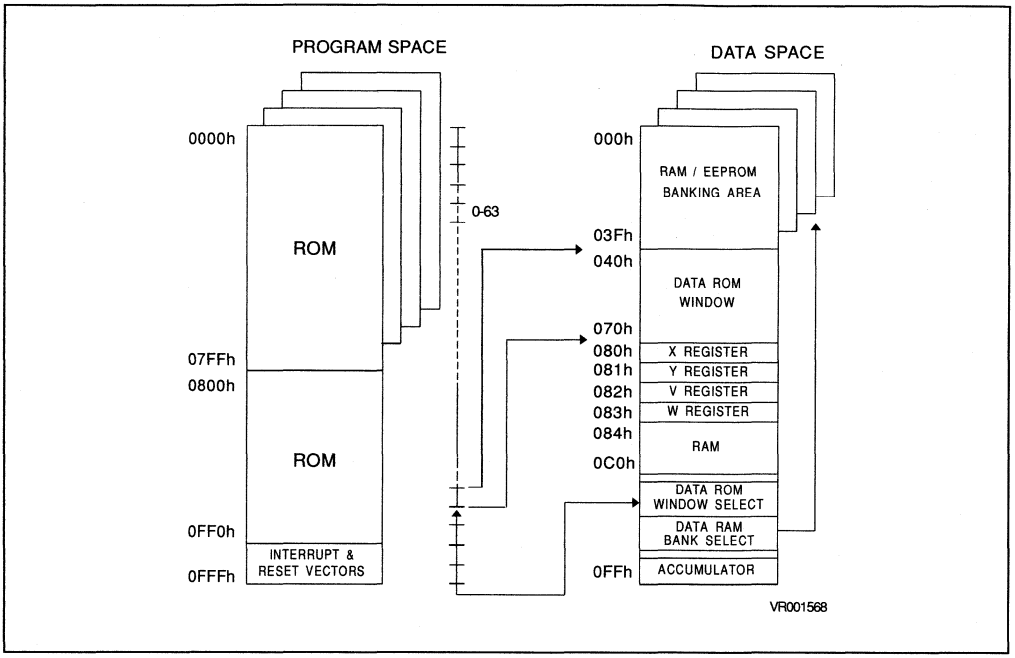
The program space is physically implemented in the ROM memory and includes all the instructions that are to be executed, as well as the data required for the immediate addressing mode instructions, the reserved test area and the user vectors. It is addressed by the 12-bit Program Counter register (PC register) and so the ST62xx core can directly address up to 4K bytes of Program Space.

Table 1. ST6245 Program ROM Memory Map

Device Address	Description
0000h-007Fh	Reserved
0080h-0F9Fh	User ROM
0FA0h-0FEFh	Reserved
0FF0h-0FF7h	Interrupt Vectors
0FF8h-0FFBh	Reserved
0FFCh-0FFDh	NMI Vector
0FFEh-0FFFh	Reset Vector

MEMORY SPACES (Continued)

Figure 6. ST62xx Memory Addressing Description Diagram



MEMORY SPACES (Continued)

Data Space

The instruction set of the ST62xx core operates on a specific space, named Data Space, that contains all the data necessary for the processing of the program. The Data Space allows the addressing of RAM memory, ST62xx core/peripheral registers, and read-only data such as constants and look-up tables.

Data ROM. All the read-only data is physically implemented in the ROM memory in which the Program Space is also implemented. The ROM memory contains consequently the program to be executed, the constants and the look-up tables needed for the program.

The locations of Data Space in which the different constants and look-up tables are addressed by the ST62xx core can be considered as being a 64-byte window through which it is possible to access to the read-only data stored in the ROM memory .

Data RAM/EEPROM. The ST6245 offers 128 bytes of data RAM memory and 64 bytes of EEPROM. 64 bytes of RAM are directly addressed in data space in the range 080h-0BFh (static space). The additional RAM are addressed using the banks of 64 bytes located between addresses 00h and 3Fh.

Additionally RAM are available in the LCD data map from E0h to F7h and are not banked.

Stack Space

The stack space consists of six 12 bit registers that are used for stacking subroutine and interrupt return addresses plus the current program counter register.

Figure 7. ST6240 Data Memory Space

DATA RAM/EEPROM BANK AREA	000h 03Fh
DATA ROM WINDOW AREA	040h 07Fh
X REGISTER	080h
Y REGISTER	081h
V REGISTER	082h
W REGISTER	083h
DATA RAM 60 BYTES	084h 0BFh
PORT A DATA REGISTER	0C0h
PORT B DATA REGISTER	0C1h
SPI INT. DISABLE REGISTER	0C2h*
RESERVED	0C3h
PORT A DIRECTION REGISTER	0C4h
PORT B DIRECTION REGISTER	0C5h
RESERVED	0C6h
RESERVED	0C7h
INTERRUPT OPTION REGISTER	0C8h*
DATA ROM WINDOW REGISTER	0C9h*
RESERVED	0CAh*
DATA RAM/EEPROM BANK REGISTER	0CBh*
PORT A OPTION REGISTER	0CCh
RESERVED	0CDh
PORT B OPTION REGISTER	0CEh
RESERVED	0CFh
A/D DATA REGISTER	0D0h
A/D CONTROL REGISTER	0D1h
TIMER 1 PRESCALER REGISTER	0D2h
TIMER 1 COUNTER REGISTER	0D3h
TIMER 1 STATUS/CONT REGISTER	0D4h
TIMER 2 PRESCALER REGISTER	0D5h
TIMER 2 COUNTER REGISTER	0D6h
TIMER 2 STATUS/CONT REGISTER	0D7h
WATCHDOG REGISTER	0D8h
RESERVED	0D9h
RESERVED	0DAh
32kHz OSC. CONTROL REGISTER	0DBh
LCD MODE CONTROL REGISTER	0DCh
SPI DATA REGISTER	0DDh
RESERVED	0DEh
EEPROM CONTROL REGISTER	0DFh
LCD RAM	0E0h 0F7h
DATA RAM 7 BYTES	0F8h 0FEh
ACCUMULATOR	0FFh

* WRITE ONLY REGISTER

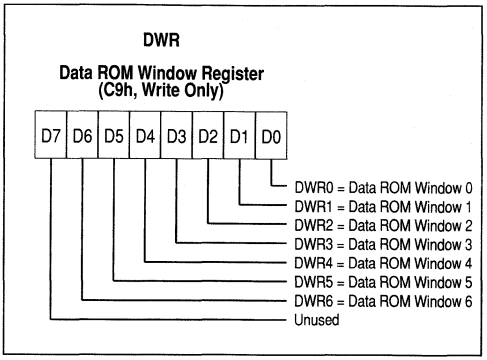
MEMORY SPACES (Continued)

Data Window register (DWR)

The Data ROM window is located from address 040h to address 7Fh in the Data space. It allows the direct reading of 64 consecutive bytes located anywhere in the ROM memory between the addresses 0000h and 1FFFh. All the bytes of the ROM memory can be used to store either instructions or read-only data. Indeed, the window can be moved by step of 64 bytes along the ROM memory in writing the appropriate code in the Write-only Data Window register (DWR register, location C9h).

The DWR register can be addressed like a RAM location in the Data Space at the address C9h, nevertheless it is a write only register that cannot be accessed with single-bit operations. This register is used to move the 64-byte read-only data window (from the 40h address to 7Fh address of the Data Space) up and down the ROM memory of the MCU in steps of 64 bytes. The effective address of the byte to be read as a data in the ROM memory is obtained by the concatenation of the 6 least significant bits of the register address given in the instruction (as least significant bits) and the content of the DWR register (as most significant bits, see Figure 8). So when addressing location 40h of dataspace, and 0 is loaded in the DWR register, the phisycal addressed location in ROM is 00h. The DWR register is not cleared at reset, therefore it must be written to before the first access to the Data ROM window area.

Figure 9. Data ROM Window Register



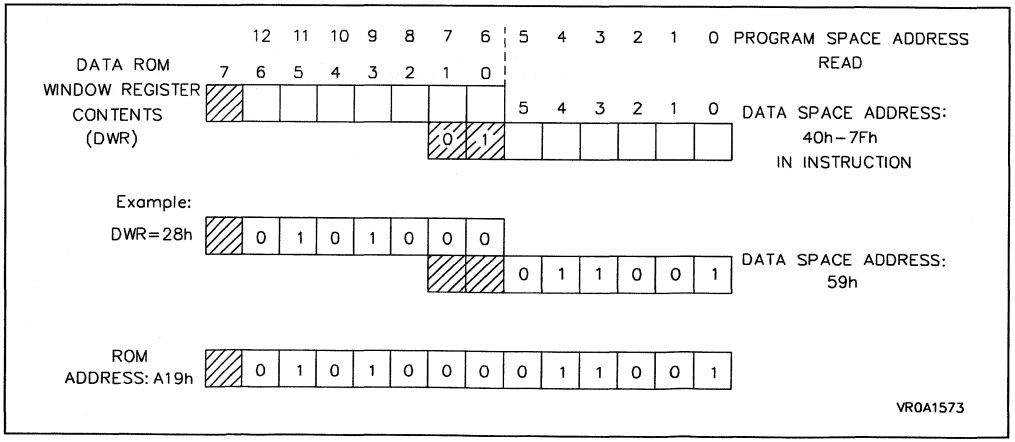
D7. This bit is not used.

DWR6-DWR0. These are the Data ROM Window bits that correspond to the upper bits of the data ROM space.

This register is undefined on reset. Neither read nor single bit instructions may be used to address this register.

Note: Care is required when handling the DWR register as it is write only. For this reason, it is not allowed to change the DWR contents while executing interrupt service routine, as the service routine cannot save and then restore its previous content. If it is impossible to avoid the writing of this register in the interrupt service routine, an image of this register must be saved in a RAM location, and each time the program writes to the DWR it must write also to the image register. The image register must be written first, so if an interrupt occurs between the two instructions the DWR is not affected.

Figure 8. Data ROM Window Memory Addressing



MEMORY SPACES (Continued)

Data RAM/EEPROM Bank Register (DRBR)

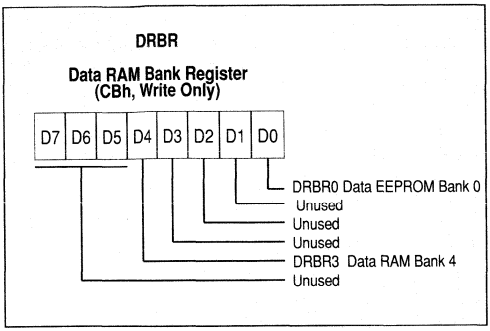
The selection of the bank is made by programming the Data RAM Bank Switch register (DRBR register) located at address CBh of the Data Space. The number of the selected bank is equal to the bit content of the DRBR register. In this way each bank of RAM or EEPROM can be selected 64 bytes at a time. No more than one bank should be set at a time.

The DRBR register can be addressed like a RAM location in the Data Space at the address CBh; nevertheless it is a write only register that cannot be accessed with single-bit operations. This register is used to select the desired 64-byte RAM/EEPROM bank of the Data Space. The number of the bank has to be loaded in the DRBR register and the instruction has to point to the selected location as if it was in bank 0 (from 00h address to 3Fh address). This register is not cleared during the MCU initialization, therefore it must be written before the first access to the Data Space bank region. Refer to the Data Space description for additional information. The DRBR register is not modified when a interrupt or a subroutine occurs.

Table 2. Data RAM Bank Register Set-up

DRBR Value	Selection
01h	EEPROM
10h	RAM

Figure 10. Data RAM Bank Register



The following Table 2 summarizes how to set the data RAM bank register in order to select the various banks or pages.

D7-D5. These bits are not used.

DRBR3. This bit, when set, will select RAM page.

D3-D1. These bits are not used.

DRBR0. This bit, when set, will select EEPROM page.

This register is undefined on reset. Neither read nor single bit instructions may be used to address this register.

Notes:

Care is required when handling the DRBR register as it is write only. For this reason, it is not allowed to change the DRBR contents while executing interrupt service routine, as the service routine cannot save and then restore its previous content. If it is impossible to avoid the writing of this register in interrupt service routine, an image of this register must be saved in a RAM location, and each time the program writes to DRBR it must write also to the image register. The image register must be written first, so if an interrupt occurs between the two instructions the DRBR is not affected.

In DRBR Register, *only 1 bit must be set.* Otherwise two or more pages are enabled in parallel, producing errors.

MEMORY SPACES (Continued)

EEPROM Description

The data space of ST62xx family from 00h to 3Fh is paged as described in Table 3. The ST6245 has 64 bytes of EEPROM located in one page of 64 bytes (page 0).

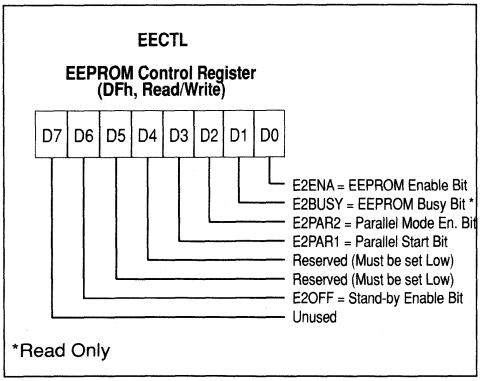
The EEPROM pages are physically organized in 32 byte modules (2 modules per page) and does not require dedicated instructions to be accessed in reading or writing. The EEPROM is controlled by the EEPROM Control Register (EECTL = DFh). In order to enable access to the EEPROM, bit 6 of this register must be cleared otherwise any access to the EEPROM will be meaningless.

Any EEPROM location can be read just like any other data location, also in terms of access time.

When writing to an EEPROM, the EEPROM is not accessible by the ST62xx. A busy flag can be read to identify the EEPROM status before attempting any access. Writing the EEPROM can work in two modes: Byte Mode (BMODE) and Parallel Mode (PMODE). BMODE is the normal way to use the EEPROM and consists in accessing one byte at a time. PMODE consists in accessing 8 bytes per time.

Readout of the EEPROM is made at the same speed as RAM acces.

Figure 11. EEPROM Control Register



D7. Not Used

E2OFF. *WRITE ONLY.* If this bit is set the EEPROM is disabled (any access will be meaningless) and the power consumption of the EEPROM is reduced to the lowest values.

D5, D4. Reserved, must be set to zero.

E2PAR1. *WRITE ONLY.* Once in Parallel Mode, as soon as the user software sets the E2PAR1 bit the parallel writing of the 8 adjacent registers will start. It is internally reset at the end of the programming procedure. Note that less than 8 bytes can be written; after parallel programming the undefined bytes will be unaffected

E2PAR2. *WRITE ONLY.* This bit must be set by the user program in order to perform parallel programming (more than one byte at a time). If E2PAR2 is set and the parallel start bit (E2PAR1) is low, up to 8 adjacent bytes can be written at maximum speed, the contents being stored in volatile registers. These 8 adjacent bytes are considered as a row, whose address lines A7, A6, A5, A4, A3 are fixed while A2, A1 and A0 are the changing bits. E2PAR2 is automatically reset at the end of any parallel programming procedure. It can be reset by the user software before starting the programming procedure, leaving the EEPROM registers unchanged.

E2BUSY. *READ ONLY.* This bit will be automatically set by the EEPROM control logic when the user program modifies an EEPROM register. The user program must test it before any read or write EEPROM operation; any attempt to access the EEPROM while the busy bit is set will be aborted and the writing procedure in progress completed.

E2•E••NA. *WRITE ONLY.* This bit MUST be set to one in order to write to any EEPROM register. If the user program attempts to write to the EEPROM when E2ENA = "0", the involved registers will be unaffected and the BS will not be set.

After RESET the content of EECTL register will be 00h.

Notes:

The data to write has to be written directly at the address that it will have inside the EEPROM space. There is no buffer memory between the data•RAM and the EEPROM spaces.

When the EEPROM is busy (E2BUSY = "1") EECTL can not be accessed in write mode, it is only possible to read the status of E2BUSY. This implies that as long as the EEPROM is busy, it is not possible to change the status of the EEPROM control register. EECTL bits 4 and 5 are reserved for test purposes, and must never be set to "1".

MEMORY SPACES (Continued)

Table 3. EEPROM Parallel Write Row Structure

Byte	0	1	2	3	4	5	6	7	Dataspace addresses. Banks 0 and 1.
ROW7									38h-3Fh
ROW6									30h-37h
ROW5									28h-2Fh
ROW4									20h-27h
ROW3									18h-1Fh
ROW2									10h-1Fh
ROW1									08h-0Fh
ROW0									00h-07h

Up to 8 bytes in each row may be programmed at the same time in Parallel Write mode

Additional Notes on Parallel Mode. If the user wishes to perform parallel programming, the first action should be to set the E2PAR2 bit to one. From this time the EEPROM will be addressed in writing, the ROW address will be latched and it will be possible to change it only at the end of the programming procedure or by resetting E2PAR2 without programming the EEPROM. After the ROW address latching the ST62xx can "see" only one EEPROM row (the selected one) and any attempt to write or read other rows will produce errors. Do not read the EEPROM while E2PAR2 is set.

As soon as E2PAR2 bit is set, the 8 volatile ROW latches are cleared. From this moment the user can load data in the whole ROW or in a subset. Setting E2PAR1 will modify the EEPROM registers corresponding to the ROW latches accessed after E2PAR2. For example, if the software sets E2PAR2 and accesses the EEPROM by writing to addresses 18h, 1Ah, 1Bh and then sets E2PAR1, these three registers will be modified at the same time; the remaining bytes will be unaffected. Note that E2PAR2 is internally reset at the end of the programming procedure. This implies that the user must set E2PAR2 bit between two parallel programming procedures. Note that if the user tries to set E2PAR1 while E2PAR2 is not set there will not be any programming procedure and the E2PAR1 bit will be unaffected. Consequently E2PAR1 bit cannot be set if E2ENA is low. E2PAR1 can be affected by the user to set it, only if E2ENA and E2PAR2 bits are also set to one.

TEST MODE

For normal operation the TEST pin must be held low. An on-chip 100kΩ pull-down resistor is internally connected to the TEST pin.

INTERRUPTS

The ST62xx core can manage 4 different maskable interrupt sources, plus one non-maskable interrupt source (top priority level interrupt). Each source is associated with a particular interrupt vector that contains a Jump instruction to the related interrupt service routine. Each vector is located in the Program Space at a particular address.
When a source provides an interrupt request, and the request processing is also enabled by the ST62xx core, then the PC register is loaded with the address of the interrupt vector (i.e. of the Jump instruction). Finally, the PC is loaded with the address of the Jump instruction and the interrupt routine is processed.

The ST6245 microcontroller has eight different interrupt sources associated to different interrupt vectors as described in Table 4.

Table 4. Interrupt Vectors - Sources Relationship

Interrupt Source	Associated Vector	Vector Address
NMI Pin	Interrupt Vector #0	(FFCh-FFDh)
SPI Peripheral	Interrupt Vector #1	(FF6h-FF7h)
Port A & B Pins	Interrupt Vector #2	(FF4h-FF5h)
TIMER 1, 2 & 32kHz Oscillator	Interrupt Vector #3	(FF2h-FF3h)
ADC Peripheral	Interrupt Vector #4	(FF0h-FF1h)

Interrupt Vectors Description

The ST62xx core includes 5 different interrupt vectors in order to branch to 5 different interrupt routines in the static page of the Program Space:

- The interrupt vector associated with the non-maskable interrupt source is named interrupt vector #0. It is located at addresses FFCh,FFDh in the Program Space. On ST6245 this vector is associated with the external falling edge sensitive interrupt pin (NMI). An on-chip 100kΩ pull-up resistor is internally connected to the NMI pin.
- The interrupt vector located at the addresses FF6h, FF7h is named interrupt vector #1. It is associated with SPI peripheral and can be programmed by software to generate an interrupt request after the falling edge or low level of the eighth external clock pulse according to the code loaded in the Interrupt Option Register (IOR).
- The interrupt vector located at the addresses FF4h, FF5h is named interrupt vector #2. It is associated with Port A and B pins and can be programmed by software either in the falling edge detection mode or in the rising edge detection mode according to the code loaded in the Interrupt Option Register (IOR).
- The interrupt vector located at the addresses FF2h, FF3h is named interrupt vector #3. It is associated with Timer 1, Timer 2 and the 32kHz Oscillator peripherals. All these interrupts are "ORed" together and are connected to interrupt line #3 of the core. Discrimination among the three interrupts must be made by polling the Status/Control registers of Timer 1 (0D4h), Timer 2 (0D7h) and 32kHz oscillator (0DBh).
- The interrupt vector located at the addresses FF0h, FF1h is named interrupt vector #4. It is associated with the A/D converter peripheral.

All the on-chip peripherals (refer to their descriptions for further details) have an interrupt request flag bit (TMZ for timer, EOC for A/D, etc.), this bit is set to one when the device wants to generate an interrupt request and a mask bit (ETI for timer, EAI for A/D, etc.) that must be set to one to allow the transfer of the flag bit to the Core.

Interrupt Priority

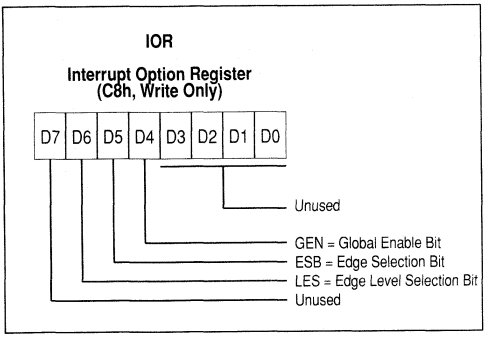
The non-maskable interrupt request has the highest priority and can interrupt any other interrupt routines at any time, nevertheless the four other interrupts cannot interrupt each other. If more than one interrupt request is pending, they are processed by the ST62xx core according to their priority level: vector #1 has the higher priority while vector #4 the lower. The priority of each interrupt source is fixed.

INTERRUPTS (Continued)

Interrupt Option Register

The Interrupt Option Register (IOR register, location C8h) is used to enable/disable the individual interrupt sources and to select the operating mode of the external interrupt inputs. This register can be addressed in the Data Space as RAM location at the address C8h, nevertheless it is a write-only register that cannot be accessed with single-bit operations. The operating modes of the external interrupt inputs associated to interrupt vectors #1 and #2 are selected through bits 5 and 6 of the IOR register.

Figure 12. Interrupt Option Register



D7. This bit is not used.

LES. Level/Edge Selection Bit. When this bit is set to one, the interrupt #1 (SPI) is low level sensitive, when cleared to zero the negative edge sensitive interrupt is selected.

ESB. Edge Selection Bit. When this bit is set to one, the interrupt #2 (Port A & B lines) is positive edge sensitive, when cleared to zero the negative edge sensitive interrupt is selected.

GEN. Global Enable Interrupt. When this bit is set to one, all the interrupts are enabled. When this bit is cleared to zero all the interrupts (including the NMI) are disabled.

This register is cleared on reset.

Table 5. Interrupt Option Register Description

GEN	SET	Enable all interrupts
	CLEARED	Disable all interrupts
ESB	SET	Rising edge mode on interrupt input #2
	CLEARED	Falling edge mode on interrupt input #2
LES	SET	Level-sensitive mode on interrupt input #1
	CLEARED	Falling edge mode on interrupt input #1
OTHERS	NOT USED	

External Interrupts Operating Modes

The NMI interrupt is associated to the NMI pin of the ST6245. The highest priority interrupt request will be generated by a falling edge applied to the NMI pin. The NMI interrupt pin signal is latched and is automatically reset by the core at the beginning of the non-maskable interrupt service routine. An on-chip pull-up resistor and a schmitt trigger is available with the NMI pin.

The two interrupt sources associated with the falling/rising edge mode of the external interrupt pins (SPI vector #1, Ports A and B vector #2,) are connected to two internal latches. Each latch is set when a falling/rising edge occurs and is cleared when the associated interrupt routine is started. So, the occurrence of an external interrupt request is stored: a second interrupt, that occurs during the processing of the first one, will be processed as soon as the first one has been finished (if there is not an higher priority interrupt request). If more than one interrupt occurs during the processing of the first one, these other interrupt requests will be lost.

The storage of the interrupt requests is not available in the level sensitive detection mode. To be taken into account, the low level must be present on the interrupt pin when the core samples the line after the execution of the instructions.

During the end of each instruction the core tests the interrupt lines and if there is an interrupt request the next instruction is not executed and the related interrupt routine is executed.

INTERRUPTS (Continued)

Interrupt Procedure. The interrupt procedure is very similar to a call procedure, indeed the user can consider the interrupt as an asynchronous call procedure. As this is an asynchronous event the user does not know about the context and the time at which it occurred. As a result the user should save all the data space registers which will be used inside the interrupt routines. There are separate sets of processor flags for normal, interrupt and non-maskable interrupt modes which are automatically switched and so these do not need to be saved.

The following list summarizes the interrupt procedure:

ST62xx actions

- Interrupt detection
- The flags C and Z of the main routine are exchanged with the flags C and Z of the interrupt routine (or the NMI flags)
- The value of the PC is stored in the first level of the stack
- The normal interrupt lines are inhibited (NMI still active)
- First internal latch is cleared
- The related interrupt vector is loaded in the PC.

User actions

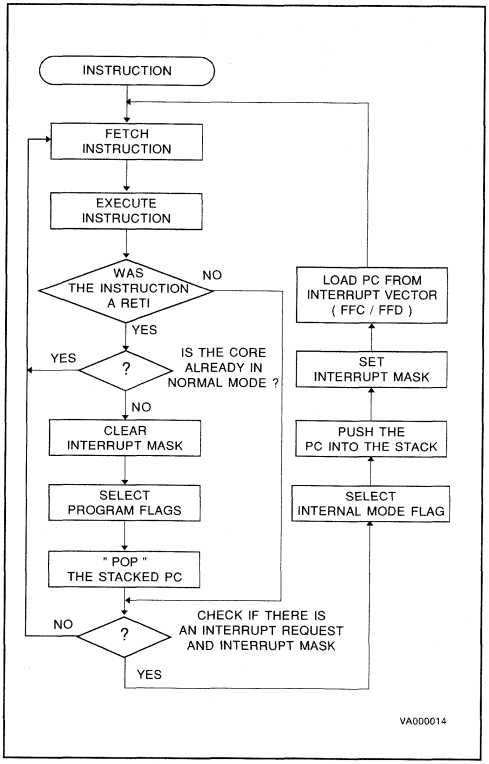
- User selected registers are saved inside the interrupt service routine (normally on a software stack)
- The source of the interrupt is found by polling (if more than one source is associated to the same vector) the interrupt flag of the source.
- Interrupt servicing
- Return from interrupt (RETI)

ST62xx actions

- Automatically the ST62xx core switches back to the normal flags (or the interrupt flags) and pops the previous PC value from the stack

The interrupt routine begins usually by the identification of the device that has generated the interrupt request (by polling). The user should save the registers which are used inside the interrupt routine (that holds relevant data) into a software stack. After the RETI instruction execution, the core carries out the previous actions and the main routine can continue.

Figure 13. Interrupt Processing Flow-Chart



WARNING. *GEN is the global enable for all interrupts except NMI. If this bit is cleared, the NMI interrupt is accepted when the ST62xx core is in the normal RUN Mode.*
If the ST62xx core is in STOP or WAIT Mode, the NMI is not accepted as a restart is disabled. This state can only be finished by a reset (fromthe Watchdog or an external Reset Signal).
As a consequence the NMI can be masked in STOP and WAIT modes, but not in RUN mode.

INTERRUPTS (Continued)

Interrupt request and mask bits

Interrupt Option Register, IOR Location C8h

- GEN. If this bit is set, all the ST62xx interrupts are enabled, if reset all interrupts are disabled (including the NMI).
- ESB. If this bit is set, all the input lines associated to interrupt vector #2 are rising edge sensitive, if reset they are falling edge sensitive.
- LES. If this bit is set, all the inputs lines associated to interrupt vector #1 are low level sensitive, if reset they are falling edge sensitive.

All other bits in this register are not used.

Timer Peripherals, TSCR1 and TSCR2 registers, locations D4h and D7h

- TMZ. A low-to-high transition indicates that the timer count register has decremented to zero. This means that an interrupt request can be generated in relation to the state of ETI bit.
- ETI. This bit, when set, enables the timer interrupt request.

A/D Converter Peripheral, ADCR register location D0h

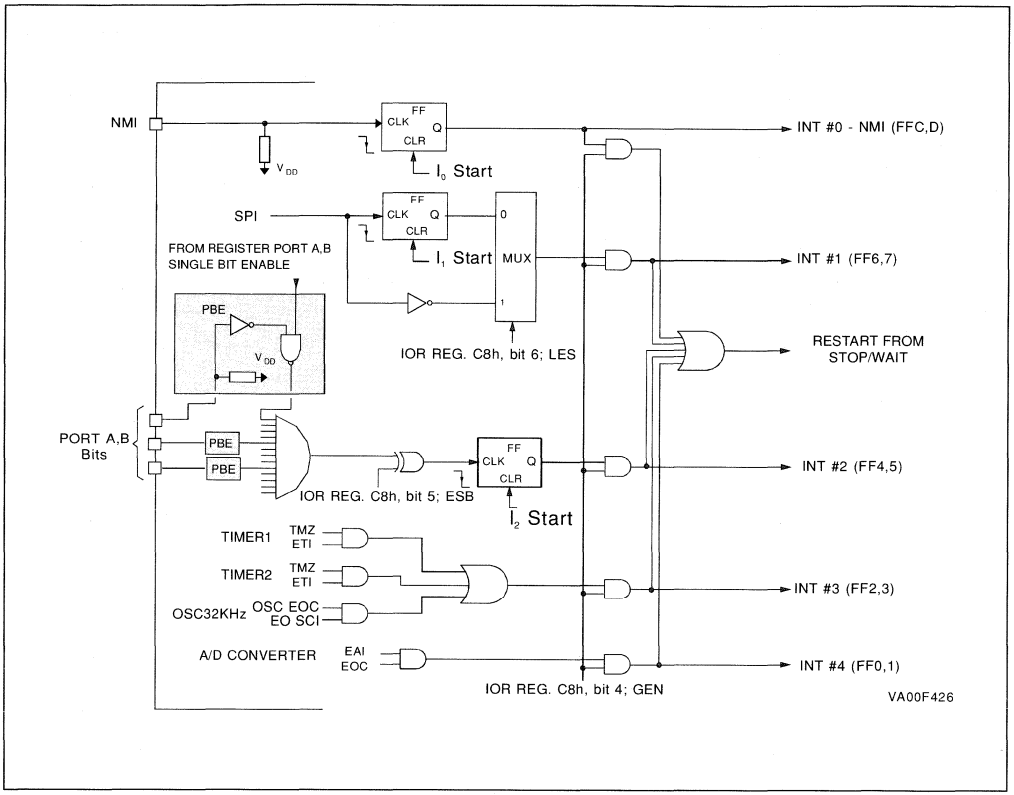
- EOC. This read only bit indicates when a conversion has been completed, by going to one. An interrupt request can be generated in relation to the state of EAI bit.
- EAI. This bit, when set, enables the A/D converter interrupt request.

32kHz Oscillator, 320CR register location DBh

- EOSCI. This bit, when set, enables the 32kHz oscillator interrupt request.
- OSCEOC. This read only bit indicates when the 32kHz oscillator has measured a 500ms elapsed time (providing a 32.768kHz quartz crystal is connected to the 32kHz oscillator dedicated pins). An interrupt request can be generated in relation to the state of EOSCI bit.

INTERRUPTS (Continued)

Figure 14. ST6245 Interrupt Circuit Diagram



RESET

The ST6245 can be reset in three ways: by the external reset input (RESET) tied low, by power-on reset and by the digital watchdog/timer peripheral.

RESET Input

The RESET pin can be connected to a device of the application board in order to restart the MCU during its operation. The activation of the RESET pin may occur in the RUN, WAIT or STOP mode. This input has to be used to reset the MCU internal state and provide a correct start-up procedure. The pin is active low. The internal reset signal is generated by adding a delay to the external signal. Therefore even short pulses at the RESET pin will be accepted. This feature is valid providing that V_{DD} has finished its rising phase and the oscillator is correctly running (normal RUN or WAIT modes).

If RESET activation occurs in the RUN or Wait mode, the MCU is configured in the Reset mode for as long as the signal of the RESET pin is low. The processing of the program is stopped (in RUN mode only) and the Input/Outputs are in the High-impedance state with pull-up resistors switched on. As soon as the level on the RESET pin becomes high, the initialization sequence is executed.

If a RESET pin activation occurs in the STOP mode, the oscillator starts and all the inputs/outputs are configured in the High-impedance state with pull-up resistors switched on for as long as the level on the RESET pin remains low. When the level of the RESET pin becomes high, a delay is generated by the ST62xx core to wait that the oscillator becomes completely stabilized. Then, the initialization sequence is started.

Power-On Reset (POR)

The function of the POR consists in waking up the MCU during the power-on sequence. At the beginning of this sequence, the MCU is configured in the Reset state: every Input/Output port is configured in the input mode (High-impedance state with pull-up) and no instruction is executed. When the power supply voltage becomes sufficient, the oscillator starts to operate, nevertheless the ST62xx core generates a delay to allow the oscillator to be completely stabilized before the execution of the first instruction. The initialization sequence is then executed.

Internal circuitry generates a Reset pulse when V_{DD} is switched on. In the case of fast rising V_{DD} (transition time $\leq 100\mu s$), this reset pulse starts the internal reset procedure without the need of external components at the RESET pin. In cases of slowly or non monotonously rising V_{DD}, an external reset signal must be provided for a proper reset of the MCU.

For as long as the reset pin is kept at the low level, the processor remains in the reset state. The reset will be released after the voltage at the reset pin reaches the high level.

Note:

To have a correct ST62xx start-up, the user should take care that the reset input does not change to the high level before the V_{DD} level is sufficient to allow MCU operation at the chosen frequency (see recommended operating conditions).

An on-chip counter circuit provides a delay of 2048 oscillator cycles between the detection of the reset high level and the release of the MCU reset.

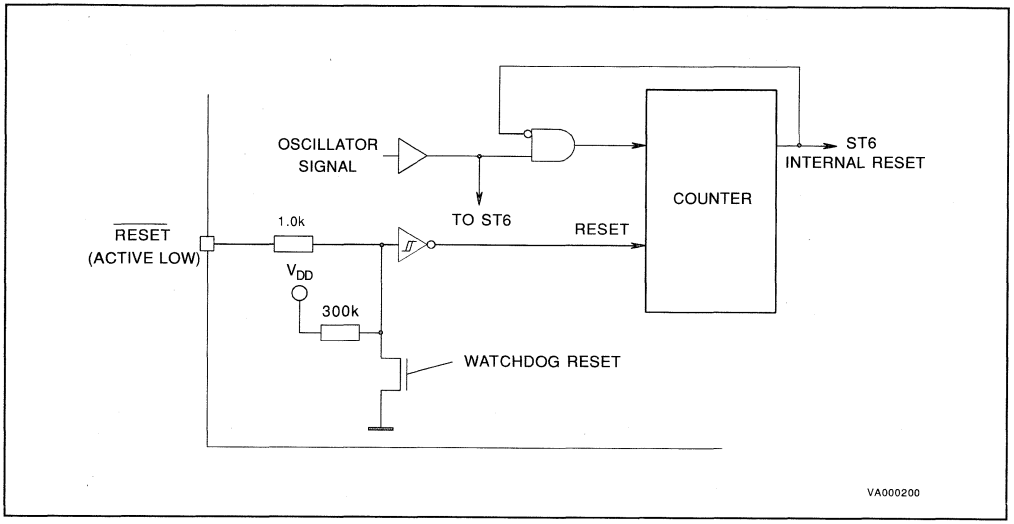
A proper reset signal for slow rising V_{DD}, i.e. the required delay between reaching sufficient operating voltage and the reset input changing to a high level, can be generally provided by an external capacitor connected between the RESET pin and V_{SS}.

RESET (Continued)

Watchdog Reset

The ST6245 provides an on-chip watchdog function in order to provide a graceful recovery from a software upset. If the watchdog register is not refreshed, preventing the end-of-count being reached, an internal circuit pulls down the $\overline{\text{RESET}}$ pin. The MCU will enter the reset state as soon as the voltage at $\overline{\text{RESET}}$ pin reaches the related low level. This also resets the watchdog which subsequently turns off the pull-down and activates the pull-up device at the $\overline{\text{RESET}}$ pin. This causes the positive transition at the $\overline{\text{RESET}}$ pin and terminates the reset state.

Figure 15. Reset Circuit

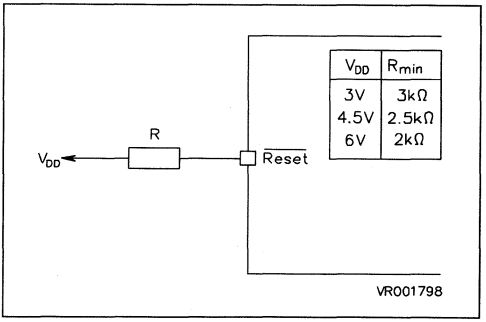


Application Notes

An external resistor between V_{DD} and reset pin is not required because an internal pull-up device is provided. If the user prefers, for any reason, to add an external pull-up resistor its value must comply with the Rmin value defined in Figure 16. If the value is lower than Rmin, the on-chip watchdog pull-down transistor might not be able to pull-down the reset pin resulting in an external deactivation of the watchdog function.

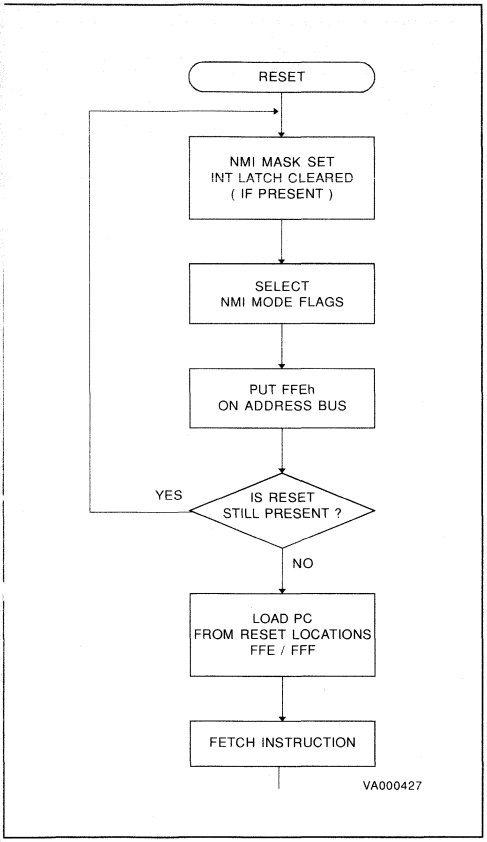
The POR function operates in a dynamic manner in the way that it brings about the initialization of the MCU when it detects a dynamic rising edge of the V_{DD} voltage. The typical detected threshold is about 2 volts, but the actual value of the detected threshold depends on the way in which the V_{DD} voltage rises up. The POR device DOES NOT allow the supervision of a static rising or falling edge of the V_{DD} voltage.

Figure 16. External Reset Resistance



RESET (Continued)

Figure 17. Reset & Interrupt Processing Flow-Chart



MCU Initialization Sequence

When a reset occurs the stack is reset to the program counter, the PC is loaded with the address of the reset vector (located in the program ROM at addresses FFEh & FFFh). A jump instruction to the beginning of the program has to be written into these locations. After a reset the interrupt mask is automatically activated so that the core is in non-maskable interrupt mode to prevent false or ghost interrupts during the restart phase. Therefore the restart routine should be terminated by a RETI instruction to switch to normal mode and enable interrupts. If no pending interrupt is present at the end of the reset routine the ST62xx will continue with the instruction after the RETI; otherwise the pending interrupt will be serviced.

Figure 18. Restart Initialization Program Flow-Chart

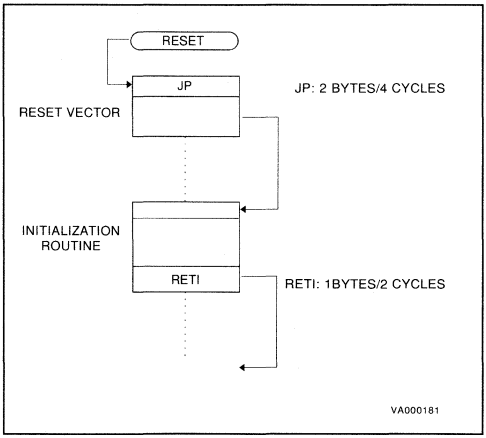


Table 6. Reset Configuration

Input/Output pins	Registers
Input Mode with pull-up and no interrupt	All cleared but A,X,Y,V,W, data RAM, LCD RAM, DWR (C9), PRPR (CA), DRBR (CB). Timers prescaler and TCR are initialized respectively at 7F and FF. Watchdog register DWDR (D8) is set to FEh.

WAIT & STOP MODES

The WAIT and STOP modes have been implemented in the ST62xx core in order to reduce the consumption of the product when the latter has no instruction to execute. These two modes are described in the following paragraphs

WAIT Mode

The configuration of the MCU in the WAIT mode occurs as soon as the WAIT instruction is executed. The microcontroller can also be considered as being in a "software frozen" state where the core stops processing the instructions of the routine, the contents of the RAM locations and peripheral registers are saved as long as the power supply voltage is higher than the RAM retention voltage but where the peripherals are still working.

The WAIT mode is used when the user wants to reduce the consumption of the MCU when it is in idle, while not losing count of time or monitoring of external events. The oscillator is not stopped in order to provide a clock signal to the peripherals. The timer counting may be enabled (writing the PSI bit in TSCR register) and the timer interrupt may be also enabled before entering the WAIT mode; this allows the WAIT mode to be left when timer interrupt occurs. The above explanation related to the timers applies also to the A/D converter.

If the exit from the WAIT mode is performed with a general RESET (either from the activation of the external pin or by watchdog reset) the MCU will enter a normal reset procedure as described in the RESET chapter. If an interrupt is generated during WAIT mode the MCU behavior depends on the state of the ST62xx core before the initialization of the WAIT sequence, but also of the kind of the interrupt request that is generated. This case will be described in the following paragraphs. In any case, the ST62xx core does not generate any delay after the occurrence of the interrupt because the oscillator clock is still available.

STOP Mode

If the Watchdog is disabled the STOP mode is available. When in STOP mode the MCU is placed in the lowest power consumption mode. In this operating mode the microcontroller can be considered as being "frozen", no instruction is executed the oscillator is stopped, the contents of the RAM locations and peripheral registers are saved as long as the power supply voltage is higher than the RAM retention voltage, and the ST62xx core waits for the occurrence of an external interrupt request or Reset activation to output from the STOP state.

If the exit from the STOP mode is performed with a general RESET (by the activation of the external pin) the MCU will enter a normal reset procedure as described in the RESET chapter. The case of an interrupt depends on the state of the ST62xx core before the initialization of the STOP sequence and also of the kind of the interrupt request that is generated.

This case will be described in the following paragraphs. In any case, the ST62xx core generates a delay after the occurrence of the interrupt request in order to wait the complete stabilization of the oscillator before the execution of the first instruction.

Exit from WAIT and STOP Modes

The following paragraphs describe the output procedure of the ST62xx core from WAIT and STOP modes when an interrupt occurs (not a RESET). It must be noted that the restart sequence depends on the original state of the MCU (normal, interrupt or non-maskable interrupt mode) before the start of the WAIT or STOP sequence, but also of the type of the interrupt request that is generated.

Normal Mode. If the ST62xx core was in the main routine when the WAIT or STOP instruction has been executed, the ST62xx core outputs from the stop or wait mode as soon as any interrupt occurs; the related interrupt routine is executed and at the end of the interrupt service routine the instruction that follows the STOP or the WAIT instruction is executed if no other interrupts are pending.

WAIT & STOP MODES (Continued)

Not Maskable Interrupt Mode. If the STOP or WAIT instruction has been executed during the execution of the non-maskable interrupt routine, the ST62xx core outputs from the stop or wait mode as soon as any interrupt occurs: the instruction that follows the STOP or the WAIT instruction is executed and the ST62xx core is still in the non-maskable interrupt mode even if another interrupt has been generated.

Normal Interrupt Mode. If the ST62xx core was in the interrupt mode before the initialization of the STOP or WAIT sequence, it outputs from the stop or wait mode as soon as any interrupt occurs. Nevertheless, two cases have to be considered:

- If the interrupt is a normal interrupt, the interrupt routine in which the WAIT or STOP was entered will be completed with the execution of the instruction that follows the STOP or the WAIT and the ST62xx core is still in the interrupt mode. At the end of this routine pending interrupts will be serviced in accordance to their priority.
- If the interrupt is a non-maskable interrupt, the non-maskable routine is processed at first. Then the routine in which the WAIT or STOP was entered will be completed with the execution of the instruction that follows the STOP or the WAIT and the ST62xx core remains in the normal interrupt mode.

Notes :

To reach the lowest power consumption the user software must take care of:

- placing the A/D converter in its power down mode by clearing the PDS bit in the A/D control register before entering the STOP instruction.
- switching off the 32kHz oscillator by clearing the oscillator start/stop bit in the 32kHz oscillator control register.
- putting the EEPROM on-chip memory in stand-by mode by setting the E2OFF bit in EEPROM Control Register to one.

The LCD Driver peripheral is automatically switched-off by the STOP instruction when the 32kHz oscillator operation is not selected.

When the hardware activated watchdog is selected or the software watchdog enabled, the STOP instruction is deactivated and any attempt to execute the STOP instruction will cause an execution of a WAIT instruction.

If all the interrupt sources are disabled (including NMI if GEN="0"), the restart of the MCU can only be done by a $\overline{\text{RESET}}$ activation. The WAIT and STOP instructions are not executed if an enabled interrupt request is pending.

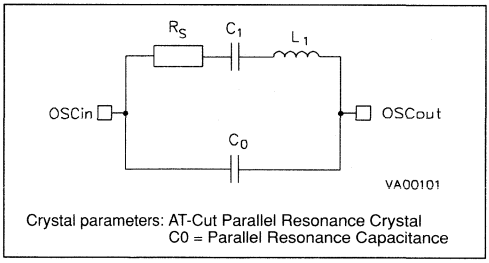
ON-CHIP CLOCK OSCILLATOR

The internal oscillator circuit is designed to require a minimum of external components. A crystal, a ceramic resonator, or an external signal (provided to the OSCin pin) may be used to generate a system clock with various stability/cost tradeoffs. The different clock generator options connection methods are shown in Figure 20.

One machine cycle takes 13 oscillator pulses; 12 clock pulses are needed to increment the PC while and additional 13th pulse is needed to stabilize the internal latches during memory addressing. This means that with a clock frequency of 8MHz the machine cycle is 1.625µs.

The crystal oscillator start-up time is a function of many variables: crystal parameters (especially R_S), oscillator load capacitance (CL), IC parameters, ambient temperature, supply voltage. It must be observed that the crystal or ceramic leads and circuit connections must be as short as possible. Typical values for CL1, CL2 are 15-22pF for a 4/8MHz crystal. The oscillator output frequency is internally divided by 13 to produce the machine cycle and by 12 to produce the Timer, the Watchdog and the A/D peripheral clock. A machine cycle is the smallest unit needed to execute any operation (i.e., increment the program counter). An instruction may need two, four, or five machine cycles to be executed.

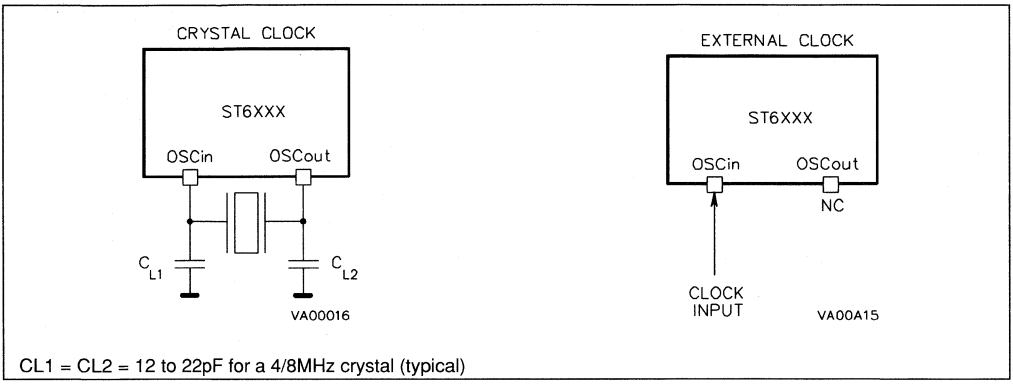
Figure 19. Crystal Parameters



Crystal parameters: AT-Cut Parallel Resonance Crystal
C0 = Parallel Resonance Capacitance

ON-CHIP CLOCK OSCILLATOR (Continued)

Figure 20. Oscillator Connection



CL1 = CL2 = 12 to 22pF for a 4/8MHz crystal (typical)

INPUT/OUTPUT PORTS

The ST6245 microcontroller has 11 Input/Output lines that can be individually programmed either in the input mode or the output mode with the following options that can be selected by software:

- Input without pull-up and without interrupt
- Input with pull-up and with interrupt
- Input with pull-up without interrupt
- Analog inputs (PA5-PA7, PB0-PB3)
- SPI control signals (PB5-PB7)
- Push-pull output
- Standard Open drain output
- 20mA Open drain output (PB4-PB7)

The lines are organized in two ports (port A,B).

Each port occupies 3 registers in the data space. Each bit of these registers is associated with a particular line (for instance, the bits 0 of the Port A Data, Direction and Option registers are associated with the PA0 line of Port A).

The two DATA registers (DRA, DRB), are used to read the voltage level values of the lines programmed in the input mode, or to write the logic value of the signal to be output on the lines configured in the output mode. The port data registers can be read to get the effective logic levels of the pins, but they can be also written by the user software, in conjunction with the related option registers, to select the different input mode options.

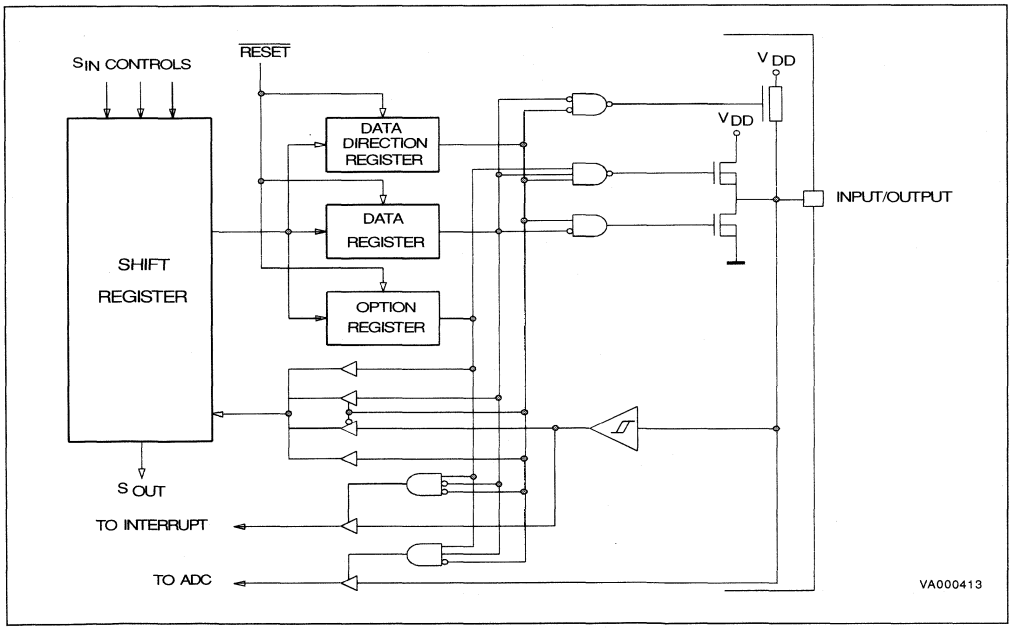
Single-bit operations on I/O registers are possible but care is necessary because reading in input mode is done from I/O pins while writing will directly affect the Port data register causing an undesired change of the input configuration.

The two Data Direction registers (DDRA, DDRB) allow the selection of the data direction of each pin (input or output).

The two Option registers (ORPA, ORPB) are used to select the different port options available both in input and in output mode.

All the I/O registers can be read or written as any other RAM location of the data space, so no extra RAM cell is needed for port data storing and manipulation. During the initialization of the MCU, all the I/O registers are cleared and the input mode with pull-up/no-interrupt is selected on all the pins, thus avoiding pin conflicts.

Figure 21. I/O Port Block Diagram



INPUT/OUTPUT PORTS (Continued)

I/O Pin Programming

Each pin can be individually programmed as input or output with different input and output configurations.

This is achieved by writing to the relevant bit in the data (DR), data direction register (DDR) and option registers (OR). Table 7 shows all the port configurations that can be selected by user software.

Input Option Description

Pull-up, High Impedance Option. All the input lines can be individually programmed with or without an internal pull-up according to the codes programmed in the OR and DR registers . If the pull-up option is not selected, the input pin is in the high-impedance state.

Interrupt Option. All the input lines can be individually connected by software to the interrupt lines of the ST62xx core according to the codes programmed in the OR and DR registers. The pins of Port A and B are "ORed" and are connected to the interrupt associated to the vector #2. The interrupt modes (falling edge sensitive, rising edge sensitive) can be selected by software for each port by programming the IOR register.

Analog Input Option. The seven PA5-PA7, PB0-PB3 pins can be configured to be analog inputs according to the codes programmed in the OR and DR registers. These analog inputs are connected to the on-chip 8-bit Analog to Digital Converter. *ONLY ONE* pin should be programmed as analog input at a time, otherwise the selected inputs will be shorted.

Figure 22. I/O Port Data Registers

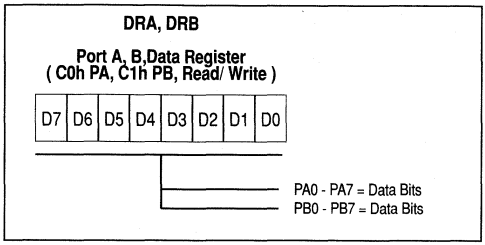


Figure 23. I/O Port Data Direction Registers

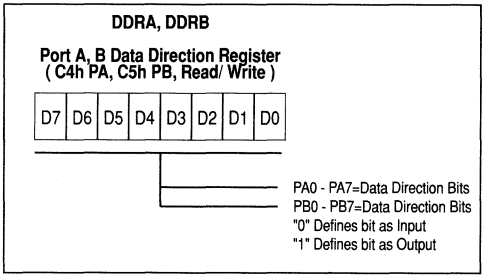
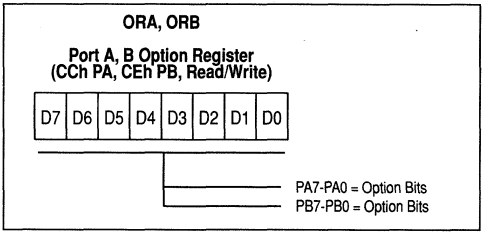


Figure 24. I/O Port Option Registers



Note: For complete coding explanation refer to Table 8.

Table 7. I/O Port Options Selection

DDR	OR	DR	MODE	OPTION
0	0	0	Input	With pull-up, no interrupt (Reset state)
0	0	1	Input	No pull-up, no interrupt
0	1	0	Input	With pull-up, with interrupt
0	1	1	Input	No pull-up, no interrupt (for PB4-PB7).
			Input	Analog input (for PA0-PA7, PB0-PB3)
1	0	X	Output	Open-drain output (20mA sink current for PB4-PB7)
1	1	X	Output	Push-pull output (20mA sink current for PB4-PB7)

Note: X. Means don't care.

INPUT/OUTPUT PORTS (Continued)

SPI alternate function Option. The I/O pins PB5-PB7 are also used by serial peripheral interface SPI. PB5 is connected with the SPI clock input SCL, PB6 is connected with the SPI data input SIN and PB7 is connected with the SPI data output SOUT.

For serial input operation PB5 and PB6 have to be programmed as inputs. For serial output operation PB7 has to be programmed as open-drain output (DDR = "1", OPR = "0"). In this operating mode the output of the SPI shift register instead of the port data register is connected to the port buffer. When PB7 is programmed as push-pull output (DDR = "1", OPR = "1"), the port data register is connected to the port buffer. When the SPI peripheral is not used PB5-PB7 can be used as general purpose I/O lines (provided that PB7 is not selected to be open-drain in output mode).

Notes:

Switching the I/O ports from one state to another should be done in a way that no unwanted side effects can happen. The recommended safe transitions are shown below. All other transitions are risky and should be avoided during change of operation mode as it is most likely that there will be an unwanted side-effect such as interrupt generation or two pins shorted together by the analog input lines.

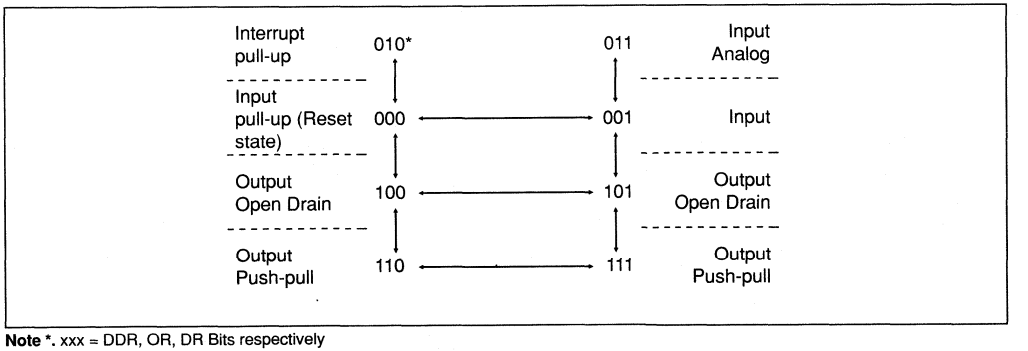
Single bit SET and RES instructions should be used very carefully with Port A and B data registers because these instructions make an implicit read and write back of the whole addressed register byte. In port input mode however data register address reads from input pins, not from data register latches and data register information in input mode is used to set characteristics of the input pin (interrupt, pull-up, analog input), therefore these characteristics may be unintentionally reprogrammed depending on the state of input pins. As general rule is better to use SET and RES instructions on data register only when the whole port is in output mode. If input or mixed configuration is needed it is recommended to keep a copy of the data register in RAM. On this copy it is possible to use single bit instructions, then the copy register could be written into the port data register.

```
SET     bit, datacopy
LD      a, datacopy
LD      DRA, a
```

The WAIT and STOP instructions allow the ST62xx to be used in situations where low power consumption is needed. The lowest power consumption is achieved by configuring I/Os in input mode with well-defined logic levels.

The user has to take care not to switch outputs with heavy loads during the conversion of one of the analog inputs in order to avoid any disturbance in the measurement.

Figure 25. I/O Port StateTransition Diagram for Safe Transitions



Note *. xxx = DDR, OR, DR Bits respectively

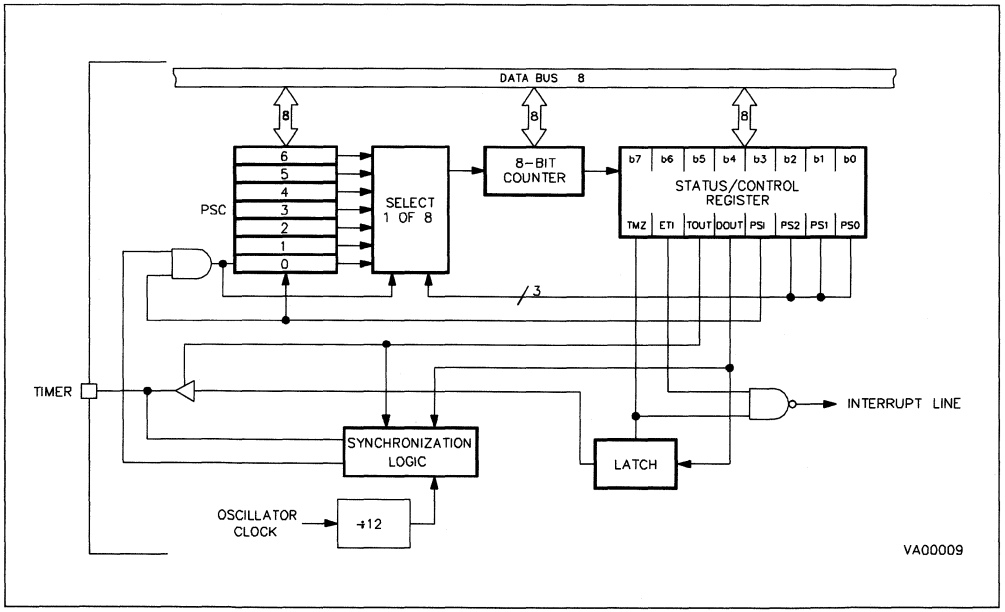
TIMERS

The ST6245 offers two on-chip Timer peripherals named Timer 1 and Timer 2. Each of these timers consists of an 8-bit counter with a 7-bit programmable prescaler, thus giving a maximum count of 2^{15}, and control logic that allows configuring the peripheral in three operating modes. Figure 26 shows the Timer block diagram. Timer 1 only has the external TIMER pin available for the user. The content of the 8-bit counter can be read/written in the Timer/Counter register TCR which is addressed in the data space as a RAM location at addresses D3h or D6h. The state of the 7-bit prescaler can be read in the PSC register at addresses D2h or D5h. The control logic device is managed in the TSCR register (addresses D4h or D7h) as described in the following paragraphs.

The 8-bit counter is decrement by the output (rising edge) coming from the 7-bit prescaler and can be loaded and read under program control. When it decrements to zero then the TMZ (Timer Zero) bit in the TSCR is set to one. If the ETI (Enable Timer Interrupt) bit in the TSCR is also set to one an interrupt request, associated to interrupt vector #3, is generated. The interrupt service routine then would determine which timer reached the end of count by poling the TMZ bits. The Timer interrupt can be used to exit the MCU from the WAIT mode.

The prescaler input can be the oscillator frequency divided by 12 or an external clock at TIMER pin (only Timer 1). The prescaler decrements on the rising edge. Depending on the division factor programmed by PS2, PS1 and PS0 bits in the TSCR, the clock input of the timer/counter register is multiplexed to different sources. On division factor 1, the clock input of the prescaler is also that of timer/counter; on factor 2, bit 0 of prescaler register is connected to the clock input of TCR. This bit changes its state with the half frequency of prescaler clock input. On factor 4, bit 1 of PSC is connected to clock input of TCR, and so on. The prescaler initialize bit (PSI) in the TSCR register must be set to one to allow the prescaler (and hence the counter) to start. If it is cleared to zero then all of the prescaler bits are set to one and the counter is inhibited from counting. The prescaler can be given any value between 0 and 7Fh by writing to addresses D2h or D5h, if bit PSI in the TSCR register is set to one. The tap of the prescaler is selected using the PS2,PS1,PS0 bits in the control register. Figure 27 shows the Timer working principle.

Figure 26. Timer Peripheral Block Diagram



TIMERS (Continued)

Timer Operating Modes

There are 3 operating modes of the Timer peripheral. They are selected by the bits TOUT and DOUT (see TSCR register). These three modes correspond to the two clock frequencies that can be connected on the 7-bit prescaler ($f_{OSC}/12$ or TIMER pin signal) and to the output mode. For this reason, only Timer 1 has all three modes, while Timer 2, which does not have a dedicated TIMER pin, must be programmed in Output Mode only.

Clock Input Mode (TOUT = "0", DOUT = "0"). In this mode the TIMER pin is an input and the prescaler is decremented on rising edge. The maximum input frequency that can be applied to the external pin in this mode is 1/8 of the oscillator frequency. This operating mode is not available on Timer 2.

Gated Mode (TOUT = "0", DOUT = "1"). In this mode the prescaler is decremented by the Timer clock input (oscillator divided by 12) but ONLY when the signal at TIMER pin is held high (giving a pulse width measurement potential). This mode is selected by the TOUT bit in TSCR register cleared to "0" (i.e. as input) and DOUT bit set to "1". This operating mode is not available on Timer 2.

Output Mode (TOUT = "1", DOUT = data out). The TIMER pin is connected to the DOUT latch. Therefore the timer prescaler is clocked by the prescaler clock input ($f_{OSC}/12$).

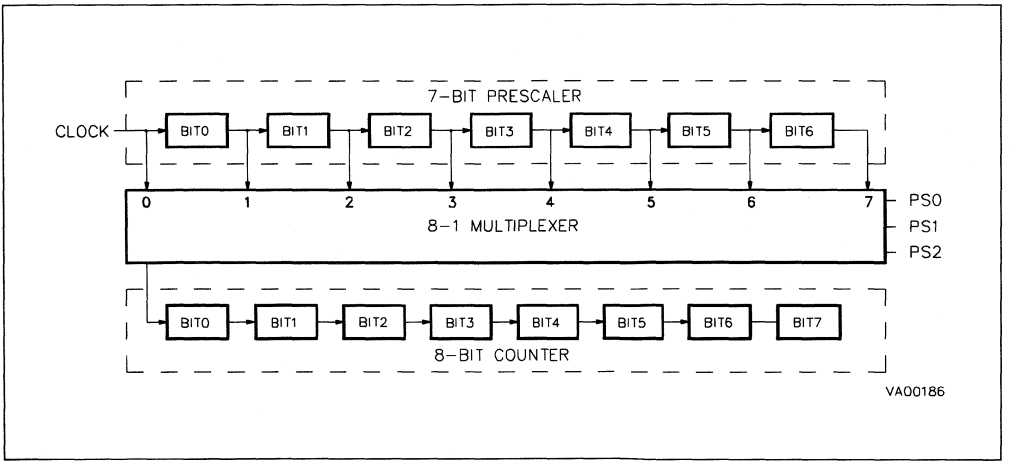
The user can select the desired prescaler division ratio through the PS2, PS1, PS0 bits. When TCR count reaches 0, it sets the TMZ bit in the TSCR. The TMZ bit can be tested under program control to perform a timer function whenever it goes high. The low-to-high TMZ bit transition is used to latch the DOUT bit of the TSCR and pass it to TIMER pin. This operating mode allows external signal generation on the TIMER pin. This is the only operating Mode allowed on Timer 2.

Table 8. Timer Operating Modes

TOUT	DOUT	Timer Pin	Timer Function
0	0	Input	Event Counter (1)
0	1	Input	Input Gated (1)
1	0	Output	Output
1	1	Output	Output

Note 1. Not allowed on Timer 2

Figure 27. Timer Working Principle



TIMERS (Continued)

Timer Interrupt

When the counter register decrements to zero and the software controlled ETI (Enable Timer Interrupt) bit is set to one then an interrupt request associated to interrupt vector #3 is generated. When the counter decrements to zero also the TMZ bit in the TSCR register is set to one.

Since only one interrupt vector is available for the two timers (ORed also with 32kHz oscillator interrupt), the interrupt service routine should determine from which source the interrupt came by polling the TMZ bits (and the OSCEOC bit of the 32kHz oscillator control register).

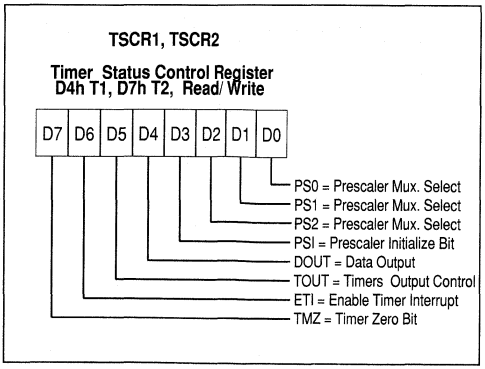
Notes:

TMZ is set when the counter reaches 00h; however, it may be set by writing 00h in the TCR register or setting bit 7 of the TSCR register. TMZ bit must be cleared by user software when servicing the timer interrupt to avoid undesired interrupts when leaving the interrupt service routine. After reset, the 8-bit counter register is loaded to FFh while the 7-bit prescaler is loaded to 7Fh , and the TSCR register is cleared which means that timer is stopped (PSI="0") and the timer interrupt is disabled.

If the Timer is programmed in output mode, DOUT bit is transferred to the TIMER pin when TMZ is set to one (by software or due to counter decrement). When TMZ is high, the latch is transparent and DOUT is copied to the timer pin. When TMZ goes low, DOUT is latched.

A write to the TCR register will predominate over the 8-bit counter decrement to 00h function, i.e. if a write and a TCR register decrement to 00h occur simultaneously, the write will take precedence, and the TMZ bit is not set until the 8-bit counter reaches 00h again. The values of the TCR and the PSC registers can be read accurately at any time.

Figure 28. Timer Status Control Register



TMZ. Low-to-high transition indicates that the timer count register has decremented to zero. This bit must be cleared by user software before starting with a new count.

ETI. This bit, when set, enables the timer interrupt request (vector #3). If ETI="0" the timer interrupt is disabled. If ETI="1" and TMZ="1" an interrupt request is generated.

TOUT. When low, this bit selects the input mode for the TIMER pin. When high the output mode is selected.

DOUT. Data sent to the timer output when TMZ is set high (output mode only). Input mode selection (input mode only). This bit is meaningless for Timer 2.

PSI. Used to initialize the prescaler and inhibit its counting. When PSI="0" the prescaler is set to 7Fh and the counter is inhibited. When PSI="1" the prescaler is enabled to count downwards. As long as PSI="0" both counter and prescaler are not running.

PS2, PS1, PS0. These bits select the division ratio of the prescaler register.

Table 9. Prescaler Division Factors

PS2	PS1	PS0	Divided by	PS2	PS1	PS0	Divided by
0	0	0	1	1	0	0	16
0	0	1	2	1	0	1	32
0	1	0	4	1	1	0	64
0	1	1	8	1	1	1	128

Figure 29. Timer Counter Register

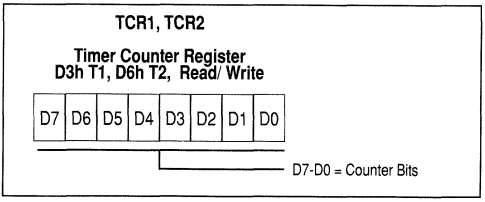
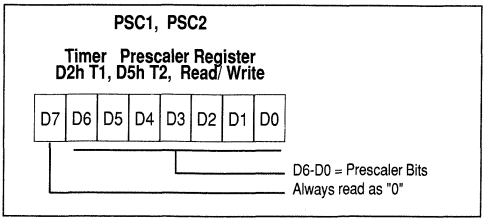


Figure 30. Prescaler Register



DIGITAL WATCHDOG

The digital Watchdog of the ST62 device consists of a down counter that can be used to provide a controlled recovery from a software upset.

The ST6245 watchdog is a the software activated watchdog.

The software activated digital watchdog consists of a down counter that can be used to provide a controlled recovery from a software upset. The watchdog uses one data space register (DWDR location D8h). The watchdog register is set to FEh after reset and the watchdog function is disabled. The watchdog time can be programmed using the 6 Most Significant Bits in the Watchdog register. The check time can be set differently for different routines within the general program.

After a reset the software Watchdog is in the off-state. The watchdog should be activated inside the Reset restart routine by writing a "1" in watchdog timer register bit 0. Bit one of this register must be set to one before programming bit zero as otherwise a reset will be immediately generated when bit 0 is set. This allows the user to generate a reset by software (bit 0 = "1", bit 1 = "0"). Once bit 0 is set, it can not be cleared by software without generating a Reset. The delay time is defined by programming bits 2-7 of the watchdog register. Bit 7 is the Least Significant Bit while bit 2 is the MSB.

This gives the possibility to generate a reset in a time between 3072 to 196608 clock cycles in 64 possible steps: (With a clock frequency of 8MHz this means from 384µs to 24.576ms). The reset is prevented if the register is reloaded with the desired value before bits 2-7 decrement from all zeros to all ones. If the watchdog is active the STOP instruction is deactivated and a WAIT instruction is automatically executed instead of a STOP. If bit 0 of the watchdog register is never set to one then bits 1-7 of the register can be used as a simple 7-bit counter which is decrement every 3072 clock cycles.

Figure 32. Watchdog Register

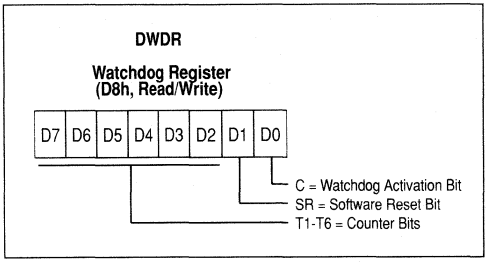
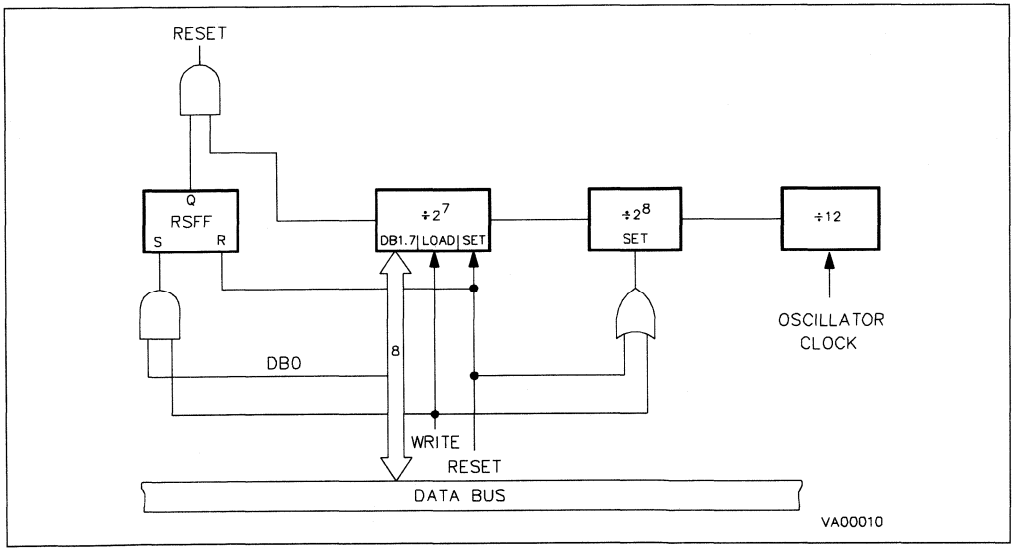


Figure 31. Digital Watchdog Block Diagram



DIGITAL WATCHDOG (Continued)

Watchdog Register

C. This is the watchdog activation bit, that, if set to one, will activate the watchdog function. When cleared to zero it allows the use of the counter as a 7-bit timer. This bit is cleared on reset.

SR. This bit is set to one during the reset and will generate a software reset if cleared to zero. When C = "0" (watchdog disabled) it is the MSB of the 7-bit timer.

T1-T6. These are the watchdog counter bits. It should be noted that D7 (T1) is the LSB of the counter and D2 (T6) is the MSB of the counter. These bits are in the opposite order to normal.

Application note:

If the Watchdog is not used during power-on reset external noise may cause the undesired activation of the Watchdog with a generation of an unexpected reset. To avoid this risk, two additional instructions, that check the state of the watchdog and eventually reset the chip are needed within the first 27 instructions, after the reset. These instructions are:

```
jrx 0, WD, #+3
ldi WD, 0FDH
```

These instructions should be executed at the very beginning of the customer program.

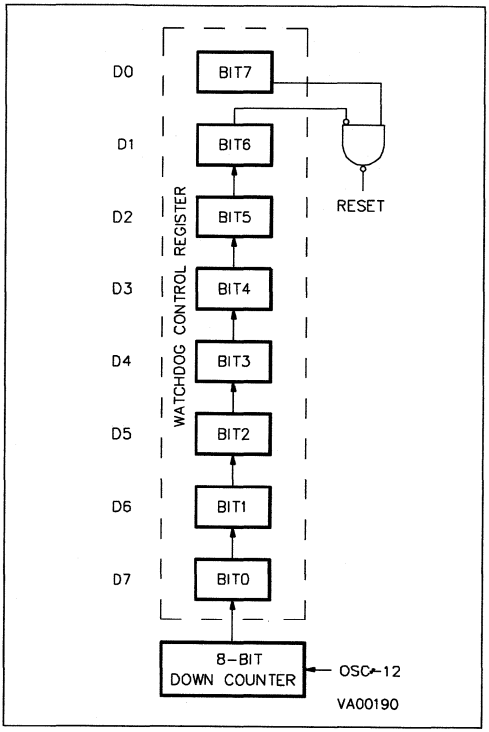
If the Watchdog is used during power-on reset the Watchdog register may be set to a low value, that could give a reset after 28 instructions earliest. To avoid undesired resets, the Watchdog must be set to the desired value within the first 27 instructions, the best is to put at the very beginning.

Alternatively the normal legal state can be checked with the following short routine:

```
ldi a, 0FEH
and a, WD
cpi a, 0FEH
jrz #+3
ldi WD, 0FDH
```

This sequence is recommended for security applications, where possible stack confusion error loops must be avoided and the Watchdog must only be refreshed after extensive checks.

Figure 33. Watchdog Working Principle



8-BIT A/D CONVERTER

The A/D converter of ST6245 is an 8-bit analog to digital converter with up to 7 analog inputs (as alternate functions of I/O lines PA5-PA7, PB0-PB3) offering 8-bit resolution with total accuracy ±2 LSB and a typical conversion time of 70µs (clock frequency of 8MHz).

The A/D peripheral converts the input voltage by a process of successive approximations using a clock frequency derived from the oscillator with a division factor of twelve. With an oscillator clock frequency less than 1.2MHz, the A/D converter accuracy is decreased.

The selection of the pin signal that has to be converted is done by configuring the related I/O line as analog input through the I/O ports option and data registers (refer to I/O ports description for additional information). Only one I/O line must be configured as analog input at a time. The user must avoid the situation in which more than one I/O pin is selected to be analog input to avoid malfunction of the ST62xx.

The ADC uses two registers in the data space: the ADC data conversion register which stores the conversion result and the ADC control register used to program the ADC functions.

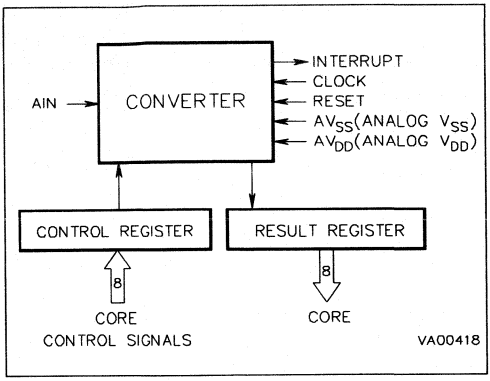
A conversion is started by writing a "1" to the Start bit (STA) in the ADC control register. This automatically clears (resets to "0") the End Of Conversion Bit (EOC). When a conversion has been finished this EOC bit is automatically set to "1" in order to flag that conversion is complete and that the data in the ADC data conversion register is valid. Each conversion has to be separately initiated by writing to the STA bit.

The STA bit is continually being scanned so that if the user sets it to "1" while a previous conversion is in progress then a new conversion is started before the previous one has been completed.The start bit (STA) is a write only bit, any attempt to read it will show a logical "0".

The A/D converter has a maskable interrupt associated to the end of conversion. This interrupt is associated to the interrupt vector #4 and occurs when the EOC bit is set, i.e. when a conversion is completed. The interrupt is masked using the EAI (interrupt mask) bit in the control register.

The power consumption of the device can be reduced by turning off the ADC peripheral. That is achieved when the PDS bit in the ADC control register is cleared to "0". If PDS="1", the A/D is supplied and enabled for conversion. This bit must be set at least one instruction before the beginning of the conversion to allow the stabilization of the A/D converter. *This action is needed also before entering the STOP instruction as the A/D comparator is not automatically disabled by the STOP mode*

Figure 34. A/D Converter Block Diagram



During reset any conversion in progress is stopped, the control register is reset to all zeros and the A/D interrupt is masked (EAI=0).

Notes:

The ST62xx A/D converter does not feature a sample and hold. The analog voltage to be measured should therefore be stable during the conversion time. Variation should not exceed ±1/2 LSB for the best accuracy in measurement.

Since the ADC is on the same chip as the microprocessor the user should not switch heavily loaded output signals during conversion if high precision is needed. This is because such switching will affect the supply voltages which are used for comparisons.

A low pass filter can be used at the analog input pins to reduce input voltage variation during the conversion. For true 8 bit conversions the impedance of the analog voltage sources should be less than 30kΩ while the impedance of the reference voltage should not exceed 2kΩ.

The accuracy of the conversion depends on the quality of the power supply voltages (V_{DD} and V_{SS}). The user must specially take care of applying regulated reference voltage on the V_{DD} and V_{SS} pins (the variation of the power supply voltage must be inferior to 5V/ms).

The converter can resolve the input voltage with a resolution of:

$$\frac{V_{DD} - V_{SS}}{256}$$

8-BIT A/D CONVERTER(Continued)

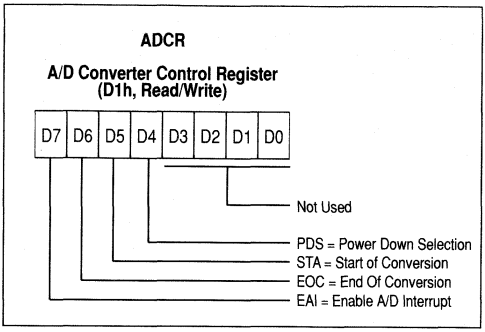
The Input voltage (Ain) which has to be converted must be constant for 1µs before conversion and remain constant during the conversion.

The resolution of the conversion can be improved if the power supply voltage (V_{DD}) of the microcontroller becomes lower.

In order to optimize the resolution of the conversion, the user can configure the microcontroller in the WAIT mode because this mode allows the minimization of the noise disturbances and the variations of the power supply voltages due to the switching of the outputs. Nevertheless, it must be take care of executing the WAIT instruction as soon as possible after the beginning of the conversion because the execution of the WAIT instruction may provide a small variation of the V_{DD} voltage (the negative effect of this variation is minimized at the beginning of the conversion because the latter is less sensitive than the end of the conversion when the less significant bits are determined).

The best configuration from a accuracy point of view is the WAIT mode with the Timer and LCD driver stopped. Indeed, only the ADC peripheral and the oscillator are still working. The MCU has to be wake-up from the WAIT mode by the interrupt of the ADC peripheral at the end of the conversion. It must be noticed that the wake-up of the microcontroller could be done also with the interrupt of the TIMER, but in this case, the Timer is working and some noise could disturb the converter in terms of accuracy.

Figure 35. A/D Converter Control Register



EAI. If this bit is set to one the A/D interrupt (vector #4) is enabled, when EAI=0 the interrupt is disabled.

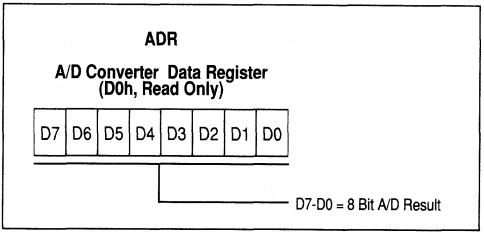
EOC. *Read Only*;This read only bit indicates when a conversion has been completed. This bit is automatically reset to zero when the STA bit is written. If the user is using the interrupt option then this bit can be used as an interrupt pending bit. Data in the data conversion register are valid only when this bit is set to one.

STA. *Write Only*; Writing a "1" in this bit will start a conversion on the selected channel and automatically reset to zero the EOC bit. If the bit is set again when a conversion is in progress, the present conversion is stopped and a new one will take place. This bit is write only, any attempt to read it will show a logical zero.

PDS. This bit activates the A/D converter if set to 1. Writing a zero into this bit will put the ADC in power down mode (idle mode).

D3-D0. Not used

Figure 36. A/D Converter Data Register



D7-D0. *Read Only*;These are the conversion result bits; the register is read only and stores the result of the last conversion. The contents of this register are valid only when EOC bit in the ADCR register is set to one (end-of-conversion).

32kHz STAND-BY OSCILLATOR

The 32kHz stand-by oscillator allows the ST6245 to generate real time interrupts and to supply the clock to the LCD driver. This enables the ST6245 to provide real time functions with the LCD display capability and lower power consumption. Figure 37 shows the 32kHz oscillator block diagram.

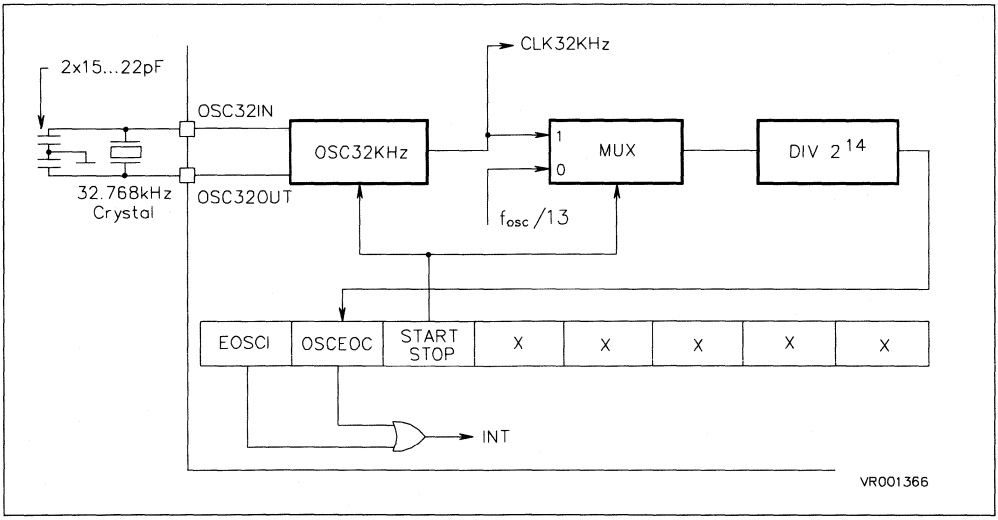
A 32.768kHz quartz crystal must be connected to the OSC32in and OSC32out pins to perform the real time clock operation. Two external capacitors of 15-22pF each must be connected between the oscillator pins and ground. The 32kHz oscillator is managed by the dedicated status/control register located at address 0DBh .

When the 32kHz stand-by oscillator is stopped (bit 5 of the Status/Control register cleared) the divider chain is supplied with a clock signal synchronous with machine cycle ($f_{OSC}/13$), this produces an interrupt request every $13x2^{14}$ clock cycle (i.e. 26.624ms) with an 8MHz quartz crystal.

When the 32kHz stand-by oscillator is enabled (bit 5 of the Status/Control register set to one) the divider chain is directly supplied with the 32kHz oscillator clock. The 32kHz clock from the standby oscillator can also be used as the LCD clock. This allows operation of the LCD in STOP mode. The interrupt output of the 32kHz oscillator peripheral generates an interrupt request every half second (500ms). This can be used to perform a real time clock function when the MCU is in STOP mode.

This interrupt signal is "ORed" with the interrupt request signals of the two on-chip timers and connected to the low level sensitive interrupt input associated to the interrupt vector #3 (FF2h, FF3h). The interrupt request has to be cleared by user software before leaving the interrupt service routine. Discrimination between the three interrupt sources is made by polling the Status/Control registers of Timer 1 (D4h), Timer 2 (D7h) and 32kHz oscillator (DBh).

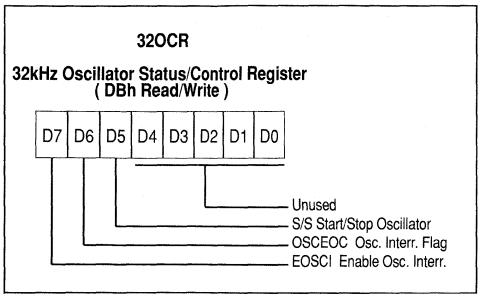
Figure 37. 32kHz Oscillator Block Diagram



32kHz STAND-BY OSCILLATOR (Continued)

32kHz Oscillator Status/Control Register

Figure 38. 32kHz Oscillator Register



EOSCI. Enable Oscillator Interrupt. This bit, when set, enables the 32kHz oscillator interrupt request.

OSCEOC. Oscillator Interrupt Flag. This bit indicates when the 32kHz oscillator has measured a 500ms elapsed time (providing a 32.768kHz quartz crystal is connected to the 32kHz oscillator dedicated pins). An interrupt request can be generated in relation to the state of EOSCI bit. This bit must be cleared by the user program before leaving the interrupt service routine.

START/STOP. Oscillator Start/Stop bit. This bit, when set, enables the 32kHz stand-by oscillator and the free running divider chain is supplied by the 32kHz oscillator signal. When this bit is cleared to zero the divider chain is supplied with the clock signal from the LCD Controller.

This register is cleared during reset.

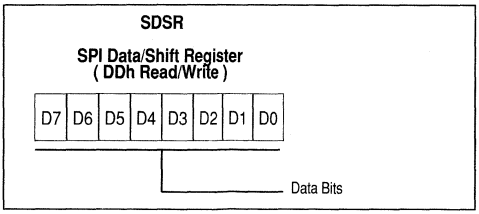
Note:
To achieve minimum power consumption in STOP mode (no system clock), the stand-by oscillator must be switched off (real time function not available) by clearing the Start/Stop bit in the oscillator status/control register.

SERIAL PERIPHERAL INTERFACE (SPI)

The ST6245 SPI is an optimized serial synchronous interface that supports a wide range of industry standard SPI specifications. The ST6245 SPI is controlled by small and simple user software to perform serial data exchange. The serial shift clock can be implemented either by software (using the bit-set and bit-reset instructions), with the on-chip Timer 1 by externally connecting the SPI clock pin to the timer pin or by directly applying an external clock to the SPI.

The peripheral is composed by an 8-bit Data/shift Register (address DDh) and a 4-bit binary counter. The SCL, Sin and Sout SPI data and clock signals are connected to the PA5, PA6 and PA7 I/O lines. With the 3 I/O pins, the SPI can operate in the following operating modes: Software SPI, S-BUS, I^2C-bus and as a standard serial I/O (clock, data, enable). An interrupt request can be generated after eight clock pulses. Figure 39 shows the SPI block diagram.

Figure 39. SPI Data/Shift Register



The PA5/SCL line clocks, on the falling edge, the shift register and the counter. To allow SPI operation the PA5/SCL must be programmed as input, an external clock supplied to this pin will drive the SPI peripheral (slave mode).

If PA5/SCL is programmed as output, a clock signal can be generated by software, setting and resetting the port line by software (master mode).

The SCL clock signal is the shift clock for the SPI data/shift register. The PA6/Sin pin is the serial shift input and PA7/Sout is the serial shift output. These two lines can be tied together to implement two wires protocols (I^2C-bus, etc). When data is serialized, the MSB is the first bit. PA6/Sin has to be programmed as input. For serial output operation PA7/Sout has to be programmed as open-drain output.

After 8 clock pulses (D7..D0) the output $\overline{Q4}$ of the 4-bit binary counter becomes low, disabling the clock from the counter and the data/shift register. Q4 enables the clock to generate an interrupt on the 8th clock falling edge as long as no reset of the counter (processor write into the 8-bit data/shift register) takes place. After a processor reset the interrupt is disabled. The interrupt is active when writing data in the shift register (DDh) and desactivated when writing any data in the register SPI Interrupt Disable (C2h).

SERIAL PERIPHERAL INTERFACE (Continued)

The generation of an interrupt to the Core provides information that new data is available (input mode) or that transmission is completed (output mode), allowing the Core to generate an acknowledge on the 9th clock pulse (I^2C-bus).

Since the SPI interrupt is connected to interrupt #1, the falling edge interrupt option should be selected by clearing to zero bit 6 of the Interrupt Option Register (IOR, C8h).

After power on reset, or after writing the data/shift register, the counter is reset to zero and the clock is enabled. In this condition the data shift register is ready for reception. No start condition has to be detected. Through the user software the Core may pull down the Sin line (Acknowledge) and slow down the SCL, as long as it is needed to carry out data from the shift register.

I^2C-bus Master-Slave, Receiver-Transmitter

When pins Sin and Sout are externally connected together it is possible to use the SPI as a receiver as well as a transmitter. With a simple software routine (by using bit-set and bit-reset on I/O line) a clock can be generated allowing I^2C-bus to work in master mode.

When implementing an I^2C-bus protocol, the start condition can be detected by setting the processor into a "wait for start" condition by simply enabling the interrupt of the PA6/Sin I/O port. This frees the processor from polling the Sin and SCL lines. After the transmission/reception the processor has to poll for the STOP condition.

In slave mode the user software can slow down the SCL clock frequency by simply putting the SCL I/O line in output open-drain mode and writing a zero into the corresponding data register bit.

As it is possible to directly read the Sin pin directly through the port register, the software can detect a difference between internal data and external data (master mode). Similar condition can be applied to the clock.

The typical speed of transmission in I^2C master or slave mode is in the range of 10kHz.

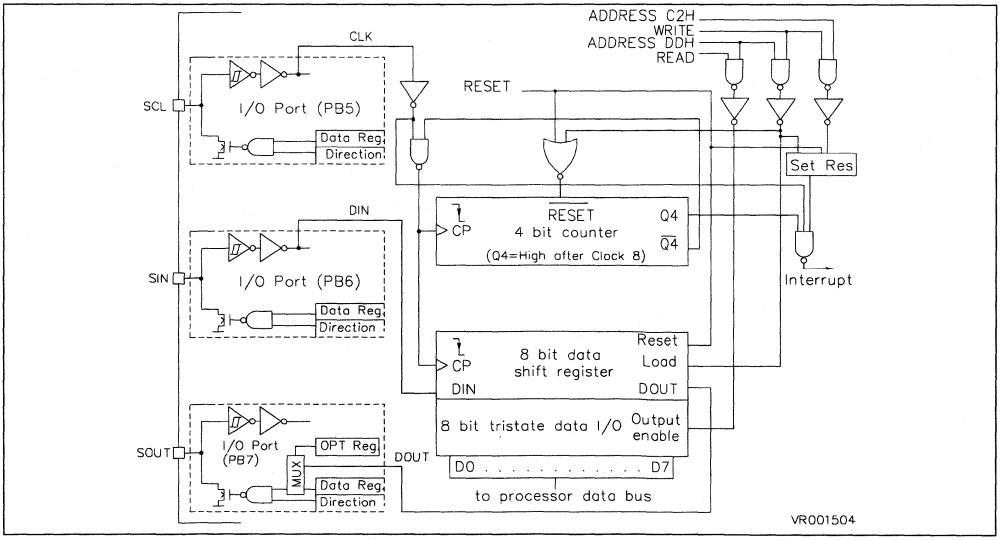
Three (Four) Wire Serial Bus

It is possible to use a single general purpose I/O pin (with the corresponding interrupt enabled) as a "chip enable" pin. SCL acts as active or passive clock pin, Sin as data in and Sout as data out (four wire bus). Sin and Sout can be connected together externally to implement three wire bus.

Note:

When the SPI is not used, the three I/O lines (Sin, SCL, Sout) can be used as normal I/O, with the following limitation: bit Sout cannot be used in open drain mode as this enables the shift register output to the port.
It is recommended, in order to avoid spurious interrupts from the SPI, to disable the SPI interrupt (the default state after reset) i.e. no write must be made to the 8-bit shift register (DDh). An explicit interrupt disable may be made in software by a dummy write to address C2h.

Figure 40. SPI Block Diagram



LCD CONTROLLER-DRIVER

The ST6245 LCD driver consists of a LCD control logic, a programmable prescaler, a 12 bytes wide dedicated LCD RAM, 24 segment and 4 common outputs. This allows a direct driving of up to 96 LCD segments.

The LCD driver is managed by the LCD Mode/Control register located at data RAM address DCh. Different display modes (1/1 duty, 1/2 duty, 1/3 duty and 1/4 duty) are available to cover a wide range of application requirements. The multiplexing display modes are software selectable by programming bits 6 and 7 of the LCD control register. Bits 0-5 are used to select the LCD drive and frame frequency (in relation to the system clock) and to switch off all segments. The LCD Driver can also be supplied by the 32kHz real-time oscillator allowing working in low power conditions and performing real time clock operation.

According to the data in the LCD RAM, the segment and the common drivers generate the segment and common signals which can directly drive an LCD panel.

The LCD control logic reads automatically the data from the LCD RAM independently and without interruption of the processor. The part of the LCD RAM that is not used for displaying can be used as normal data memory.

The scale factor of the clock prescaler can be fixed by software, therefore different frame frequencies can be defined.

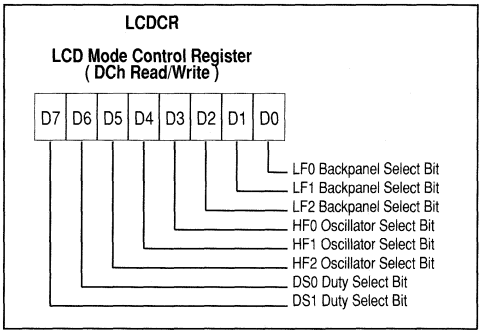
The ST6245 oscillator should operate with a 1.0486, 2.0972, 4.1943, 8.3886MHz frequency quartz crystal. This allows the associated division rates to achieve an internal reference frequency of 32.768kHz. The different division rates can be achieved by programming bits 3, 4, 5 in the LCD control register (see Table 11). It is not recommended to select an internal frequency lower than 32.768kHz as the clock supervisor circuit may switch off the LCD peripheral if the lower frequency is detected.

When the display is turned off, all segment and common outputs are switched to ground, causing all the segments to be switched off regardless of the contents of the LCD RAM.

When the Stand-by oscillator function is selected, the 32kHz stand-by oscillator is selected as clock source for the LCD.

To avoid incomplete frames of the LCD, the mode control bits do not immediately influence the LCD controller when the LCD control register is written. They are stored in a temporary register and change the LCD function only at the end of the frame. Special care must be taken when entering the STOP mode. After switching the LCD clock source from the main oscillator to the 32kHz standby oscillator it must be guaranteed that enough clock pulses are delivered to complete the current frame before entering the STOP mode. Otherwise the LCD function will not be changed and the LCD will be switched OFF after entering the STOP mode. Different LCD frame frequencies for each display mode are selected by bits in the LCD control register (see Table 12).

Figure 41. LCD Mode Control Register



DS0, DS1. Duty cycle select bits. These bits select the number of common backplanes used by the LCD control. This allows different multiplexing conditions.

HF0, HF1, HF2. These bits allow the LCD controller to be supplied with the correct frequency when different high main oscillator frequencies are selected as system clock. Table 11 shows the set-up for different clock crystals.

LF0, LF1, LF2. These bits control the LCD base operational frequency of the LCD common lines. Table 12 shows the set-up to select the different

Table 10. Duty Cycle Selection

DS1	DS0	Display Mode	Active Blackplanes	Max. Number of Segments Driven
0	0	1/4 duty	COM1, 2, 3, 4	180
0	1	1/1 duty	COM1	45
1	0	1/2 duty	COM1, 2	90
1	1	1/3 duty	COM1, 2, 3	135

LCD CONTROLLER-DRIVER (Continued)

Table 11. High Frequency Select Bits

HF2	HF1	HF0	Function	fosc
0	0	0	Display off	
0	0	1	for stand-by Oscillator	32.768kHz
0	1	0	NOT TO BE USED	
0	1	1	÷ 32 for main oscillator	1.048MHz
1	0	0	÷ 64 for main oscillator	2.097MHz
1	0	1	÷ 128 for main oscillator	4.194MHz
1	1	0	÷ 256 for main oscillator	8.388MHz
1	1	1	NOT TO BE USED	

Notes :

1. The usage f_{osc} values different from those defined in this table cause the LCD to operate at a reference frequency different from 32.768kHz.
2. It is not recommended to select an internal frequency lower than 32.768kHz as the clock supervisor circuit may switch off the LCD peripheral if lower frequency is detected.

Table 12. LCD Frequency Select Bits

LF2	LF1	LF0	f_{LCD} (Hz)
0	0	0	64
0	0	1	85
0	1	0	128
0	1	1	171
1	0	0	256
1	0	1	341
1	1	0	512
1	1	1	Not to be Used

Table 13. Available Frame Frequencies for LCD

f_{LCD} (Hz)	Frame Frequency f_F (Hz)			
	1/1 duty	1/2 duty	1/3 duty	1/4 duty
512	512	256	171	128
341	341	171	114	85
256	256	128	85	64
171	171	85	57	43
128	128	64	43	32
85	85	43	28	21
64	64	32	21	16

frequencies while Table 13 shows the corresponding frame values with the different multiplexing conditions.

According to the selected LCD drive frequency f_{LCD} the frame frequencies come out as shown in Table 13.

The Figure 48 illustrates the waveforms of the different duty signals.

The value of the VLCD voltage can be chosen independently from V_{DD} according to the display requirements. The intermediate VLCD levels 2/3 VLCD, 1/3 VLCD and 1/2 VLCD are generated by an internal resistor network as shown in Figures 46 and 47. The half VLCD level for 1/2 duty cycle is obtained by the external connection of VLCD1/3 and VLCD2/3 pins. All intermediate VLCD levels are connected to pins to enable external capacitive buffering or resistive shunting.

LCD CONTROLLER-DRIVER (Continued)

The internal resistive divider network is realized with two parallel dividers. One has high resistivity, the other one low resistivity. The high resistive divider (R_H) is permanently switched on during the LCD operation. The low resistive divider (R_L) is only switched on for a short period of time when the levels of common lines and segment lines are changed. This method combines low source impedance for fast switching of the LCD pixels with high source impedance for low power consumption. Figure 42 shows the typical current into V_{LCD} pin in dependency of the display voltage V_{LCD}.

When the display is switched off (by program or reset) the internal resistor network is also switched off to achieve minimum power consumption. The low resistivity divider is active at each edge of f_{LCD} during 8 clock cycles of F_{32kHz}.

The internal resistor network is implemented with resistive transistor elements to achieve high precision. For display voltages $V_{LCD} < 4.5V$ the resistivity of the divider may be too high for some applications (especially using 1/3 or 1/4 duty display mode). In that case an external resistive divider must be used to achieve the desired resistivity.

Figure 42. Typical Current Consumption on VLCD Pin (25°C, no load, fLCD= 512 Hz, mux= 1/3 -1/4)

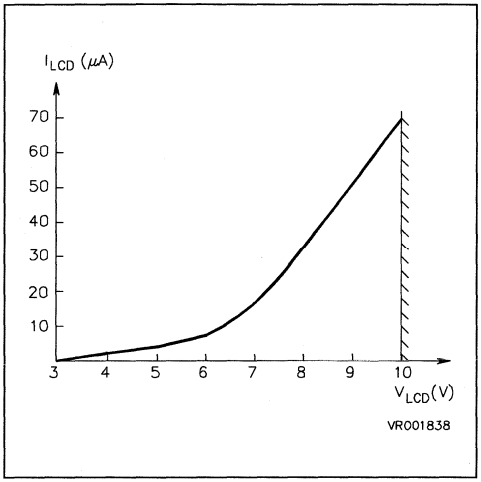
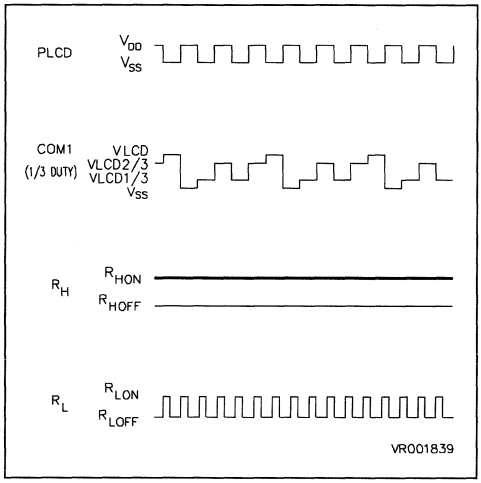


Figure 43. Typical Chronogram of Activation of the V_{LCD} Divider Network



LCD CONTROLLER-DRIVER (Continued)

Typical External resistances values are in the range of 100 kΩ to 150 kΩ. External capacitances in the range of 10 to 47 nF can be added to V_{LCD} 2/3 and V_{LCD} 1/3 pins and to V_{LCD} if the V_{LCD} connection is highly impedant.

When the program is switched off (by program or reset) the internal resistor network is also switched off to achieve minimum power consumption.

Figure 44. Typical Network to connect to V_{LCD} pins if $V_{LCD} \leq 4.5V$

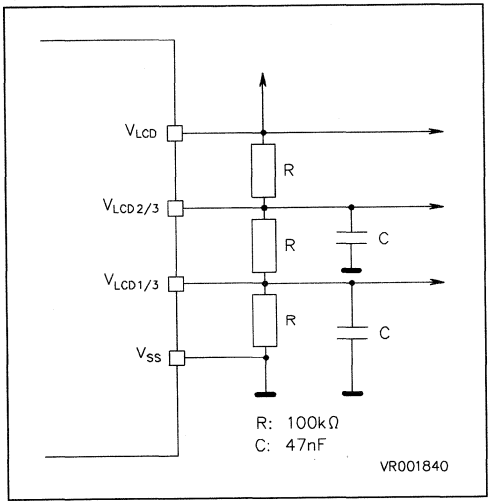


Figure 45. Generation of the 32kHz clock

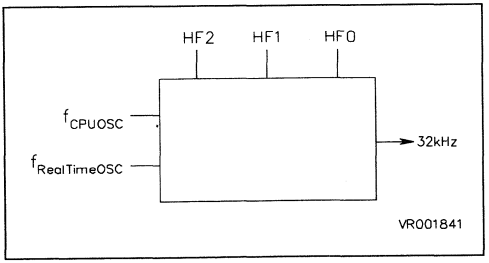


Figure 46. Bias Config for 1/2 Duty

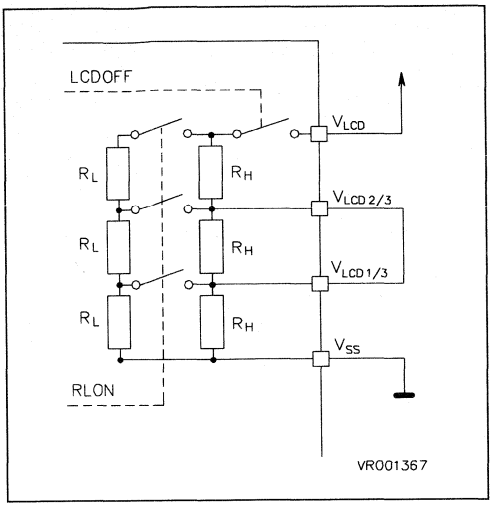
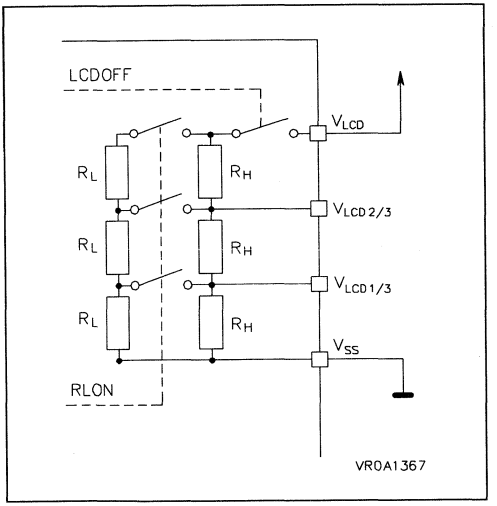


Figure 47. Bias Configuration for 1/1, 1/3 and 1/4 Duty Operation of LCD



LCD CONTROLLER-DRIVER (Continued)

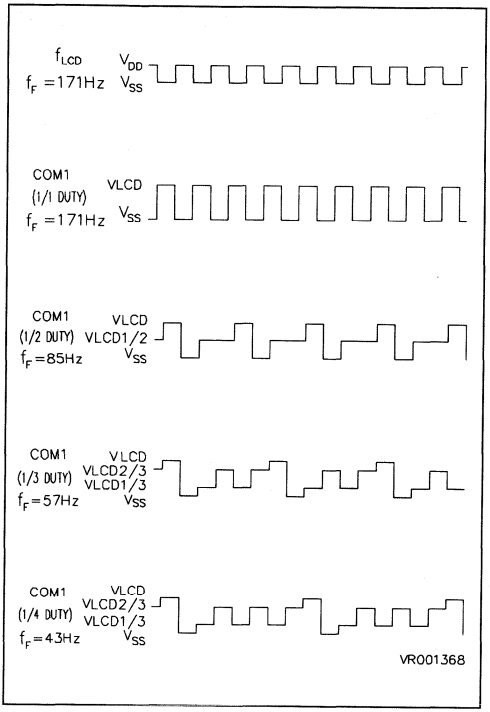
Address Mapping of the Display Segments.

The LCD RAM is located in the ST6245 data space from addresses E0h to F7h. The LCD forms a matrix of 24 segment lines (rows) and up to 4 common lines (columns). Each bit of the LCD RAM is mapped to one element of the LCD matrix, as described in Figure 48. If a bit is set, the corresponding LCD segment is switched on; if it is reset, the segment is switched off. The segments outputs S1, S2 and S3 are not connected to any pin.

When multiplex rates lower than 1/4 are selected, the unused LCD RAM is free for general use. In the 1/2 duty mode, for instance, half of the LCD RAM is available for storing general purpose data. The address range from F8h to FEh can be used as general purpose data RAM, but not for displaying data (it is reserved for future LCD expansion).

After a reset, the LCD RAM is not initializated and contains arbitrary information. As the LCD control register is reset, the LCD is completely switched off.

Figure 48. Common Signal Waveforms



LCD CONTROLLER-DRIVER (Continued)

Figure 49. Addressing Map of the LCD RAM

Data RAM Address	MSB							LSB	
E0*									
E1*									
E2	S24	S23	S22	S21	S20	S19	S18	S17	COM1
E3	S32	S31	S30	S29	S28	S27	S26	S25	
E4	S40	S39	S38	S37	S36	S35	S34	S33	
E5*									
E6*									
E7*									
E8	S24	S23	S22	S21	S20	S19	S18	S17	COM2
E9	S32	S31	S30	S29	S28	S27	S26	S25	
EA	S40	S39	S38	S37	S36	S35	S34	S33	
EB*									
EC*									
ED*									
EE	S24	S23	S22	S21	S20	S19	S18	S17	COM3
EF	S32	S31	S30	S29	S28	S27	S26	S25	
F0	S40	S39	S38	S37	S36	S35	S34	S33	
F1*									
F2*									
F3*									
F4	S24	S23	S22	S21	S20	S19	S18	S17	COM4
F5	S32	S31	S30	S29	S28	S27	S26	S25	
F6	S40	S39	S38	S37	S36	S35	S34	S33	
F7*									
F8 - FE*									

Note: *. Row to be used as general purpose data RAM (not for display data)

Notes:

In STOP mode no clock is available for the LCD controller from the main oscillator. If the 32kHz oscillator is activated the LCD can also operate in STOP mode. If the stand-by oscillator is not active, the LCD controller is switched off when STOP instruction is executed; this mode has to be selected to reach the lowest power consumption.

A missing LCD clock (no oscillator active, broken crystal, etc.) is detected by a clock supervisor circuit that switches all the segments and common lines to ground to avoid destructive DC levels at the LCD.

The LCD function change is only effective at the end of a frame. For this reason special care has to be taken when entering the STOP mode. After switching the LCD clock source from the main oscillator to the 32kHz stand-by oscillator it must be guaranteed that enough clock pulses are delivered to complete the current frame before entering the STOP mode. Otherwise the LCD function will not be changed and the LCD is switched off after entering the STOP mode.

The RAM address F8-FEh are not used for LCD display purposes. So they are available as 7 additional Data RAM registers.

SOFTWARE DESCRIPTION

The ST62xx software has been designed to fully use the hardware in the most efficient way possible while keeping byte usage to a minimum, in short to provide byte efficient programming capability. The ST62xx core has the ability to set or clear any register or RAM location bit of the Data space with a single instruction. Furthermore, the program may branch to a selected address depending on the status of any bit of the Data space. The carry bit is stored with the value of the bit when the SET or RES instruction is processed.

Addressing Modes

The ST62xx core has nine addressing modes which are described in the following paragraphs. The ST62xx core uses three different address spaces: Program space, Data space, and Stack space. Program space contains the instructions which are to be executed, plus the data for immediate mode instructions. Data space contains the Accumulator, the X,Y,V and W registers, peripheral and Input/Output registers, the RAM locations and Data ROM locations (for storage of tables and constants). Stack space contains six 12-bit RAM cells used to stack the return addresses for subroutines and interrupts.

Immediate. In the immediate addressing mode, the operand of the instruction follows the opcode location. As the operand is a ROM byte, the immediate addressing mode is used to access constants which do not change during program execution (e.g., a constant used to initialize a loop counter).

Direct. In the direct addressing mode, the address of the byte that is processed by the instruction is stored in the location that follows the opcode. Direct addressing allows the user to directly address the 256 bytes in Data Space memory with a single two-byte instruction.

Short Direct. The core can address the four RAM registers X,Y,V,W (locations 80h, 81h, 82h, 83h) in the short-direct addressing mode . In this case, the instruction is only one byte and the selection of the location to be processed is contained in the opcode. Short direct addressing is a subset of the direct addressing mode. (Note that 80h and 81h are also indirect registers).

Extended. In the extended addressing mode, the 12-bit address needed to define the instruction is obtained by concatenating the four less significant bits of the opcode with the byte following the opcode. The instructions (JP, CALL) that use the extended addressing mode are able to branch to any address of the 4K bytes Program space.

An extended addressing mode instruction is two-byte long.

Program Counter Relative. The relative addressing mode is only used in conditional branch instructions. The instruction is used to perform a test and, if the condition is true, a branch with a span of -15 to +16 locations around the address of the relative instruction. If the condition is not true, the instruction that follows the relative instruction is executed. The relative addressing mode instruction is one-byte long. The opcode is obtained in adding the three most significant bits that characterize the kind of the test, one bit that determines whether the branch is a forward (when it is 0) or backward (when it is 1) branch and the four less significant bits that give the span of the branch (0h to Fh) that must be added or subtracted to the address of the relative instruction to obtain the address of the branch.

Bit Direct. In the bit direct addressing mode, the bit to be set or cleared is part of the opcode, and the byte following the opcode points to the address of the byte in which the specified bit must be set or cleared. Thus, any bit in the 256 locations of Data space memory can be set or cleared.

Bit Test & Branch. The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit test and branch instruction is three-byte long. The bit identification and the tested condition are included in the opcode byte. The address of the byte to be tested follows immediately the opcode in the Program space. The third byte is the jump displacement, which is in the range of -126 to +129. This displacement can be determined using a label, which is converted by the assembler.

Indirect. In the indirect addressing mode, the byte processed by the register-indirect instruction is at the address pointed by the content of one of the indirect registers, X or Y (80h,81h). The indirect register is selected by the bit 4 of the opcode. A register indirect instruction is one byte long.

Inherent. In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. These instructions are one byte long.

SOFTWARE DESCRIPTION (Continued)

Instruction Set

The ST62xx core has a set of 40 basic instructions. When these instructions are combined with nine addressing modes, 244 usable opcodes can be obtained. They can be divided into six different types:load/store, arithmetic/logic, conditional branch, control instructions, jump/call, bit manipulation. The following paragraphs describe the different types.

All the instructions within a given type are presented in individual tables.

Load & Store. These instructions use one,two or three bytes in relation with the addressing mode. One operand is the Accumulator for LOAD and the other operand is obtained from data memory using one of the addressing modes.

For Load Immediate one operand can be any of the 256 data space bytes while the other is always immediate data.

Table 14. Load & Store Instructions

Instruction	Addressing Mode	Bytes	Cycles	Flags	
				Z	C
LD A, X	Short Direct	1	4	Δ	*
LD A, Y	Short Direct	1	4	Δ	*
LD A, V	Short Direct	1	4	Δ	*
LD A, W	Short Direct	1	4	Δ	*
LD X, A	Short Direct	1	4	Δ	*
LD Y, A	Short Direct	1	4	Δ	*
LD V, A	Short Direct	1	4	Δ	*
LD W, A	Short Direct	1	4	Δ	*
LD A, rr	Direct	2	4	Δ	*
LD rr, A	Direct	2	4	Δ	*
LD A, (X)	Indirect	1	4	Δ	*
LD A, (Y)	Indirect	1	4	Δ	*
LD (X), A	Indirect	1	4	Δ	*
LD (Y), A	Indirect	1	4	Δ	*
LDI A, #N	Immediate	2	4	Δ	*
LDI rr, #N	Immediate	3	4	*	*

Notes:
X,Y. Indirect Register Pointers, V & W Short Direct Registers
#. Immediate data (stored in ROM memory)
rr. Data space register
Δ. Affected
*. Not Affected

SOFTWARE DESCRIPTION (Continued)

Arithmetic and Logic. These instructions are used to perform the arithmetic calculations and logic operations. In AND, ADD, CP, SUB instructions one operand is always the accumulator while the other can be either a data space memory content or an immediate value in relation with the addressing mode. In CLR, DEC, INC instructions the operand can be any of the 256 data space addresses. In COM, RLC, SLA the operand is always the accumulator.

Table 15. Arithmetic & Logic Instructions

Instruction	Addressing Mode	Bytes	Cycles	Flags	
				Z	C
ADD A, (X)	Indirect	1	4	Δ	Δ
ADD A, (Y)	Indirect	1	4	Δ	Δ
ADD A, rr	Direct	2	4	Δ	Δ
ADDI A, #N	Immediate	2	4	Δ	Δ
AND A, (X)	Indirect	1	4	Δ	*
AND A, (Y)	Indirect	1	4	Δ	*
AND A, rr	Direct	2	4	Δ	*
ANDI A, #N	Immediate	2	4	Δ	*
CLR A	Short Direct	2	4	Δ	Δ
CLR rr	Direct	3	4	*	*
COM A	Inherent	1	4	Δ	Δ
CP A, (X)	Indirect	1	4	Δ	Δ
CP A, (Y)	Indirect	1	4	Δ	Δ
CP A, rr	Direct	2	4	Δ	Δ
CPI A, #N	Immediate	2	4	Δ	Δ
DEC X	Short Direct	1	4	Δ	*
DEC Y	Short Direct	1	4	Δ	*
DEC V	Short Direct	1	4	Δ	*
DEC W	Short Direct	1	4	Δ	*
DEC A	Direct	2	4	Δ	*
DEC rr	Direct	2	4	Δ	*
DEC (X)	Indirect	1	4	Δ	*
DEC (Y)	Indirect	1	4	Δ	*
INC X	Short Direct	1	4	Δ	*
INC Y	Short Direct	1	4	Δ	*
INC V	Short Direct	1	4	Δ	*
INC W	Short Direct	1	4	Δ	*
INC A	Direct	2	4	Δ	*
INC rr	Direct	2	4	Δ	*
INC (X)	Indirect	1	4	Δ	*
INC (Y)	Indirect	1	4	Δ	*
RLC A	Inherent	1	4	Δ	Δ
SLA A	Inherent	2	4	Δ	Δ
SUB A, (X)	Indirect	1	4	Δ	Δ
SUB A, (Y)	Indirect	1	4	Δ	Δ
SUB A, rr	Direct	2	4	Δ	Δ
SUBI A, #N	Immediate	2	4	Δ	Δ

Notes:
X,Y. Indirect Register Pointers, V & W Short Direct Registers
. Immediate data (stored in ROM memory)
rr. Data space register
Δ. Affected
*. Not Affected

SOFTWARE DESCRIPTION (Continued)

Conditional Branch. The branch instructions achieve a branch in the program when the selected condition is met.

Bit Manipulation Instructions. These instructions can handle any bit in data space memory. One group either sets or clears. The other group (see Conditional Branch) performs the bit test branch operations.

Control Instructions. The control instructions control the MCU operations during program execution.

Jump and Call. These two instructions are used to perform long (12-bit) jumps or subroutines call inside the whole program space.

Table 16. Conditional Branch Instructions

Instruction	Branch If	Bytes	Cycles	Flags	
				Z	C
JRC e	C = 1	1	2	*	*
JRNC e	C = 0	1	2	*	*
JRZ e	Z = 1	1	2	*	*
JRNZ e	Z = 0	1	2	*	*
JRR b, rr, ee	Bit = 0	3	5	*	Δ
JRS b, rr, ee	Bit = 1	3	5	*	Δ

Notes:
b. 3-bit address
e. 5 bit signed displacement in the range -15 to +16
ee. 8 bit signed displacement in the range -126 to +129
rr. Data space register
Δ. Affected
*. Not Affected

Table 17. Bit Manipulation Instructions

Instruction	Addressing Mode	Bytes	Cycles	Flags	
				Z	C
SET b,rr	Bit Direct	2	4	*	*
RES b,rr	Bit Direct	2	4	*	*

Notes:
b. 3-bit address;
rr. Data space register;
*. Not Affected

Table 18. Control Instructions

Instruction	Addressing Mode	Bytes	Cycles	Flags	
				Z	C
NOP	Inherent	1	2	*	*
RET	Inherent	1	2	*	*
RETI	Inherent	1	2	Δ	Δ
STOP (1)	Inherent	1	2	*	*
WAIT	Inherent	1	2	*	*

Notes:
1. This instruction is deactivated and a WAIT is automatically executed instead of a STOP if the Watchdog function is selected.
Δ. Affected
*. Not Affected

Table 19. Jump & Call Instructions

Instruction	Addressing Mode	Bytes	Cycles	Flags	
				Z	C
CALL abc	Extended	2	4	*	*
JP abc	Extended	2	4	*	*

Notes:
abc.12-bit address;
*. Not Affected

SOFTWARE DESCRIPTION (Continued)

Opcode Map Summary. The following table contains an opcode map for the instructions used on the MCU.

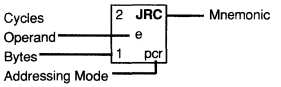
LOW / HI	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111	8 1000	9 1001	A 1010	B 1011	C 1100	D 1101	E 1110	F 1111	LOW / HI
0 0000	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b0,rr,ee 3 bt	2 JRZ e 1 pcr	#	2 JRC e 1 prc	4 LD a,(x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b0,rr 2 b.d	2 JRZ e 1 pcr	4 LDI rr,nn 3 imm	2 JRC e 1 pcr	4 LD a,(y) 1 ind	0 0000
1 0001	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b0,rr,ee 3 bt	2 JRZ e 1 pcr	4 INC x 1 sd	2 JRC e 1 prc	4 LDI a,nn 2 imm	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b0,rr 2 b.d	2 JRZ e 1 pcr	4 DEC x 1 sd	2 JRC e 1 pcr	4 LD a,rr 2 dir	1 0001
2 0010	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b4,rr,ee 3 bt	2 JRZ e 1 pcr	#	2 JRC e 1 prc	4 CP a,(x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b4,rr 2 b.d	2 JRZ e 1 pcr	4 COM a 1 inh	2 JRC e 1 pcr	4 CP a,(y) 1 ind	2 0010
3 0011	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b4,rr,ee 3 bt	2 JRZ e 1 pcr	4 LD a,x 1 sd	2 JRC e 1 prc	4 CPI a,nn 2 imm	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b4,rr 2 b.d.	2 JRZ e 1 pcr	4 LD x,a 1 sd	2 JRC e 1 pcr	4 CP a,rr 2 dir	3 0011
4 0100	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b2,rr,ee 3 bt	2 JRZ e 1 pcr	#	2 JRC e 1 prc	4 ADD a,(x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b2,rr 2 b.d	2 JRZ e 1 pcr	2 RETI 1 inh	2 JRC e 1 pcr	4 ADD a,(y) 1 ind	4 0100
5 0101	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b2,rr,ee 3 bt	2 JRZ e 1 pcr	4 INC y 1 sd	2 JRC e 1 prc	4 ADDI a,nn 2 imm	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b2,rr 2 b.d	2 JRZ e 1 pcr	4 DEC y 1 sd	2 JRC e 1 pcr	4 ADD a,rr 2 dir	5 0101
6 0110	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b6,rr,ee 3 bt	2 JRZ e 1 pcr	#	2 JRC e 1 prc	4 INC (x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b6,rr 2 b.d	2 JRZ e 1 pcr	2 STOP 1 inh	2 JRC e 1 pcr	4 INC (y) 1 ind	6 0110
7 0111	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b6,rr,ee 3 bt	2 JRZ e 1 pcr	4 LD a,y 1 sd	2 JRC e 1 prc	#	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b6,rr 2 b.d	2 JRZ e 1 pcr	4 LD y,a 1 sd	2 JRC e 1 pcr	4 INC rr 2 dir	7 0111
8 1000	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b1,rr,ee 3 bt	2 JRZ e 1 pcr	#	2 JRC e 1 prc	4 LD (x),a 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b1,rr 2 b.d	2 JRZ e 1 pcr	#	2 JRC e 1 pcr	4 LD (y),a 1 ind	8 1000
9 1001	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b1,rr,ee 3 bt	2 JRZ e 1 pcr	4 INC v 1 sd	2 JRC e 1 prc	#	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b1,rr 2 b.d	2 JRZ e 1 pcr	4 DEC v 1 sd	2 JRC e 1 pcr	4 LD rr,a 2 dir	9 1001
A 1010	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b5,rr,ee 3 bt	2 JRZ e 1 pcr	#	2 JRC e 1 prc	4 AND a,(x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b5,rr 2 b.d	2 JRZ e 1 pcr	4 RLC a 1 inh	2 JRC e 1 pcr	4 AND a,(y) 1 ind	A 1010
B 1011	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b5,rr,ee 3 bt	2 JRZ e 1 pcr	4 LD a,v 1 sd	2 JRC e 1 prc	4 ANDI a,nn 2 imm	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b5,rr 2 b.d	2 JRZ e 1 pcr	4 LD v,a 1 sd	2 JRC e 1 pcr	4 AND a,rr 2 dir	B 1011
C 1100	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b3,rr,ee 3 bt	2 JRZ e 1 pcr	#	2 JRC e 1 prc	4 SUB a,(x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b3,rr 2 b.d	2 JRZ e 1 pcr	2 RET 1 inh	2 JRC e 1 pcr	4 SUB a,(y) 1 ind	C 1100
D 1101	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b3,rr,ee 3 bt	2 JRZ e 1 pcr	4 INC w 1 sd	2 JRC e 1 prc	4 SUBI a,nn 2 imm	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b3,rr 2 b.d	2 JRZ e 1 pcr	4 DEC w 1 sd	2 JRC e 1 pcr	4 SUB a,rr 2 dir	D 1101
E 1110	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b7,rr,ee 3 bt	2 JRZ e 1 pcr	#	2 JRC e 1 prc	4 DEC (x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b7,rr 2 b.d	2 JRZ e 1 pcr	2 WAIT 1 inh	2 JRC e 1 pcr	4 DEC (y) 1 ind	E 1110
F 1111	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b7,rr,ee 3 bt	2 JRZ e 1 pcr	4 LD a,w 1 sd	2 JRC e 1 prc	#	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b7,rr 2 b.d	2 JRZ e 1 pcr	4 LD w,a 1 sd	2 JRC e 1 pcr	4 DEC rr 2 dir	F 1111

Abbreviations for Addressing Modes:

dir	Direct
sd	Short Direct
imm	Immediate
inh	Inherent
ext	Extended
b.d	Bit Direct
bt	Bit Test
pcr	Program Counter Relative
ind	Indirect

Legend:

#	Indicates Illegal Instructions
e	5 Bit Displacement
b	3 Bit Address
rr	1byte dataspace address
nn	1 byte immediate data
abc	12 bit address
ee	8 bit Displacement



ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings

This product contains devices to protect the inputs against damage due to high static voltages, however it is advised to take normal precaution to avoid application of any voltage higher than maximum rated voltages.

For proper operation it is recommended that V_I and V_O must be higher than V_{SS} and smaller than V_{DD}. Reliability is enhanced if unused inputs are connected to an appropriated logic voltage level (V_{DD} or V_{SS}).

Power Considerations. The average chip-junction temperature, Tj, in Celsius can be obtained from:

$$Tj = T_A + PD \times RthJA$$

Where : T_A = Ambient Temperature.

RthJA = Package thermal resistance (junction-to ambient).

PD = Pint + Pport.

Pint = $I_{DD} \times V_{DD}$ (chip internal power).

Pport = Port power dissipation (determinated by the user).

Symbol	Parameter	Value	Unit
V_{DD}	Supply Voltage	-0.3 to 7.0	V
V_{LCD}	Display Voltage	-0.3 to 11.0	V
V_I	Input Voltage	V_{SS} - 0.3 to V_{DD} + 0.3	V
V_O	Output Voltage	V_{SS} - 0.3 to V_{DD} + 0.3	V
I_O	Current Drain per Pin Excluding V_{DD} & V_{SS}	± 10	mA
IV_{DD}	Total Current into V_{DD} (source)	50	mA
IV_{SS}	Total Current out of V_{SS} (sink)	50	mA
Tj	Junction Temperature	150	°C
T_{STG}	Storage Temperature	-60 to 150	°C

Note : Stresses above those listed as "absolute maximum ratings" may cause permanent damage to the device . This is a stress rating only and functional operation of the device at these conditions is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

THERMAL CHARACTERISTIC

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
RthJA	Thermal Resistance	PQFP52		70		°C/W

RECOMMENDED OPERATING CONDITIONS

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
T_A	Operating Temperature	1 Suffix Version 6 Suffix Version	0 -40		70 85	°C
V_{DD}	Operating Supply Voltage		3		6	V
V_{LCD}	Display Voltage		3		10	V
V_{DD}	RAM Retention Voltage		2			V

RECOMMENDED OPERATING CONDITIONS (Continued)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
f_{OSC}	Oscillator Frequency (1)(4)	$V_{DD} \geq 4.5V$ $V_{DD} \geq 3V$	0.01 0.01		8.388 2	MHz
I_{INJ+}	Pin Injection Current (positive) Digital Input (2) Analog Input (3)	V_{DD} = 4.5 to 5.5V			+5	mA
I_{INJ-}	Pin Injection Current (negative) Digital Input (2) Analog Input	V_{DD} = 4.5 to 5.5V			-5	mA

Notes :
1. An oscillator frequency above 1MHz is recommanded for reliable A/D results.
2. A current of ± 5mA can be forced on each pin of the digital section without affecting the functional behaviour of the device. For a positive current injected into one pin, a part of this current (~ 10%) can be expected to flow from the neighbouring pins. A current of -5mA can be forced on one input of the analog section at a time (or -2.5mA for all inputs at a time) without affecting the conversion.
3. If a total current of +1mA is flowing into the single analog channel or if the total current flowing into all the analog inputs is of 1mA, all the conversion is resulting shifted of +1LSB. If a total positive current of +5mA is flowing into the single analog channel or if the total current flowing into all the analog inputs is of 5mA, all the conversion is resulting shifted of +2LSB.
4. Operation below 0.01 MHz is possible but requires increased supply current.

EEPROM INFORMATION

The ST62xx EEPROM single poly process has been specially developed to achieve 300.000 Write/Erase cycles and a 10 years data retention.

DC ELECTRICAL CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
V_{IL}	Input Low Level Voltage	RESET, NMI, TIMER, WDON Pin			$0.3V_{DD}$	V
V_{IH}	Input High Level Voltage	TIMER	$0.80V_{DD}$			V
		RESET, NMI, WDON Pin	$0.70V_{DD}$			V
I_{IL} I_{IH}	Input Leakage Current	RESET Pin V_{DD} =5V $V_{IN} = V_{DD}$ (1) $V_{IN} = V_{DD}$ (2) $V_{IN} = V_{SS}$ (5)			10 1 50	µA mA µA
V_{OL}	Low Level Output Voltage	TIMER, I_{OL} = 5.0mA			$0.2V_{DD}$	V
V_{OH}	High Level Output Voltage	TIMER, I_{OL} = -5.0mA	$0.65V_{DD}$			V
R_{PU}	Pull-up Resistor	V_{IN}=0V V_{DD}=5V WDON - NMI	40	100	200	kΩ
		RESET	200	300	500	kΩ

Notes on next page

DC ELECTRICAL CHARACTERISTICS (Continued)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
I_{IL} I_{IH}	Input Leakage Current	TIMER $V_{IN} = V_{DD}$ or V_{SS}		0.1	1.0	µA
I_{IL} I_{IH}	Input Leakage Current	NMI VDD = 5.V $V_{IN} = V_{SS}$ (5) $V_{IN} = V_{DD}$			100 1.0	µA
I_{IL} I_{IH}	Input Leakage Current	WDON VDD = 5V $V_{IN} = V_{SS}$ (5) $V_{IN} = V_{DD}$			100 1.0	µA
I_{DD}	Supply Current RUN Mode	f_{OSC} = 8MHz, I_{LOAD} = 0mA V_{DD} = 5.5V		4	7	mA
	Supply Current WAIT Mode (4)	f_{OSC} = 8MHz, I_{LOAD} = 0mA V_{DD} = 5.0V		1	3	mA
	Supply Current RESET Mode	f_{OSC} = 8MHz, $V_{RESET} = V_{SS}$		1	7	mA
	Supply Current STOP Mode (3)	I_{LOAD} = 0mA V_{DD} = 5.5V		1	10	µA

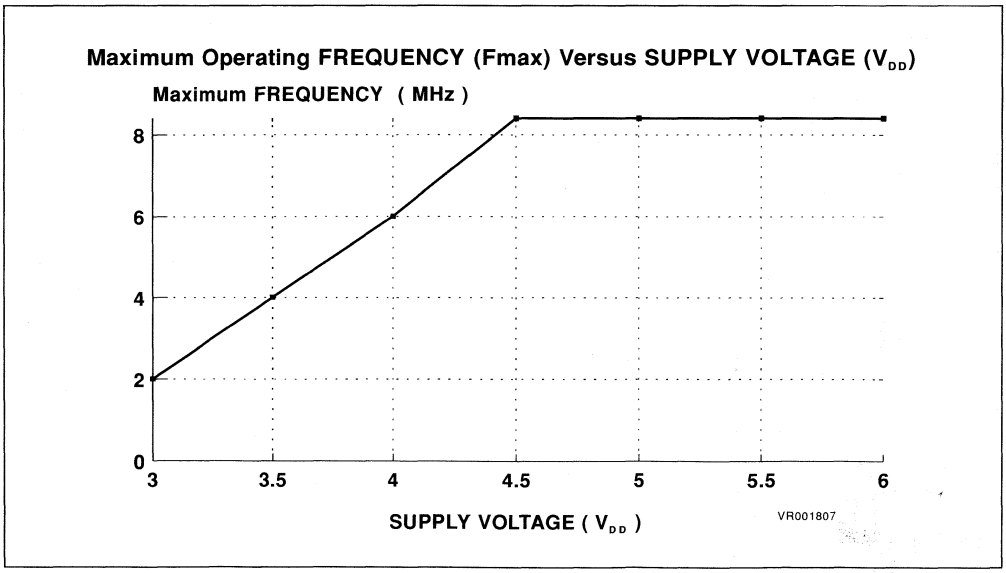
Notes :
1. No Watchdog Reset activated.
2. Reset generated by Watchdog.
3. When the watchdog function is actvated the STOP instruction is deactivated. WAIT instruction is automatically executed.
4. All on-chip peripherals in OFF state
5. Pull-up resistor

AC ELECTRICAL CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
f_{OSC}	Oscillator Frequency (2)	$V_{DD} \geq 4.5V$ $V_{DD} \geq 3V$	0.01		8.388 2	MHz
t_{SU}	Oscillator Start-up Time	$C_{L1} = C_{L2}$ = 22pF - crystal		5	10	ms
t_{SR}	Supply Rise Time	10% to 90%	0.01		100	
t_{REC}	Supply Recovery Time (1)		100			
T_W	Minimum Pulse Width	NMI Pin V_{DD} = 5V	100			ns
		RESET Pin	100			ns
T_{WEE}	EEPROM Write Time	T_A = 25°C One Byte T_A = 85°C One Byte		5 15	10 25	ms
Endurance	EEPROM WRITE/ERASE Cycles	Q_A L_{OT} Acceptance Criteria	300.000	> 1 million		cycles
Retention	EEPROM Data Retention	T_A = 55°C	10			years
C_{IN}	Input Capacitance	All Inputs Pins			10	pF

Notes:
1. Period for which V_{DD} has to be connected or at 0V to allow internal Reset function at next power-up.
2. Operation below 0.01 MHz is possible but requires increased supply current.



I/O PORTS
(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
C_{OUT}	Output Capacitance	All Outputs Pins			10	pF
V_{IL}	Input Low Level Voltage	I/O Pins			$0.3V_{DD}$	V
V_{IH}	Input High Level Voltage	I/O Pins	$0.7V_{DD}$			V
V_{OL}	Low Level Output Voltage	I/O Pins, I_O = 10µA (sink)			0.1	V
		I/O Pins, $I_{OL} = V_{DD}$x1mA V_{DD} = 4.5 to 6V			$0.16xV_{DD}$	V
		I/O Pins, I_{OL} = 1.6mA V_{DD} = 3V			0.4	V
	Low Level Output Voltage, PB4-PB7 Only	I/O Pins, $I_{OL} = V_{DD}$x2mA V_{DD} = 4.5 to 6V			$0.16xV_{DD}$	V
		I/O Pins, I_{OL} = 3.2mA V_{DD} = 3V			0.4	V
		I/O Pins, $I_{OL} = V_{DD}$x4mA V_{DD} = 4.5 to 6V			$0.26xV_{DD}$	V
		I/O Pins, I_{OL} = 6.4mA V_{DD} = 3V			0.8	V
V_{OH}	High Level Output Voltage	I/O Pins, I_O = -10µA (source)	V_{DD}-0.1			V
		I/O Pins, $I_{OL} = -V_{DD}$x1mA V_{DD} = 5.0V	$0.6xV_{DD}$			V
I_{IL} I_{IH}	Input Leakage Current	I/O Pins,(1)		0.1	1.0	µA
R_{PU}	Pull-up Resistor	I/O Pins V_{IN} = 0V, V_{DD} = 5.0V	40	100	200	KΩ

Note 1. Pull-up resistor off

SPI ELECTRICAL CHARACTERISTICS
(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
F_{CL}	Clock Frequency	applied on PB5/SCL			500	kHz
t_{SU}	Set-up Time	applied on PB6/Sin		50		ns
t_h	Hold Time	applied on PB6/Sin		100		ns

A/D CONVERTER CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
Res	Resolution (3)			8		Bit
A_{TOT}	Total Accuracy (3)	f_{OSC} > 1.2 MHz f_{OSC} > 32kHz			± 2 ±4	LSB
t_C[1]	Conversion Time	f_{OSC} = 8MHz		70		µs
V_{AN}	Conversion Range		V_{SS}		V_{DD}	V
ZIR	Zero Input Reading	Conversion result when $V_{IN} = V_{SS}$	00			Hex
FSR	Full Scale Reading	Conversion result when $V_{IN} = V_{DD}$			FF	Hex
AD_I	Analog Input Current During Conversion	V_{DD}= 4.5V			1.0	µA
AC_{IN}[2]	Analog Input Capacitance			2	5	pF
ASI	Analog Source Impedance				30	kΩ
SSI	Analog Reference Supply Impedence				2	kΩ

Notes:
1. With oscillator frequencies less than 1MHz, the A/D Converter accuracy is decreased.
2. Excluding Pad Capacitance
3. Noise at V_{DD},V_{SS} ≤ 10mV

TIMER CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
t_{RES}	Resolution		$\frac{12}{fosc}$			second
f_{IN}	Input Frequency on TIMER Pin				$\frac{fosc}{8}$	MHz
t_W	Pulse Width at TIMER Pin	$V_{DD} \geq 3V$ $V_{DD} \geq 4.5V$	1 125			µs ns

LCD ELECTRICAL CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

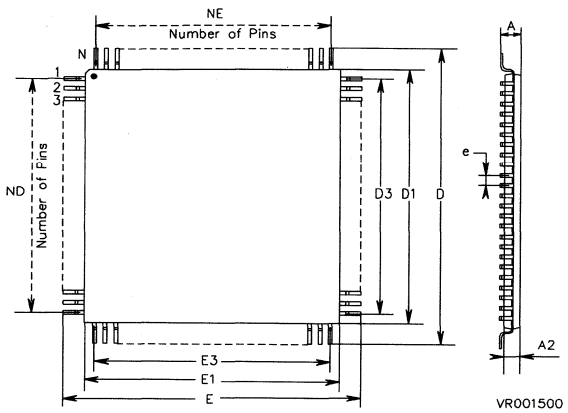
Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
f_{FR}	Frame Frequency	1/4 Duty f_{OSC} = 1, 2, 4, 8MHz	16		128	Hz
V_{OS}	DC Offset Voltage(1)	$V_{LCD} = V_{DD}$, no load			50	mV
V_{OH}	COM High Level, Output Voltage	I = 100µA, V_{LCD} = 5V	4.5V			V
V_{OL}	COM Low Level, Output Voltage	I = 100µA, V_{LCD} = 5V			0.5V	V
V_{OH}	SEG High Level, Output Voltage	I = 50µA, V_{LCD} = 5V	4.5V			V
V_{OL}	SEG Low Level, Output Voltage	I = 50µA, V_{LCD} = 5V			0.5V	V
V_{LCD}	Display Voltage	Note 2	3		10	V

Notes :

1. The DC offset voltage refers to all segment and common outputs. It is the difference between the measured voltage value and nominal value for every voltage level. Ri of voltage meter must be greater than or equal to 100MΩ.
2. An external resistances network is required when $V_{LCD} \leq 4.5V$.

PACKAGE MECHANICAL DATA

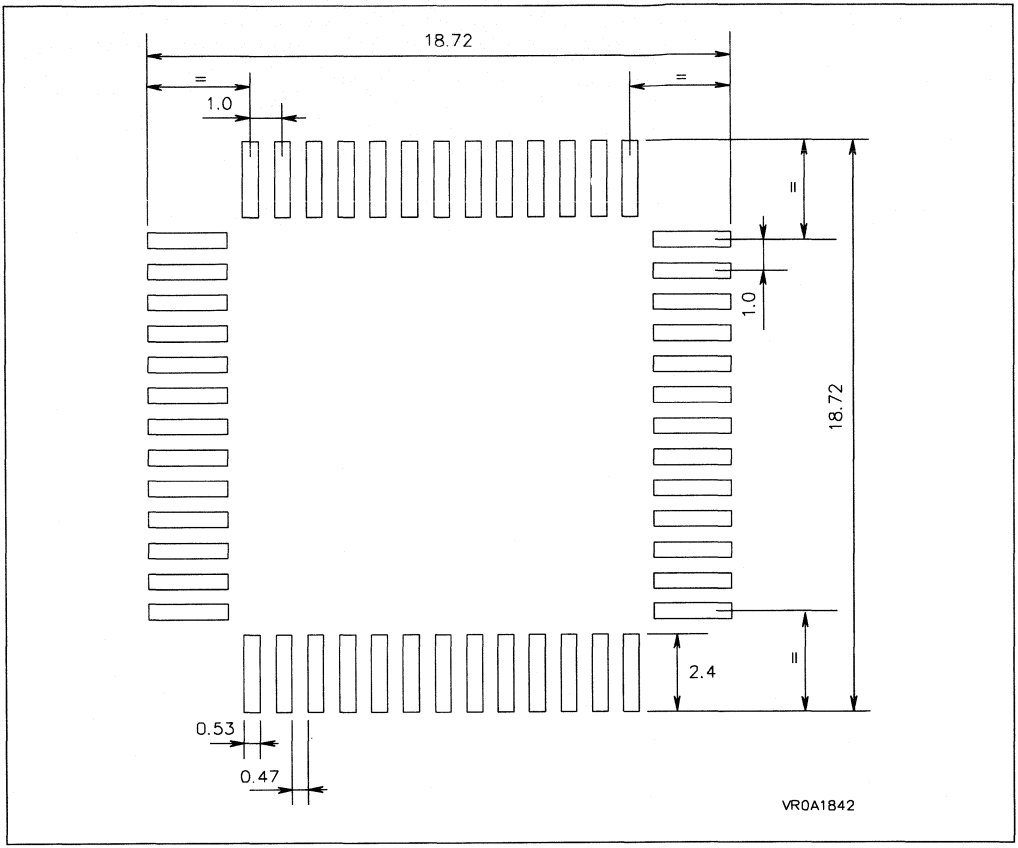
Figure 50. ST6245 52 Pin Plastic Quad Flat Pack Package



Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A			3.40			0.134
A2	2.55	2.80	3.05	0.100	0.110	0.120
D	16.95	17.20	17.45	0.667	0.677	0.687
D1	13.90	14.00	14.10	0.547	0.551	0.555
D3		12.00			0.472	
E	16.95	17.20	17.45	0.667	0.677	0.687
E1	13.90	14.00	14.10	0.547	0.551	0.555
E3		12.00			0.472	
e		1.00			0.039	
Number of Pins						
ND	13					
NE	13					
N	52					

PACKAGE MECHANICAL DATA (Continued)

Recommanded Solder Pad Footprint For QFP52 (in mm)



ORDERING INFORMATION

The following chapter deals with the procedure for transfer the Program/Data ROM codes to SGS-THOMSON.

Communication of the ROM Codes. To communicate the contents of Program/Data ROM memories to SGS-THOMSON, the customer has to send :

- one file in INTEL INTELLEC 8/MDS FORMAT (in an MS-DOS 5" diskette) for the PROGRAM Memory
- one file in INTEL INTELLEC 8/MDS FORMAT (in a MS-DOS 5" diskette) for the EEPROM initial content (this file is optional)
- a filled Option List form as described in the OPTION LIST paragraph.

The program ROM should respect the ROM Memory Map as in Table 22.

The ROM code must be generated with ST6 assembler. Before programming the EPROM, the buffer of the EPROM programmer must be filled with FFh.

Table 22. ROM Memory Map

ROM Page	Device Address	Description
Page 0	0000h-007Fh 0080h-07FFh	Reserved User ROM
Page 1 "STATIC"	0800h-0F9Fh 0FA0h-0FEFh 0FF0h-0FF7h 0FF8h-0FFBh 0FFCh-0FFDh 0FFEh-0FFFh	User ROM Reserved Interrupt Vectors Reserved NMI Vector Reset Vector

Note : EPROM addresses are related to the ROM file to be processed.

Customer EEPROM Initial Contents : Format

a. The content should be written into an INTEL INTELLEC format file.

b. In the case of 128 bytes of EEPROM, the starting address in 000h and the end in 7Fh.

c. Undefined or don't care bytes should have the content FFh.

Listing Generation & Verification. When SGS-THOMSON receives the Codes, they are compared and a computer listing is generated from them. This listing refers exactly to the mask that will be used to produce the microcontroller. Then the listing is returned to the customer that must thoroughly check, complete, sign and return it to SGS-THOMSON. The signed list constitutes a part of the contractual agreement for the creation of the customer mask. SGS-THOMSON sales organization will provide detailed information on contractual points.

ORDERING INFORMATION TABLE

Sales Types	Memory Type	Temperature Range	Package
ST6245Q1/XX	4K ROM 64 bytes EEPROM	0 to + 70°C	PQFP52
ST6245Q6/XX	4K ROM 64 bytes EEPROM	-40 to + 85°C	PQFP52

Note : "XX" is the ROM code identifier allocated by SGS-THOMSON after receipt of all required options and the related ROM file.

ST6245 MICROCONTROLLER OPTION LIST

Customer . . . . . .

Address . . . . . .

Contact . . . . . .

Phone No . . . . . .

Reference . . . . . .

SGS-THOMSON Microelectronics references

Device	[ ] ST6245	
Package	[ ] Plastic Quad Flat Package	
Temperature Range	[ ] 0°C to + 70°C	[ ] -40°C to + 85°C

Special Marking [ ] No
[ ] Yes "_ _ _ _ _ _ _ _ _ _"

Authorized characters are Letters, digits, '.', '–', '/' and spaces only.
For marking one line with 10 characters maximum is possible.

Comments :
- Number of LCD segments used :
- Number of LCD backplanes used :

Note :

Signature

Date

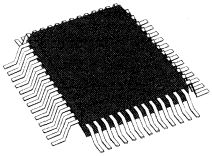
SGS-THOMSON MICROELECTRONICS

ST62E45
ST62T45

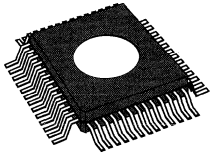
8-BIT EPROM HCMOS MCU WITH LCD DRIVER, EEPROM AND A/D CONVERTER

PRELIMINARY DATA

- 3 to 6V supply operating range
- 8.4MHz Maximum Clock Frequency
- Run, Wait & Stop Modes
- 5 different interrupt vectors
- Look-up table capability in EPROM
- User EPROM: 3884 bytes
  Data RAM: 128 bytes
  LCD RAM: 12 bytes
  EEPROM: 64 bytes
- PQFP52 and CQFP52-W packages
- 11 fully software programmable I/O as:
  - Input with/without pull-up resistor
  - Input with interrupt generation
  - Open-Drain or Push-pull outputs
  - Analog Inputs (7 pins)
- 4 I/O lines can sink up to 20mA for direct LED or TRIAC driving and have SPI alternate functions
- Two 8-bit counters and 7-bit programmable prescalers (Timers 1 and 2)
- Software activated digital watchdog
- 8-bit A/D converter with up to 7 analog inputs
- 8-bit synchronous serial peripheral interface (SPI)
- LCD driver with 24 segment outputs, 4 backplane outputs and selectable duty cycle for up to 96 LCD segments direct driving
- 32kHz oscillator for stand-by LCD operation
- One external not maskable interrupt
- 9 powerful addressing modes
- The accumulator, the X, Y, V & W registers, the port and peripherals data & control registers are addressed in the data space as RAM locations.
- The ST62E45 is the EPROM version, ST62T45 is the OTP version, fully compatible with ST6245 ROM version.



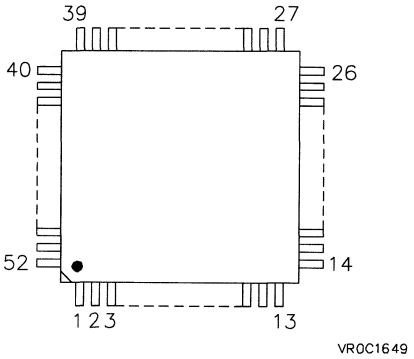
PQFP52



CQFP52-W

(Ordering Information at the end of the datasheet)

Figure 1. 52 Pin Quad Flat Pack (QFP) Package Pinout



ST62E45/T45 Pin Description

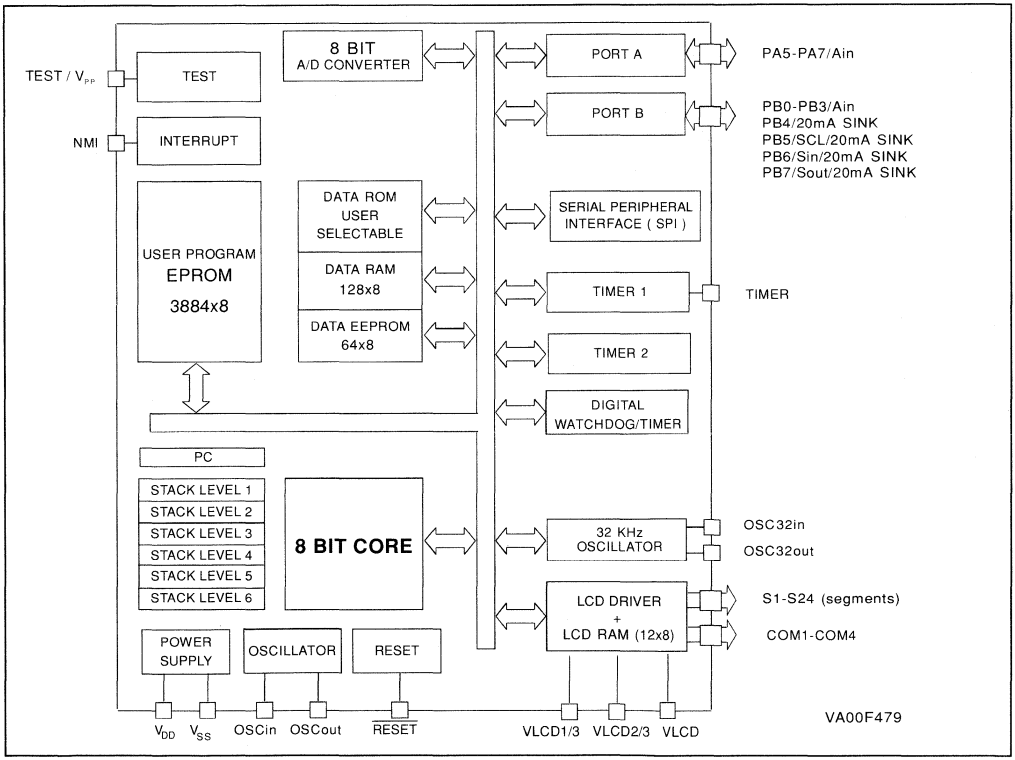
Pin number	Pin name	Pin number	Pin name	Pin number	Pin name	Pin number	Pin name
1	COM4	14	$\overline{\text{RESET}}$	27	OSC32out	40	S12
2	COM3	15	OSCout	28	OSC32in	41	S13
3	COM2	16	OSCin	29	S1	42	S14
4	COM1	17	NMI	30	S2	43	S15
5	VLCD1/3	18	TIMER	31	S3	44	S16
6	VLCD2/3	19	PB7/Sout (1)	32	S4	45	S17
7	VLCD	20	PB6/Sin (1)	33	S5	46	S18
8	PA7/Ain	21	PB5/SCL (1)	34	S6	47	S19
9	PA6/Ain	22	PB4 (1)	35	S7	48	S20
10	PA5/Ain	23	PB3/Ain	36	S8	49	S21
11	TEST	24	PB2/Ain	37	S9	50	S22
12	V_{DD}	25	PB1/Ain	38	S10	51	S23
13	V_{SS}	26	PB0/Ain	39	S11	52	S24

Note 1: 20mA SINK

GENERAL DESCRIPTION

The ST62E45,T45 microcontrollers are members of the 8-bit HCMOS ST62xx family, a series of devices oriented to low-medium complexity applications. They are the EPROM/OTP versions of the ST6245 ROM device and are suitable for product prototyping and low volume production. All ST62xx members are based on a building block approach: a common core is associated with a combination of on-chip peripherals (macrocells). The macrocells of the ST6245 family are: a high performance LCD controller/driver with 24 segment outputs and 4 backplanes able to drive up to 96 segments, two Timer peripherals each including an 8-bit counter with a 7-bit software programmable prescaler (Timer), the digital watchdog timer (DWD), an 8-bit A/D Converter with up to 7 analog inputs and an 8-bit synchronous Serial Peripheral Interface (SPI). In addition these devices offer 64 bytes of EEPROM for storage of non volatile data. Thanks to these peripherals the ST6245 family is well suited for general purpose, automotive, security, appliance and industrial applications.

Figure 2. ST62E45 Block Diagram



Note: Ain = Analog Input

PIN DESCRIPTION

V_{DD} and V_{SS}. Power is supplied to the MCU using these two pins. V_{DD} is power and V_{SS} is the ground connection.

OSCin and OSCout. These pins are internally connected with the on-chip oscillator circuit. A quartz crystal or a ceramic resonator can be connected between these two pins in order to allow the correct operation of the MCU with various stability/cost trade-offs. OSCin is the input pin, OSCout is the output pin. An external clock signal can be applied to OSCin.

$\overline{\text{RESET}}$. The active low $\overline{\text{RESET}}$ pin is used to restart the microcontroller at the beginning of its program. The $\overline{\text{RESET}}$ pin is provided with an on-chip pull-up resistor and schmitt trigger input characteristics.

TEST/V_{PP}. The TEST must be held at V_{SS} for normal operation (an internal pull-down resistor selects normal operating mode if TEST pin is not connected).

NMI. The NMI pin provides the capability for asynchronous applying an external top priority interrupt to the MCU.This pin is falling edge sensitive. The NMI pin is provided with an on-chip pull-up resistor and schmitt trigger input characteristics.

TIMER. This is the TIMER 1 I/O pin. In input mode it is connected to the prescaler and acts as external timer clock or as control gate for the internal timer clock. In the output mode the TIMER pin outputs the data bit when a time-out occurs.

PA5-PA7. These 3 lines are organized as one I/O port (A). Each line may be configured under software control as an input with or without pull-up resistor, interrupt generating input with pull-up resistor, open-drain or push-pull output or as analog input for the A/D converter. Port A has a 5mA drive capability in output mode.

PB0-PB3,PB4-PB7. These 8 lines are organized as one I/O port (B). Each line may be configured under software control as an input with or without internal pull-up resistor, interrupt generating input with pull-up resistor, open-drain or push-pull output. PB0-PB3 can be programmed as analog inputs for the A/D converter while PB4-PB7 can also sink 20mA for direct LED driving. PB5-PB7 can also be used as respectively Clock, Data in and Data out pins for the on-chip SPI to carry the synchronous serial I/O signals.

COM1-COM4. These four pins are the LCD peripheral common outputs. They are the outputs of the on-chip backplane voltage generator which is used for multiplexing the 24 LCD lines allowing up to 96 segments to be driven.

S1-S24. These pins are the 24 LCD peripheral driver outputs of ST62E45. Segments S1-S3 are not connected to any pin.

VLCD. Display voltage supply. It determines the high voltage level on COM1-COM4 and S1-S24 pins.

VLCD1/3, VLCD2/3. Display supply voltage inputs for determining the display voltage levels on COM1-COM4 and S1-S24 pins during multiplex operation.

OSC32in and OSC32out. These pins are internally connected with the on-chip 32kHz oscillator circuit. A 32.768kHz quartz crystal can be connected between these two pins if it is necessary to provide the LCD stand-by clock and real time interrupt. OSC32in is the input pin, OSC32out is the output pin.

ST62E45,T45 EPROM/OTP DESCRIPTION

The ST62E45 is the EPROM version of the ST6245 ROM product. It is intended for use during the development of an application, and for pre-production and small volume production. The ST62T45 OTP has the same characteristics. Both include EPROM memory instead of the ROM memory of the ST6245, and so the program and constants of the program can be easily modified by the user with the ST62E45 EPROM programming board from SGS-THOMSON.

From a user point of view (with the following exception) the ST62E45,T45 products have exactly the same software and hardware features of the ROM version. An additional mode is used to configure the part for programming of the EPROM, this is set by a +12.5V voltage applied to the TEST/V_{PP} pin. The programming of the ST62E45,T45 is described in the User Manual of the EPROM Programming board.

On the ST62E45, all the 3884 bytes of PROGRAM memory are available for the user, as all the EPROM memory can be erased by exposure to UV light. On the ST62T45 (OTP) device) a reserved area for test purposes exists, as for the ST6245 ROM device. In order to avoid any discrepancy between program functionality when using the EPROM, OTP and ROM it is recommended not to use these reserved areas, even when using the ST62E45.

Notes on programming:

In order to emulate exactly the ST6245 features with the ST62E45 and ST6245, some software precautions have to be taken:

1. I/O: To prevent floating input or uncontrolled I/O interrrupt on the EPROM/OTP devices, the port bits PA0-PA4 must be programmed as push-pull outputs.

2. When programming for the EPROM/OTP parts, it is suggested that the conditional assembly technique is used for controlling the I/O ports in order to disable the appropriate code for the ROM device.

3. Do not access data space locations CAh, DAh .

Other than this exception, the ST62E45,T45 parts are fully compatible with the ROM ST6245 equivalent, this datasheet thus provides only information specific to the EPROM based devices.

THE READER IS ASKED TO REFER TO THE DATASHEET OF THE ST6240 ROM-BASED DEVICE FOR FURTHER DETAILS.

EPROM ERASING

The EPROM of the windowed package of the ST62E45 may be erased by exposure to Ultra Violet light.

The erasure characteristic of the ST62E45 EPROM is such that erasure begins when the memory is exposed to light with wave lengths shorter than approximately 4000Å. It should be noted that sunlight and some types of fluorescent lamps have wavelengths in the range 3000-4000Å. It is thus recommended that the window of the ST62E45 package be covered by an opaque label to prevent unintentional erasure problems when testing the application in such an environment.
The recommended erasure procedure of the ST62E45 EPROM is exposure to short wave ultraviolet light which has wavelength 2537Å. The integrated dose (i.e. UV intensity x exposure time) for erasure should be a minimum of 15 W-sec/cm^2. The erasure time with this dosage is approximately 15 to 20 minutes using an ultraviolet lamp with 12000µW/cm^2 power rating. The ST62E45 should be placed within 2.5 cm (1 inch) of the lamp tubes during erasure.

ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings

This product contains devices to protect the inputs against damage due to high static voltages, however it is advised to take normal precaution to avoid application of any voltage higher than maximum rated voltages.

For proper operation it is recommended that V_I and V_O must be higher than V_{SS} and smaller than V_{DD}. Reliability is enhanced if unused inputs are connected to an appropriated logic voltage level (V_{DD} or V_{SS}).

Power Considerations. The average chip-junction temperature, Tj, in Celsius can be obtained from:

$Tj = T_A + PD \times RthJA$

Where : T_A = Ambient Temperature.

RthJA = Package thermal resistance (junction-to ambient).

PD = Pint + Pport.

Pint = $I_{DD} \times V_{DD}$ (chip internal power).

Pport = Port power dissipation (determinated by the user).

Symbol	Parameter	Value	Unit
V_{DD}	Supply Voltage	-0.3 to 7.0	V
V_{LCD}	Display Voltage	-0.3 to 11.0	V
V_I	Input Voltage	V_{SS} - 0.3 to V_{DD} + 0.3	V
V_O	Output Voltage	V_{SS} - 0.3 to V_{DD} + 0.3	V
I_O	Current Drain per Pin Excluding V_{DD} & V_{SS}	± 10	mA
IV_{DD}	Total Current into V_{DD} (source)	50	mA
IV_{SS}	Total Current out of V_{SS} (sink)	50	mA
Tj	Junction Temperature	150	°C
T_{STG}	Storage Temperature	-60 to 150	°C

THERMAL CHARACTERISTIC

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
RthJA	Thermal Resistance	PQFP52 CQFP52-W		70 70		°C/W

RECOMMENDED OPERATING CONDITIONS

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
T_A	Operating Temperature	1 Suffix Version 6 Suffix Version	0 -40		70 85	°C
V_{DD}	Operating Supply Voltage		3		6	V
V_{PP}	EPROM Prog. Voltage		12	12.5	13	V
V_{LCD}	Display Voltage		3		10	V
V_{RM}	RAM Retention Voltage		2			V

RECOMMENDED OPERATING CONDITIONS (Continued)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
f_{OSC}	Oscillator Frequency (1)(4)	$V_{DD} \geq 4.5V$ $V_{DD} \geq 3V$	0.01 0.01		8.388 2	MHz
I_{INJ+}	Pin Injection Current (positive) Digital Input (2) Analog Input (3)	V_{DD} = 4.5 to 5.5V			+5	mA
I_{INJ-}	Pin Injection Current (negative) Digital Input (2) Analog Input	V_{DD} = 4.5 to 5.5V			-5	mA

Notes :

1. An oscillator frequency above 1MHz is recommanded for reliable A/D results.
2. A current of ± 5mA can be forced on each pin of the digital section without affecting the functional behaviour of the device. For a positive current injected into one pin, a part of this current (~ 10%) can be expected to flow from the neighbouring pins. A current of -5mA can be forced on one input of the analog section at a time (or -2.5mA for all inputs at a time) without affecting the conversion.
3. If a total current of +1mA is flowing into the single analog channel or if the total current flowing into all the analog inputs is of 1mA, all the conversion is resulting shifted of +1LSB. If a total positive current of +5mA is flowing into the single analog channel or if the total current flowing into all the analog inputs is of 5mA, all the conversion is resulting shifted of +2LSB.
4. Operation below 0.01 MHz is possible but requires increased supply current.

EEPROM INFORMATION

The ST62xx EEPROM single poly process has been specially developed to achieve 300.000 Write/Erase cycles and a 10 years data retention.

DC ELECTRICAL CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
V_{IL}	Input Low Level Voltage	RESET, NMI, TIMER, WDON Pin			$0.3V_{DD}$	V
V_{IH}	Input High Level Voltage	TIMER	$0.80V_{DD}$			V
		RESET, NMI, WDON Pin	$0.70V_{DD}$			V
I_{IL} I_{IH}	Input Leakage Current	RESET Pin V_{DD} =5V $V_{IN} = V_{DD}$ (1) $V_{IN} = V_{DD}$ (2) $V_{IN} = V_{SS}$ (5)			10 1 50	µA mA µA
V_{OL}	Low Level Output Voltage	TIMER, I_{OL} = 5.0mA			$0.2V_{DD}$	V
V_{OH}	High Level Output Voltage	TIMER, I_{OL} = -5.0mA	$0.65V_{DD}$			V

Notes on next page

DC ELECTRICAL CHARACTERISTICS (Continued)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
R_{PU}	Pull-up Resistor	V_{IN}=0V V_{DD}=5V WDON - NMI	40	100	200	kΩ
		RESET	100	300	500	kΩ
I_{IL} I_{IH}	Input Leakage Current	TIMER $V_{IN} = V_{DD}$ or V_{SS}		0.1	1.0	µA
I_{IL} I_{IH}	Input Leakage Current	NMI VDD=5V $V_{IN} = V_{SS}$ (5) $V_{IN} = V_{DD}$			 100 1.0	µA
I_{IL} I_{IH}	Input Leakage Current	WDON VDD=5V $V_{IN} = V_{SS}$ (5) $V_{IN} = V_{DD}$			 100 1.0	µA
I_{DD}	Supply Current RUN Mode	f_{OSC} = 8MHz, I_{LOAD} = 0mA V_{DD} = 5.5V		4	7	mA
	Supply Current WAIT Mode (4)	f_{OSC} = 8MHz, I_{LOAD} = 0mA V_{DD} = 5.0V		1	3	mA
	Supply Current RESET Mode	f_{OSC} = 8MHz, $V_{RESET} = V_{SS}$		1	7	mA
	Supply Current STOP Mode (3)	I_{LOAD} = 0mA V_{DD} = 5.5V		1	10	µA

Notes :

1. No Watchdog Reset activated.
2. Reset generated by Watchdog.
3. When the watchdog function is actvated the STOP instruction is deactivated. WAIT instruction is automatically executed.
4. All on-chip peripherals in OFF state
5. Pull-up resistor

AC ELECTRICAL CHARACTERISTICS

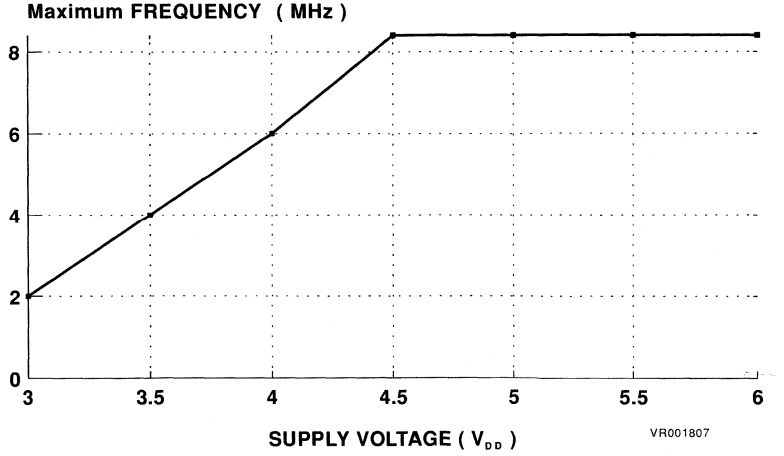
(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
f_{OSC}	Oscillator Frequency (2)	$V_{DD} \geq 4.5V$ $V_{DD} \geq 3V$	0.01 0.01		8.388 2	MHz
t_{SU}	Oscillator Start-up Time	$C_{L1} = C_{L2}$ = 22pF - crystal		5	10	ms
t_{SR}	Supply Rise Time	10% to 90%	0.01		100	
t_{REC}	Supply Recovery Time (1)		100			
T_W	Minimum Pulse Width	NMI Pin V_{DD} = 5V	100			ns
		RESET Pin	100			ns
T_{WEE}	EEPROM Write Time	T_A = 25°C One Byte T_A = 85°C One Byte		5 15	10 25	ms ms
Endurance	EEPROM WRITE/ERASE Cycles	Q_A L_{OT} Acceptance Criteria	300.000	> 1 million		cycles
Retention	EEPROM Data Retention	T_A = 55°C	10			years
C_{IN}	Input Capacitance	All Inputs Pins			10	pF

Notes:

1. Period for which V_{DD} has to be connected or at 0V to allow internal Reset function at next power-up.
2. Operation below 0.01 MHz is possible but requires increased supply current.

Maximum Operating FREQUENCY (Fmax) Versus SUPPLY VOLTAGE (V_{DD})



I/O PORTS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
C_{OUT}	Output Capacitance	All Outputs Pins			10	pF
V_{IL}	Input Low Level Voltage	I/O Pins			$0.3V_{DD}$	V
V_{IH}	Input High Level Voltage	I/O Pins	$0.7V_{DD}$			V
V_{OL}	Low Level Output Voltage	I/O Pins, I_O = 10µA (sink)			0.1	V
		I/O Pins, $I_{OL} = V_{DD}$x1mA V_{DD} = 4.5 to 6V			$0.16xV_{DD}$	V
		I/O Pins, I_{OL} = 1.6mA V_{DD} = 3V			0.4	V
	Low Level Output Voltage, PB4-PB7 Only	I/O Pins, $I_{OL} = V_{DD}$x2mA V_{DD} = 4.5 to 6V			$0.16xV_{DD}$	V
		I/O Pins, I_{OL} = 3.2mA V_{DD} = 3V			0.4	V
		I/O Pins, $I_{OL} = V_{DD}$x4mA V_{DD} = 4.5 to 6V			$0.26xV_{DD}$	V
		I/O Pins, I_{OL} = 6.4mA V_{DD} = 3V			0.8	V
V_{OH}	High Level Output Voltage	I/O Pins, I_O = -10µA (source)	V_{DD}-0.1			V
		I/O Pins, $I_{OL} = -V_{DD}$x1mA V_{DD} = 5.0V	$0.6xV_{DD}$			V
I_{IL} I_{IH}	Input Leakage Current	I/O Pins,(1)		0.1	1.0	µA
R_{PU}	Pull-up Resistor	I/O Pins V_{IN} = 0V, V_{DD} = 5.0V	40	100	200	kΩ

Note 1. Pull-up resistor off

SPI ELECTRICAL CHARACTERISTICS

(T_A = -40 to +85°C unless otherwhise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
F_{CL}	Clock Frequency	applied on PB5/SCL			500	kHz
t_{SU}	Set-up Time	applied on PB6/Sin		50		ns
t_h	Hold Time	applied on PB6/Sin		100		ns

A/D CONVERTER CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
Res	Resolution (3)			8		Bit
A_{TOT}	Total Accuracy (3)	f_{OSC} > 1.2 MHz f_{OSC} > 32kHz			± 2 ±4	LSB
t_C[1]	Conversion Time	f_{OSC} = 8MHz		70		µs
V_{AN}	Conversion Range		V_{SS}		V_{DD}	V
ZIR	Zero Input Reading	Conversion result when $V_{IN} = V_{SS}$	00			Hex
FSR	Full Scale Reading	Conversion result when $V_{IN} = V_{DD}$			FF	Hex
AD_I	Analog Input Current During Conversion	V_{DD}= 4.5V			1.0	µA
AC_{IN}[2]	Analog Input Capacitance			2	5	pF
ASI	Analog Source Impedance				30	kΩ
SSI	Analog Reference Supply Impedence				2	kΩ

Notes:

1. With oscillator frequencies less than 1MHz, the A/D Converter accuracy is decreased.
2. Excluding Pad Capacitance
3. Noise at V_{DD},$V_{SS} \leq$ 10mV

TIMER CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
t_{RES}	Resolution		$\frac{12}{f_{OSC}}$			second
f_{IN}	Input Frequency on TIMER Pin				$\frac{f_{OSC}}{8}$	MHz
t_W	Pulse Width at TIMER Pin	$V_{DD} \geq 3V$ $V_{DD} \geq 4.5V$	1 125			µs ns

LCD ELECTRICAL CHARACTERISTICS

(T_A = -40 to +85°C unless otherwise specified)

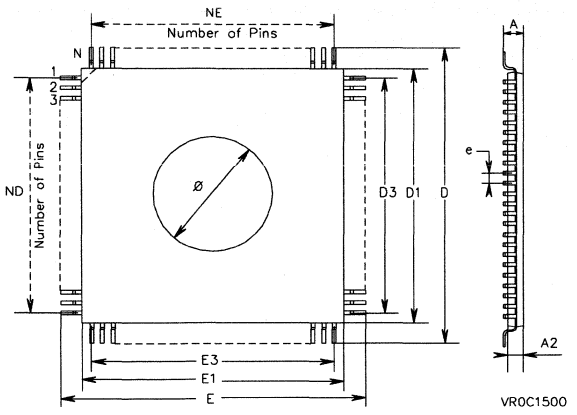
Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
f_{FR}	Frame Frequency	1/4 Duty f_{OSC} = 1, 2, 4, 8MHz	16		128	Hz
V_{OS}	DC Offset Voltage(1)	$V_{LCD} = V_{DD}$, no load			50	mV
V_{OH}	COM High Level, Output Voltage	I = 100µA V_{LCD} = 5V	4.5V			V
V_{OL}	COM Low Level, Output Voltage	I = 100µA V_{LCD} = 5V			0.5V	V
V_{OH}	SEG High Level, Output Voltage	I = 50µA V_{LCD} = 5V	4.5V			V
V_{OL}	SEG Low Level, Output Voltage	I = 50µA V_{LCD} = 5V			0.5V	V
V_{LCD}	Display Voltage	Note 2	3		10	V

Notes :

1. The DC offset voltage refers to all segment and common outputs. It is the difference between the measured voltage value and nominal value for every voltage level. Ri of voltage meter must be greater than or equal to 100MΩ.
2. An external resistances network is required when $V_{LCD} \leq 4.5V$.

PACKAGE MECHANICAL DATA

Figure 3.ST62E45 52 Pin Ceramic Quad Flat Package with Window



Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A		3.55			0.14	
A2		3.40			0.133	
D		17.20			0.705	
D1		14.00			0.551	
D3		12.00			0.472	
E		17.20			0.705	
E1		14.00			0.551	
E3		12.00			0.472	
Ø		7.62			0.3	
e		17.20			0.032	
Number of Pins						
ND	13					
NE	13					
N	52					

ORDERING INFORMATION TABLE

Sales Types	Memory type	Temperature Range	Package
ST62E45G1	4K EPROM	0 to + 70°C	CQFP52-W
ST62T45Q6	4K EPROM	-40 to + 85°C	PQFP52

ST628x DATASHEETS

ST628x DATASHEETS INDEX

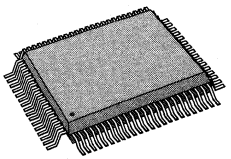
SGS-THOMSON MICROELECTRONICS

ST6280

8-BIT HCMOS MCU WITH DOT MATRIX LCD DRIVER EEPROM AND A/D CONVERTER

PRODUCT OVERVIEW

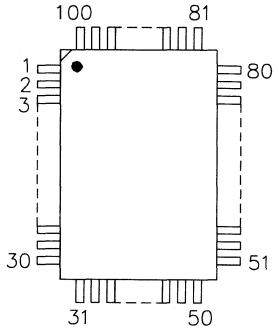
- 4.5 to 5.5V supply operating range
- 8.4 MHz Maximum Clock Frequency
- -40 to +85°C Operating Temperature Range
- Run, Wait & Stop Modes
- 5 different interrupt vectors
- Look-up table capability in ROM
- User ROM: 7948 bytes
  Reserved ROM: 244 bytes
  Data RAM: 192 bytes
  LCD RAM: 128 bytes
  EEPROM: 128 bytes
- PQFP100 package
- 12 fully software programmable I/O as:
  - Input with/without pull-up resistor
  - Input with interrupt generation
  - Open-Drain or Push-pull outputs
  - Analog Inputs
- 10 I/O lines can sink up to 20mA for direct LED or TRIAC driving and have SPI alternate functions
- One 8-bit counter with 7-bit programmable prescalers (Timer 1)
- One 8-bit auto-reload timer with 7-bit programmable prescaler (Timer 2)
- Software activated digital watchdog
- 8-bit A/D converter with up to 12 analog inputs
- 8-bit synchronous serial peripheral interface (SPI)
- LCD driver controller with 48 segments outputs, 8 backplane and 8 software selectable segment/backplane outputs able to drive up to 48x16 (768) or 56x8 (448) segments.
- 32kHz oscillator for stand-by LCD operation
- One external not maskable interrupt
- 9 powerful addressing modes
- The accumulator, the X, Y, V & W registers, the port and peripherals data & control registers are addressed in the data space as RAM locations.
- The ST62E80 is the EPROM version, ST62T80 is the OTP version



PQFP100

(Ordering information at the end of the datasheet)

Figure 1. 100 Pin Quad Flat Pack (QFP) Package Pinout



ST6280 Pin Description

Pin number	Pin name
1	S39
2	S40
3	S41
4	S42
5	S43
6	S44
7	S45
8	S46
9	S47
10	S48
11	S49
12	S50
13	S51
14	S52
15	S53
16	S54
17	S55
18	S56
19	PB7/TIMOUT2 /Ain
20	PB6/TIMIN2 /Ain
21	PB5/Ain
22	PB4/Ain
23	PB3/Ain
24	PB2/Ain
25	PB1/Ain
26	PB0/Ain
27	TEST
28	OSCout
29	OSCin
30	$\overline{\text{RESET}}$

Pin number	Pin name
31	PC7/Ain
32	PC6/Ain
33	PC5/Ain
34	PC4/Ain
35	PC3 (1)
36	PC2 (1)
37	PC1 (1)
38	PC0 (1)
39	NMI
40	V_{DD}
41	V_{SS}
42	VLCD
43	VLCD4/5
44	VLCD3/5
45	VLCD2/5
46	VLCD1/5
47	PA7/Sout (1)
48	PA6/Sin (1)
49	PA5/SCL (1)
50	PA4/TIM1 (1)

Pin number	Pin name
51	PA3 (1)
52	PA2 (1)
53	OSC32out
54	OSC32in
55	COM1
56	COM2
57	COM3
58	COM4
59	COM5
60	COM6
61	COM7
62	COM8
63	COM9/S1
64	COM10/S2
65	COM11/S3
66	COM12/S4
67	COM13/S5
68	COM14/S6
69	COM15/S7
70	COM16/S8
71	S9
72	S10
73	S11
74	S12
75	S13
76	S14
77	S15
78	S16
79	S17
80	S18

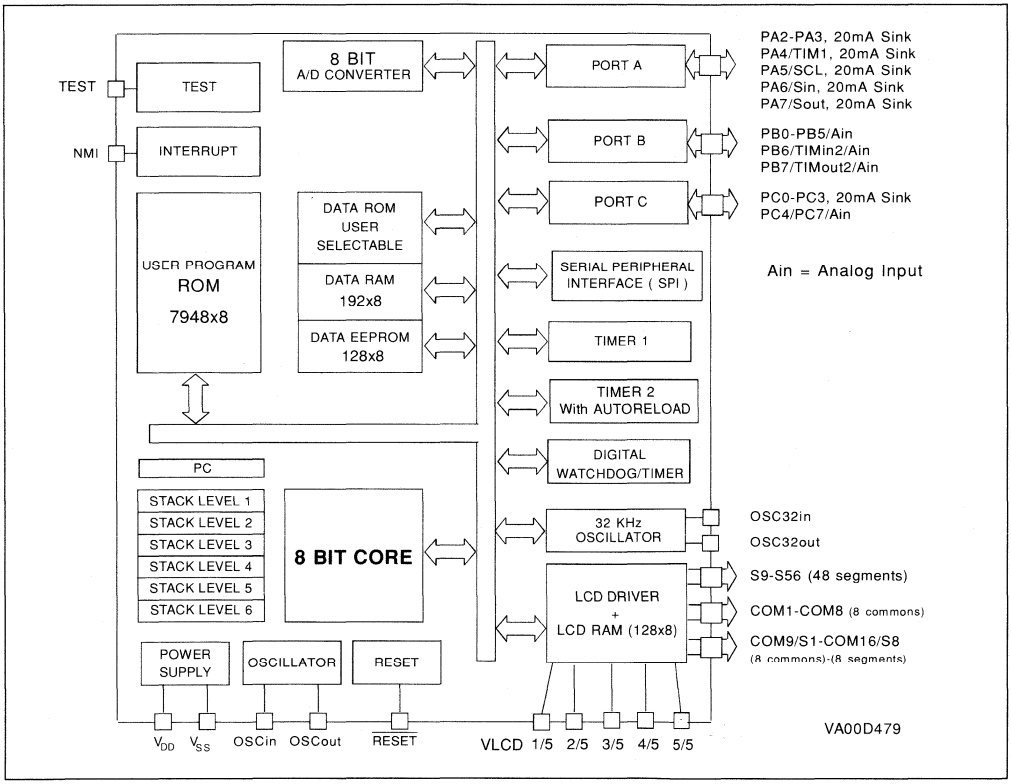
Pin number	Pin name
81	S19
82	S20
83	S21
84	S22
85	S23
86	S24
87	S25
88	S26
89	S27
90	S28
91	S29
92	S30
93	S31
94	S32
95	S33
96	S34
97	S35
98	S36
99	S37
100	S38

Note 1: 20mA SINK

GENERAL DESCRIPTION

The ST6280 microcontrollers is a member of the 8-bit HCMOS ST62xx family, a series of devices oriented to low-medium complexity applications. All ST62xx members are based on a building block approach: a common core is associated with a combination of on-chip peripherals (macrocells). The macrocells of the ST6280 are an advanced LCD driver/controller with 48 segments, 8 backplanes and 8 software selectable segment/backplane outputs able to drive up to 48×16 (768) or 56×8 (448) segments, one 8 bit reload timer with 7 bit programmable prescaler (Timer 2), one 8 bit standard timer/counter with a 7-bit software programmable prescaler (Timer 1), the digital watchdog timer (DWD), an 8-bit A/D Converter with up to 12 analog inputs, a 32kHz Oscillator, and an 8-bit synchronous serial peripheral interface (SPI). In addition this device offers 128 bytes of EEPROM for storage of non-volatile data. Thanks to these peripherals the ST6280 is well suited for general purpose, automotive, security, appliance and industrial applications. The ST62E80 EPROM version is available for prototyping and low-volume production, an OTP version is also available

Figure 2. ST6280 Block Diagram



PIN DESCRIPTION

V_{DD} and V_{SS}. Power is supplied to the MCU using these two pins. V_{DD} is power and V_{SS} is the ground connection.

OSCin and OSCout. These pins are internally connected with the on-chip oscillator circuit. A quartz crystal or a ceramic resonator can be connected between these two pins in order to allow the correct operation of the MCU with various stability/cost trade-offs. OSCin is the input pin, OSCout is the output pin. An external clock signal can be applied to OSCin.

$\overline{\text{RESET}}$. The active low $\overline{\text{RESET}}$ pin is used to restart the microcontroller at the beginning of its program. The $\overline{\text{RESET}}$ pin is provided with an on-chip pull-up resistor and schmitt trigger input characteristics.

TEST. The TEST pin is used to place the MCU into special operating mode. TEST must be held at V_{SS} for normal operation (an internal pull-down resistor selects normal operating mode if TEST pin is not connected).

NMI. The NMI pin provides the capability for asynchronous applying an external top priority interrupt to the MCU. This pin is falling edge sensitive. The NMI pin is provided with an on-chip pull-up resistor and schmitt trigger input characteristics.

PA4/TIM1. This pin can be used as Timer 1 I/O pin. In input mode it is connected to the timer prescaler input and acts as the external timer clock or as the control gate for the internal timer clock. In the output mode the timer pin outputs the timer data bit when a time out occurs.
To use this pin as Timer output the I/O pin has to be programmed as open-drain output. To use this pin as Timer input the I/O pin has to be programmed as input.

PB6/TIMIN2, PB7/TIMOUT2. These pins are the Input and Output pins of Autoreload Timer 2. The timer input pin TIMIN2 is connected to port line PB6. To use the line as timer input function, PB6 has to be programmed as input with or without pull-up. The timer output pin is connected to the port line PB7. A dedicated bit in the TIMER 2 mode control register sets the line as timer output function .

PA2-PA7. These 6 lines are organized as one I/O port (A). Each line may be configured under software control as an input with or without internal pull-up resistor, an interrupt generating input with pull-up resistor, or an open-drain or push-pull output. In output mode these lines can also sink 20mA for direct LED or triac driving. PA5-PA7 can also be used as respectively Clock, Data in and Data out pins for the on-chip SPI to carry the synchronous serial I/O signals. PA4 can also be used as the TIMER 1 I/O pin.

PB0-PB7. These 8 lines are organized as one I/O port (B). Each line may be configured under software control as an input with or without internal pull-up resistor, an interrupt generating input with pull-up resistor, open-drain or push-pull output or as an analog input for the A/D converter. PB6 is also connected to the TIMER 2 input function while PB7 can act as the TIMER 2 output. Port B has schmitt trigger inputs and a 5mA drive capability in output mode.

PC0-PC3, PC4-PC7. These 8 lines are organized as one I/O port (C). Each line may be configured under software control as an input with or without internal pull-up resistor, an interrupt generating input with pull-up resistor, or an open-drain or push-pull output. PC0-PC3 can also sink 20mA for direct LED or triac driving while PC4-PC7 can be programmed as analog inputs for the A/D converter. Port C has schmitt trigger inputs and a 5mA drive capability in output mode.

COM1-COM8. These eight pins are the LCD peripheral common outputs. They are the outputs of the on-chip backplane voltage generator which is used for multiplexing the LCD segment lines.

S9-S56. These pins are the 48 LCD peripheral driver outputs of the ST6280. Segments S1-S8 are multiplexed with COM9-COM16 and their function is software selectable.

COM9/S1-COM16-S8. These pins are the 8 multiplexed common/segment lines. Under software selected control, they can act as LCD common outputs allowing a 48×16 dot matrix operation, or they can act as segment outputs allowing 56×8 dot matrix operation.

VLCD1/5-VLCD4/5. Resistor network nodes for determining the intermediate display voltage levels on COM1-COM8/COM16 and S1/S8-S56 pins during multiplex operation.

OSC32in and OSC32out. These pins are internally connected with the on-chip 32kHz oscillator circuit. A 32.768kHz quartz crystal can be connected between these two pins if it is necessary to provide the LCD stand-by clock and real time interrupt. OSC32in is the input pin, OSC32out is the output pin.

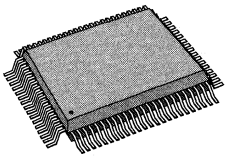
SGS-THOMSON MICROELECTRONICS

ST62E80
ST62T80

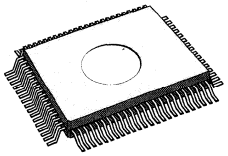
8-BIT EPROM HCMOS MCU WITH DOT MATRIX LCD DRIVER EEPROM AND A/D CONVERTER

PRODUCT OVERVIEW

- 4.5 to 5.5V supply operating range
- 8.4MHz Maximum Clock Frequency
- -40 to +85°C Operating Temperture Range
- Run, Wait & Stop Modes
- 5 different interrupt vectors
- Look-up table capability in EPROM
  User EPROM: 8192 bytes
  Data RAM: 192 bytes
  LCD RAM: 128 bytes
  EEPROM: 128 bytes
- PQFP100 and CQFP100-W packages
- 12 fully software programmable I/O as:
  - Input with/without pull-up resistor
  - Input with interrupt generation
  - Open-Drain or Push-pull outputs
  - Analog Inputs
- 10 I/O lines can sink up to 20mA for direct LED or TRIAC driving and have SPI alternate functions
- One 8-bit counter with 7-bit programmable prescalers (Timer 1)
- One 8-bit auto-reload timer with 7-bit programmable prescaler (Timer 2)
- Software activated digital watchdog
- 8-bit A/D converter with up to 12 analog inputs
- 8-bit synchronous serial peripheral interface (SPI)
- LCD driver with 48 segment outputs, 8 backplane outputs and 8 software selectable segment/backplane outputs able to drive up to 48x16 (768) or 56x8 (448) LCD segments.
- 32kHz oscillator for stand-by LCD operation
- One external not maskable interrupt
- 9 powerful addressing modes
- The accumulator, the X, Y, V & W registers, the port and peripherals data & control registers are addressed in the data space as RAM locations.
- The ST62E80 is the EPROM version, ST62T80 is the OTP version, fully compatible with ST6280 ROM version.



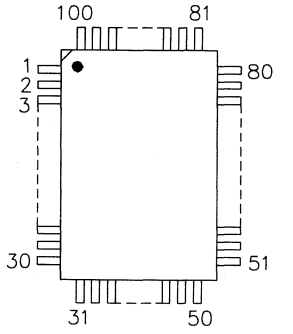
PQFP100



CQFP100-W

(Ordering Information at the end of the datasheet)

Figure 3. 100 Pin Quad Flat Pack (QFP) Package Pinout



ST62E80/T80 Pin Description

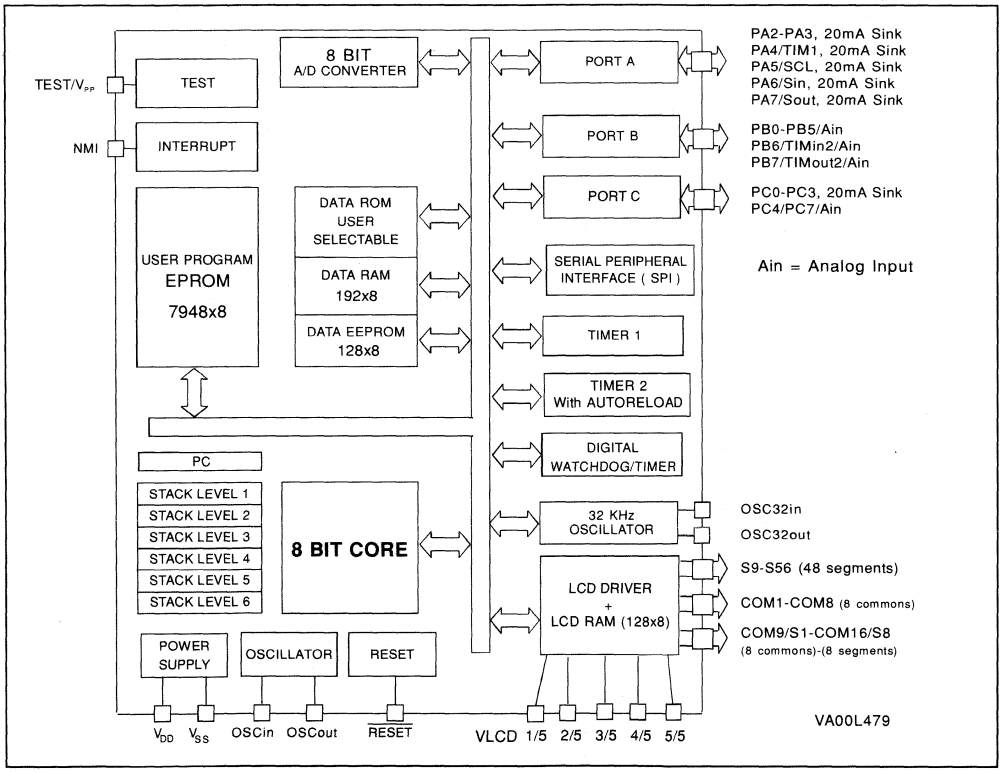
Pin number	Pin name	Pin number	Pin name	Pin number	Pin name	Pin number	Pin name
1	S39	31	PC7/Ain	80	S18	100	S38
2	S40	32	PC6/Ain	79	S17	99	S37
3	S41	33	PC5/Ain	78	S16	98	S36
4	S42	34	PC4/Ain	77	S15	97	S35
5	S43	35	PC3 (1)	76	S14	96	S34
6	S44	36	PC2 (1)	75	S13	95	S33
7	S45	37	PC1 (1)	74	S12	94	S32
8	S46	38	PC0 (1)	73	S11	93	S31
9	S47	39	NMI	72	S10	92	S30
10	S48	40	V_{DD}	71	S9	91	S29
11	S49	41	V_{SS}	70	COM16/S8	90	S28
12	S50	42	VLCD	69	COM15/S7	89	S27
13	S51	43	VLCD4/5	68	COM14/S6	88	S26
14	S52	44	VLCD3/5	67	COM13/S5	87	S25
15	S53	45	VLCD2/5	66	COM12/S4	86	S24
16	S54	46	VLCD1/5	65	COM11/S3	85	S23
17	S55	47	PA7/Sout (1)	64	COM10/S2	84	S22
18	S56	48	PA6/Sin (1)	63	COM9/S1	83	S21
19	PB7/TIMout2 /Ain	49	PA5/SCL (1)	62	COM8	82	S20
		50	PA4/TIM1 (1)	61	COM7	81	S19
20	PB6/TIMin2 /Ain			60	COM6		
				59	COM5		
21	PB5/Ain			58	COM4		
22	PB4/Ain			57	COM3		
23	PB3/Ain			56	COM2		
24	PB2/Ain			55	COM1		
25	PB1/Ain			54	OSC32in		
26	PB0/Ain			53	OSC32out		
27	TEST/V_{PP}			52	PA2 (1)		
28	OSCout			51	PA3 (1)		
29	OSCin						
30	RESET						

Note 1: 20mA SINK

GENERAL DESCRIPTION

The ST62E80, ST62T80 microcontrollers are members of the 8-bit HCMOS ST62xx family, a series of devices oriented to low-medium complexity applications. They are the EPROM/OTP versions of the ST6280 ROM device and are suitable for prototyping and low-volume production. All ST62xx members are based on a building block approach: a common core is associated with a combination of on-chip peripherals (macrocells). The macrocells of the ST6280 family are: an advanced LCD driver/controller with 48 segments, 8 backplanes and 8 software selectable segment/backplane outputs able to drive up to 48×16 (768) or 56×8 (448) segments, one 8 bit reload timer with 7 bit programmable prescaler (Timer 2), one 8 bit standard timer/counter with a 7-bit software programmable prescaler (Timer 1), the digital watchdog timer (DWD), an 8-bit A/D Converter with up to 12 analog inputs, a 32kHz Oscillator, and an 8-bit synchronous serial peripheral interface (SPI). In addition this device offers 128 bytes of EEPROM for storage of non-volatile data. Thanks to these peripherals the ST6280 family is well suited for general purpose, automotive, security, appliance and industrial applications.

Figure 2. ST62E80/T80 Block Diagram



PIN DESCRIPTION

V_{DD} and V_{SS}. Power is supplied to the MCU using these two pins. V_{DD} is power and V_{SS} is the ground connection.

OSCin and OSCout. These pins are internally connected with the on-chip oscillator circuit. A quartz crystal or a ceramic resonator can be connected between these two pins in order to allow the correct operation of the MCU with various stability/cost trade-offs. OSCIN is the input pin, OSCOUT is the output pin. An external clock signal can be applied to OSCIN.

RESET. The active low $\overline{\text{RESET}}$ pin is used to restart the microcontroller at the beginning of its program. The $\overline{\text{RESET}}$ pin is provided with an on-chip pull-up resistor and schmitt trigger input characteristics.

TEST/Vpp The TEST pin is used to place the MCU into special operating mode. TEST must be held at V_{SS} for normal operation (an internal pull-down resistor selects normal operating mode if TEST pin is not connected).

NMI. The NMI pin provides the capability for asynchronous applying an external top priority interrupt to the MCU. This pin is falling edge sensitive. The NMI pin is provided with an on-chip pull-up resistor and schmitt trigger input characteristics.

PA4/TIM1. This pin can be used as Timer 1 I/O pin. In input mode it is connected to the timer prescaler input and acts as the external timer clock or as the control gate for the internal timer clock. In the output mode the timer pin outputs the timer data bit when a time out occurs.
To use this pin as Timer output the I/O pin has to be programmed as open-drain output. To use this pin as Timer input the I/O pin has to be programmed as input.

PB6/TIMIN2, PB7/TIMOUT2. These pins are the Input and Output pins of Autoreload Timer 2. The timer input pin TIMIN2 is connected to port line PB6. To use the line as timer input function, PB6 has to be programmed as input with or without pull-up. The timer output pin is connected to the port line PB7. A dedicated bit in the TIMER 2 mode control register sets the line as timer output function .

PA2-PA7. These 6 lines are organized as one I/O port (A). Each line may be configured under software control as an input with or without internal pull-up resistor, an interrupt generating input with pull-up resistor, or an open-drain or push-pull output. In output mode these lines can also sink 20mA for direct LED or triac driving. PA5-PA7 can also be used as respectively Clock, Data in and Data out pins for the on-chip SPI to carry the synchronous serial I/O signals. PA4 can also be used as the TIMER 1 I/O pin.

PB0-PB7. These 8 lines are organized as one I/O port (B). Each line may be configured under software control as an input with or without internal pull-up resistor, an interrupt generating input with pull-up resistor, open-drain or push-pull output or as an analog input for the A/D converter. PB6 is also connected to the TIMER 2 input function while PB7 can act as the TIMER 2 output. Port B has schmitt trigger inputs and a 5mA drive capability in output mode.

PC0-PC3, PC4-PC7. These 8 lines are organized as one I/O port (C). Each line may be configured under software control as an input with or without internal pull-up resistor, an interrupt generating input with pull-up resistor, or an open-drain or push-pull output. PC0-PC3 can also sink 20mA for direct LED or triac driving while PC4-PC7 can be programmed as analog inputs for the A/D converter. Port C has schmitt trigger inputs and a 5mA drive capability in output mode.

COM1-COM8. These eight pins are the LCD peripheral common outputs. They are the outputs of the on-chip backplane voltage generator which is used for multiplexing the LCD segment lines.

S9-S56. These pins are the 48 LCD peripheral driver outputs of the ST62E80, ST62T80. Segments S1-S8 are multiplexed with COM9-COM16 and their function is software selectable.

COM9/S1-COM16/S8. These pins are the 8 multiplexed common/segment lines. Under software selected control, they can act as LCD common outputs allowing a 48×16 dot matrix operation, or they can act as segment outputs allowing 56×8 dot matrix operation.

VLCD1/5-VLCD4/5. Resistor network nodes for determining the intermediate display voltage levels on COM1-COM8/COM16 and S1/S8-S56 pins during multiplex operation.

OSC32in and OSC32out. These pins are internally connected with the on-chip 32kHz oscillator circuit. A 32.768kHz quartz crystal can be connected between these two pins if it is necessary to provide the LCD stand-by clock and real time interrupt. OSC32in is the input pin, OSC32out is the output pin.

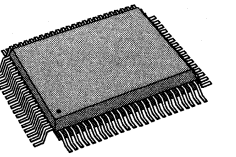
SGS-THOMSON MICROELECTRONICS

ST6285

8-BIT HCMOS MCU WITH DOT MATRIX LCD DRIVER AND A/D CONVERTER

PRODUCT OVERVIEW

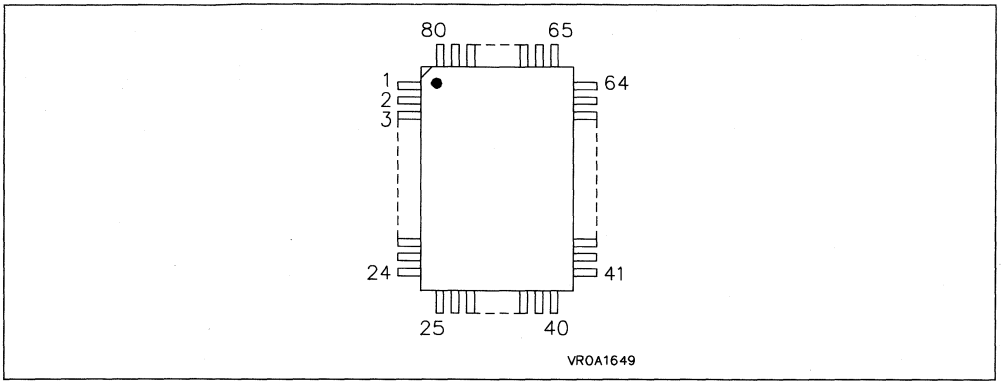
- 4.5 to 5.5V supply operating range
- 8.4 MHz Maximum Clock Frequency
- -40 to +85°C Operating Temperature Range
- Run, Wait & Stop Modes
- 5 different interrupt vectors
- Look-up table capability in ROM
- User ROM: 7948 bytes
  Reserved ROM: 244 bytes
  Data RAM: 192 bytes
  LCD RAM: 96 bytes
- PQFP80 package
- 8 fully software programmable I/O as:
  - Input with/without pull-up resistor
  - Input with interrupt generation
  - Open-Drain or Push-pull outputs
  - Analog Inputs
- 4 I/O lines can sink up to 20mA for direct LED or TRIAC driving and have SPI alternate functions
- Software activated digital watchdog
- 8-bit A/D converter with up to 8 analog inputs
- 8-bit synchronous serial peripheral interface (SPI)
- LCD driver controller with 40 segments outputs, 8 backplane and 8 software selectable segment/backplane outputs able to drive up to 40x16 (640) or 48x8 (384) segments.
- One external not maskable interrupt
- 9 powerful addressing modes
- The accumulator, the X, Y, V & W registers, the port and peripherals data & control registers are addressed in the data space as RAM locations.
- The ST62E85 is the EPROM version, ST62T85 is the OTP version



PQFP80

(Ordering information at the end of the datasheet)

Figure 4. 80 Pin Quad Flat Pack (QFP) Package Pinout



ST6285 Pin Description

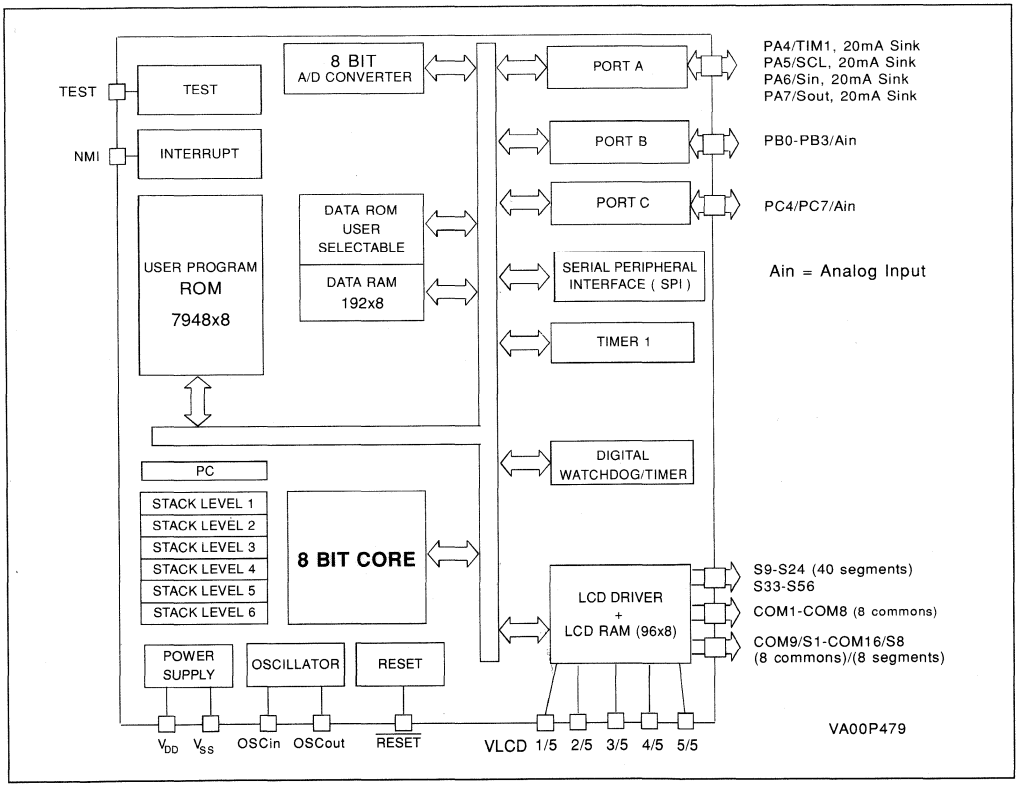
Pin number	Pin name	Pin number	Pin name	Pin number	Pin name	Pin number	Pin name
1	S41	25	PC7	64	S16	80	S40
2	S42	26	PC6	63	S15	79	S39
3	S43	27	PC5	62	S14	78	S38
4	S44	28	PC4	61	S13	77	S37
5	S45	29	NMI	60	S12	76	S36
6	S46	30	V_{DD}	59	S11	75	S35
7	S47	31	V_{SS}	58	S10	74	S34
8	S48	32	VLCD	57	S9	73	S33
9	S49	33	VLCD4/5	56	COM16/S8	72	S24
10	S50	34	VLCD3/5	55	COM15/S7	71	S23
11	S51	35	VLCD2/5	54	COM14/S6	70	S22
12	S52	36	VLCD1/5	53	COM13/S5	69	S21
13	S53	37	PA7/Sout (1)	52	COM12/S4	68	S20
14	S54	38	PA6/Sin (1)	51	COM11/S3	67	S19
15	S55	39	PA5/SCL (1)	50	COM10/S2	66	S18
16	S56	40	PA4/TIM1 (1)	49	COM9/S1	65	S17
17	PB3			48	COM8		
18	PB2			47	COM7		
19	PB1			46	COM6		
20	PB0			45	COM5		
21	TEST/V_{PP}			44	COM4		
22	OSCout			43	COM3		
23	$\overline{\text{OSCin}}$			42	COM2		
24	$\overline{\text{RESET}}$			41	COM1		

Note 1: 20mA SINK

GENERAL DESCRIPTION

The ST6285 microcontrollers is a member of the 8-bit HCMOS ST62xx family, a series of devices oriented to low-medium complexity applications. All ST62xx members are based on a building block approach: a common core is associated with a combination of on-chip peripherals (macrocells). The macrocells of the ST6285 are an advanced LCD driver/controller with 40 segments, 8 backplanes and 8 software selectable segment/backplane outputs able to drive up to 40×16 (640) or 48×8 (384) segments, one 8 bit standard timer/counter with a 7-bit software programmable prescaler (Timer 1), the digital watchdog timer (DWD), an 8-bit A/D Converter with up to 12 analog inputs and an 8-bit synchronous serial peripheral interface (SPI). Thanks to these peripherals the ST6285 is well suited for general purpose, automotive, security, appliance and industrial applications. The ST62E85 EPROM version is available for prototyping and low-volume production, an OTP version is also available.

Figure 5. ST6285 Block Diagram



PIN DESCRIPTION

V_{DD} and V_{SS}. Power is supplied to the MCU using these two pins. V_{DD} is power and V_{SS} is the ground connection.

OSCin and OSCout. These pins are internally connected with the on-chip oscillator circuit. A quartz crystal or a ceramic resonator can be connected between these two pins in order to allow the correct operation of the MCU with various stability/cost trade-offs. OSCin is the input pin, OSCout is the output pin. An external clock signal can be applied to OSCin.

$\overline{\text{RESET}}$. The active low $\overline{\text{RESET}}$ pin is used to restart the microcontroller at the beginning of its program. The $\overline{\text{RESET}}$ pin is provided with an on-chip pull-up resistor and schmitt trigger input characteristics.

TEST/Vpp The TEST pin is used to place the MCU into special operating mode. TEST must be held at V_{SS} for normal operation (an internal pull-down resistor selects normal operating mode if TEST pin is not connected).

NMI. The NMI pin provides the capability for asynchronous applying an external top priority interrupt to the MCU. This pin is falling edge sensitive. The NMI pin is provided with an on-chip pull-up resistor and schmitt trigger input characteristics.

PA4/TIM1. This pin can be used as Timer 1 I/O pin. In input mode it is connected to the timer prescaler input and acts as the external timer clock or as the control gate for the internal timer clock. In the output mode the timer pin outputs the timer data bit when a time out occurs.
To use this pin as Timer output the I/O pin has to be programmed as open-drain output. To use this pin as Timer input the I/O pin has to be programmed as input.

PA4-PA7. These 4 lines are organized as one I/O port (A). Each line may be configured under software control as an input with or without internal pull-up resistor, an interrupt generating input with pull-up resistor, or an open-drain or push-pull output. In output mode these lines can also sink 20mA for direct LED or triac driving. PA5-PA7 can also be used as respectively Clock, Data in and Data out pins for the on-chip SPI to carry the synchronous serial I/O signals. PA4 can also be used as the TIMER 1 I/O pin.

PB0-PB3. These 4 lines are organized as one I/O port (B). Each line may be configured under software control as an input with or without internal pull-up resistor, an interrupt generating input with pull-up resistor, open-drain or push-pull output or as an analog input for the A/D converter.

PC4-PC7. These 4 lines are organized as one I/O port (C). Each line may be configured under software control as an input with or without internal pull-up resistor, an interrupt generating input with pull-up resistor, or an open-drain or push-pull output. PC4-PC7 can be programmed as analog inputs for the A/D converter. Port C has schmitt trigger inputs and a 5mA drive capability in output mode.

COM1-COM8. These eight pins are the LCD peripheral common outputs. They are the outputs of the on-chip backplane voltage generator which is used for multiplexing the LCD segment lines.

S9/S24-S33/S56. These pins are the 40 LCD peripheral driver outputs of the ST62E85, ST62T85. Segments S1-S8 are multiplexed with COM9-COM16 and their function is software selectable.

COM9/S1-COM16/S8. These pins are the 8 multiplexed common/segment lines. Under software selected control, they can act as LCD common outputs allowing a 40×16 dot matrix operation, or they can act as segment outputs allowing 48×8 dot matrix operation.

VLCD1/5-VLCD4/5. Resistor network nodes for determining the intermediate display voltage levels on COM1-COM8/COM16 and S1/S8-S56 pins during multiplex operation.

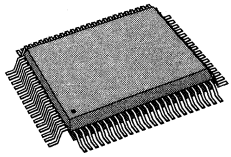
SGS-THOMSON MICROELECTRONICS

ST62E85
ST62T85

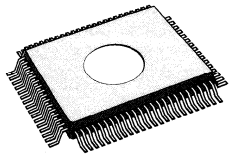
8-BIT EPROM HCMOS MCU WITH DOT MATRIX LCD DRIVER AND A/D CONVERTER

PRODUCT OVERVIEW

- 4.5 to 5.5V supply operating range
- 8.4MHz Maximum Clock Frequency
- -40 to +85°C Operating Temperture Range
- Run, Wait & Stop Modes
- 5 different interrupt vectors
- Look-up table capability in EPROM
  User EPROM: 8192 bytes
  Data RAM: 192 bytes
  LCD RAM: 96 bytes
- PQFP80 and CQFP80-W packages
- 8 fully software programmable I/O as:
  - Input with/without pull-up resistor
  - Input with interrupt generation
  - Open-Drain or Push-pull outputs
  - Analog Inputs
- 4 I/O lines can sink up to 20mA for direct LED or TRIAC driving and have SPI alternate functions
- One 8-bit counter with 7-bit programmable prescalers (Timer 1)
- Software activated digital watchdog
- 8-bit A/D converter with up to 8 analog inputs
- 8-bit synchronous serial peripheral interface (SPI)
- LCD driver with 40 segment outputs, 8 backplane outputs and 8 software selectable segment/backplane outputs able to drive up to 40x16 (640) or 48x8 (384) LCD segments.
- One external not maskable interrupt
- 9 powerful addressing modes
- The accumulator, the X, Y, V & W registers, the port and peripherals data & control registers are addressed in the data space as RAM locations.
- The ST62E85 is the EPROM version, ST62T85 is the OTP version, fully compatible with ST6285 ROM version.



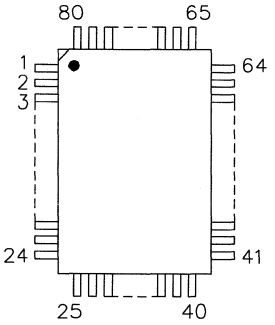
PQFP80



CQFP80-W

(Ordering Information at the end of the datasheet)

Figure 6. 80 Pin Quad Flat Pack (QFP) Package Pinout



ST62E85 Pin Description

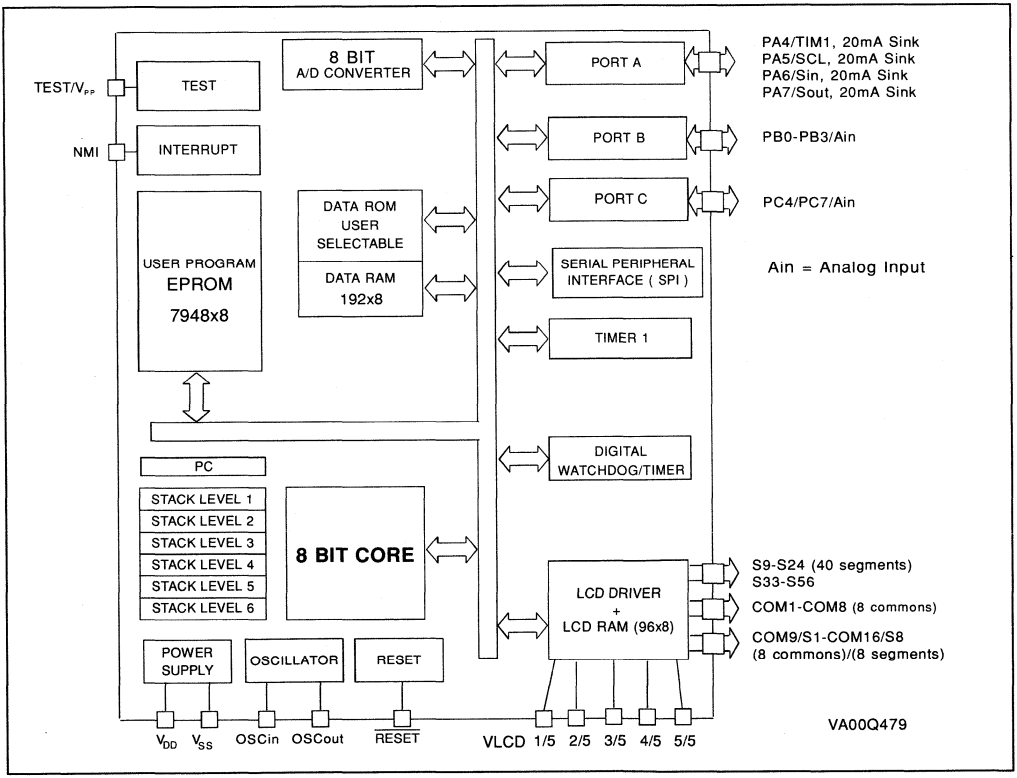
Pin number	Pin name	Pin number	Pin name	Pin number	Pin name	Pin number	Pin name
1	S41	25	PC7	64	S16	80	S40
2	S42	26	PC6	63	S15	79	S39
3	S43	27	PC5	62	S14	78	S38
4	S44	28	PC4	61	S13	77	S37
5	S45	29	NMI	60	S12	76	S36
6	S46	30	V_{DD}	59	S11	75	S35
7	S47	31	V_{SS}	58	S10	74	S34
8	S48	32	VLCD	57	S9	73	S33
9	S49	33	VLCD4/5	56	COM16/S8	72	S24
10	S50	34	VLCD3/5	55	COM15/S7	71	S23
11	S51	35	VLCD2/5	54	COM14/S6	70	S22
12	S52	36	VLCD1/5	53	COM13/S5	69	S21
13	S53	37	PA7/Sout (1)	52	COM12/S4	68	S20
14	S54	38	PA6/Sin (1)	51	COM11/S3	67	S19
15	S55	39	PA5/SCL (1)	50	COM10/S2	66	S18
16	S56	40	PA4/TIM1 (1)	49	COM9/S1	65	S17
17	PB3			48	COM8		
18	PB2			47	COM7		
19	PB1			46	COM6		
20	PB0			45	COM5		
21	TEST/V_{PP}			44	COM4		
22	OSCout			43	COM3		
23	$\overline{\text{OSCin}}$			42	COM2		
24	$\overline{\text{RESET}}$			41	COM1		

Note 1: 20mA SINK

GENERAL DESCRIPTION

The ST62E85, ST62T85 microcontrollers are members of the 8-bit HCMOS ST62xx family, a series of devices oriented to low-medium complexity applications. They are the EPROM/OTP versions of the ST6285 ROM device and are suitable for prototyping and low-volume production. All ST62xx members are based on a building block approach: a common core is associated with a combination of on-chip peripherals (macrocells). The macrocells of the ST6285 family are: an advanced LCD driver/controller with 40 segments, 8 backplanes and 8 software selectable segment/backplane outputs able to drive up to 40×16 (640) or 48×8 (384) segments, one 8 bit standard timer/counter with a 7-bit software programmable prescaler (Timer 1), the digital watchdog timer (DWD), an 8-bit A/D Converter with up to 12 analog inputs and an 8-bit synchronous serial peripheral interface (SPI). Thanks to these peripherals the ST6285 family is well suited for general purpose, automotive, security, appliance and industrial applications.

Figure 2. ST62E85/T85 Block Diagram



PIN DESCRIPTION

V_{DD} and V_{SS}. Power is supplied to the MCU using these two pins. V_{DD} is power and V_{SS} is the ground connection.

OSCin and OSCout. These pins are internally connected with the on-chip oscillator circuit. A quartz crystal or a ceramic resonator can be connected between these two pins in order to allow the correct operation of the MCU with various stability/cost trade-offs. OSCin is the input pin, OSCout is the output pin. An external clock signal can be applied to OSCin.

$\overline{\text{RESET}}$. The active low $\overline{\text{RESET}}$ pin is used to restart the microcontroller at the beginning of its program. The $\overline{\text{RESET}}$ pin is provided with an on-chip pull-up resistor and schmitt trigger input characteristics.

TEST/Vpp The TEST pin is used to place the MCU into special operating mode. TEST must be held at V_{SS} for normal operation (an internal pull-down resistor selects normal operating mode if TEST pin is not connected).

NMI. The NMI pin provides the capability for asynchronous applying an external top priority interrupt to the MCU. This pin is falling edge sensitive. The NMI pin is provided with an on-chip pull-up resistor and schmitt trigger input characteristics.

PA4/TIM1. This pin can be used as Timer 1 I/O pin. In input mode it is connected to the timer prescaler input and acts as the external timer clock or as the control gate for the internal timer clock. In the output mode the timer pin outputs the timer data bit when a time out occurs.
To use this pin as Timer output the I/O pin has to be programmed as open-drain output. To use this pin as Timer input the I/O pin has to be programmed as input.

PA4-PA7. These 4 lines are organized as one I/O port (A). Each line may be configured under software control as an input with or without internal pull-up resistor, an interrupt generating input with pull-up resistor, or an open-drain or push-pull output. In output mode these lines can also sink 20mA for direct LED or triac driving. PA5-PA7 can also be used as respectively Clock, Data in and Data out pins for the on-chip SPI to carry the synchronous serial I/O signals. PA4 can also be used as the TIMER 1 I/O pin.

PB0-PB3. These 4 lines are organized as one I/O port (B). Each line may be configured under software control as an input with or without internal pull-up resistor, an interrupt generating input with pull-up resistor, open-drain or push-pull output or as an analog input for the A/D converter.

PC4-PC7. These 4 lines are organized as one I/O port (C). Each line may be configured under software control as an input with or without internal pull-up resistor, an interrupt generating input with pull-up resistor, or an open-drain or push-pull output. PC4-PC7 can be programmed as analog inputs for the A/D converter. Port C has schmitt trigger inputs and a 5mA drive capability in output mode.

COM1-COM8. These eight pins are the LCD peripheral common outputs. They are the outputs of the on-chip backplane voltage generator which is used for multiplexing the LCD segment lines.

S9/S24-S33/S56. These pins are the 40 LCD peripheral driver outputs of the ST62E85, ST62T85. Segments S1-S8 are multiplexed with COM9-COM16 and their function is software selectable.

COM9/S1-COM16/S8. These pins are the 8 multiplexed common/segment lines. Under software selected control, they can act as LCD common outputs allowing a 40×16 dot matrix operation, or they can act as segment outputs allowing 48×8 dot matrix operation.

VLCD1/5-VLCD4/5. Resistor network nodes for determining the intermediate display voltage levels on COM1-COM8/COM16 and S1/S8-S56 pins during multiplex operation.

SGS-THOMSON MICROELECTRONICS

ST6240-KIT

STARTER KIT FOR ST624x MCU FAMILY

Preliminary Data

HARDWARE FEATURES

- Immediate evaluation of ST62E40 with a demonstration examples
- Program debugging by connection of an application environment to the board
- On board programming of ST62E40 and ST62T40
- In-circuit programming of ST62E4x and ST62T4x through the Starter Kit

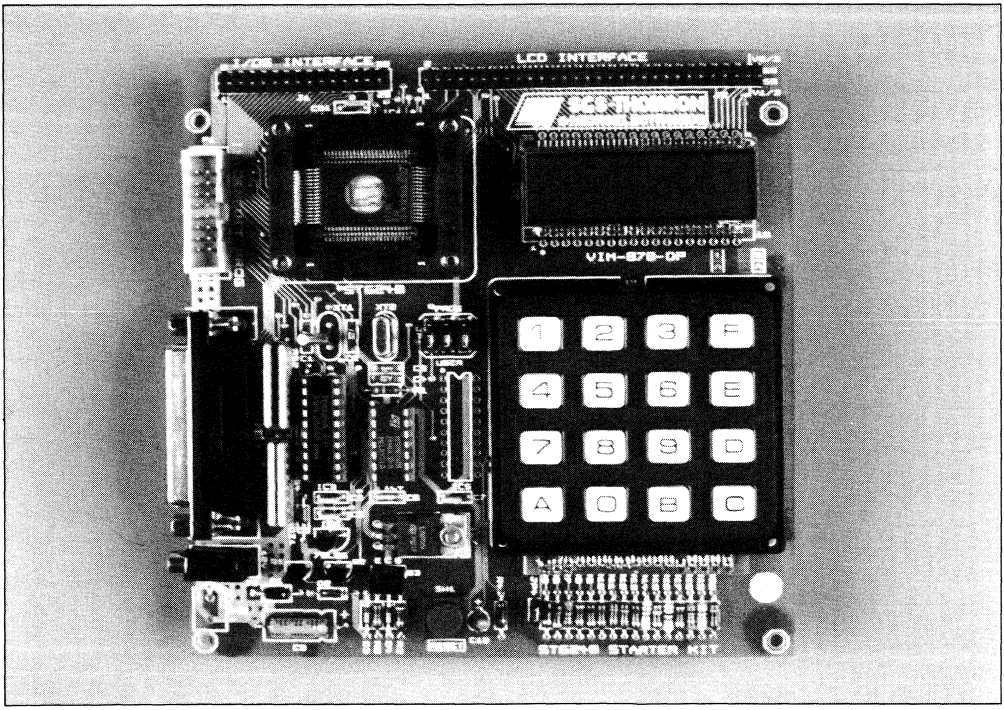
SOFTWARE FEATURES

- Software simulator including LCD display and I/O read/write
- Assembler, linker, debugger
- EPROM/OTP programming utilities
- Application examples

DESCRIPTION

The ST6240 Starter Kit can be used for evaluation, simulation and emulation purposes. First, it can be used to demonstrate the capabilites of the ST6240. It is only necessary to connect the supply to the board and to load the demonstration software provided with the Kit into the ST62E40 sample; LCD and keyboard interfacing can be immediately evaluated.

The same board can be used as a hardware interface to the software simulator when connected to the PC, allowing display values from the simulator to be displayed directly on the LCD. Analog or digital values from the ST624x I/O pins can also be loaded directly to the simulator.



DESCRIPTION (Continued)

Once the program is successfully simulated, it can be loaded in the ST62E40 sample with the on-board programmer.
The application environment can be connected to the Starter Kit via the I/O connector to perform a full evaluation of the user application.

In addition, since the LCD is connected to the PCB via a socket, it can easily be removed and replaced by a customized LCD.

QFP packages are difficult to handle manually, so an in-circuit programming facility is provided with the Kit to enable programming, via the Starter Kit board, of any ST62E4x (EPROM) or ST62T4x (OTP) already soldered in the user application board.

Hardware items

The Kit PCB includes a QFP80 socket, a 16 key keyboard, a 32 segment x 4 LCD, an ST62E40 and cables plus a power supply.

Pins are available for direct connection to an application.

The board is connected to the PC via the parallel port.

Software items

The diskette provided with this kit includes an enhanced simulator including control of the external LCD display and I/O read/write, assembler, linker, debugger, EPROM/OTP ST6 programming facilities and demonstration examples.

Documentation

A full set of documents is provided with the Kit including the ST62 LCD drive data book, a Kit guide and the ST62/63 Software Development Tools user manual.

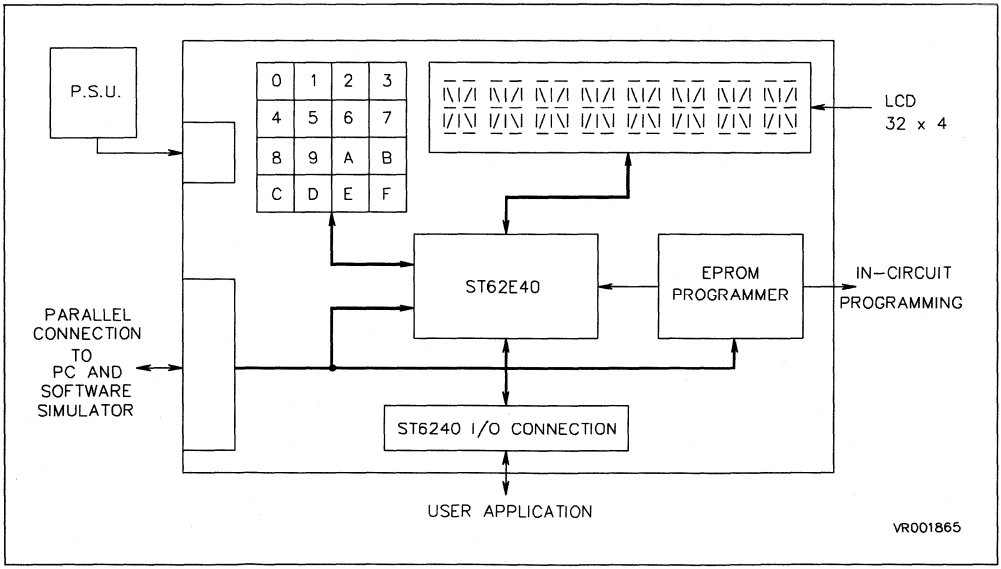
System requirement

The ST624x Starter Kit communicates with a PC-AT compatible Personal Computer equipped with a hard disk and a 5"1/4 diskette drive, 640k of conventional memory, one parallel Centronic port and MS-DOS version 3.10 or higher.

Ordering information

Sales type	Description
ST6240-KIT/220	Complete kit for operation from 220Vac mains
ST6240-KIT/110	Complete kit for operation from 110Vac mains
ST6240-KIT/UK	Complete kit for operation in UK

Block Diagram of ST6240 Starter Kit



SGS-THOMSON MICROELECTRONICS

*fuzzy*TECH™ ST6 Explorer Edition

FUZZY LOGIC COMPILER FOR ST6

GRAPHIC DESIGN EDITORS

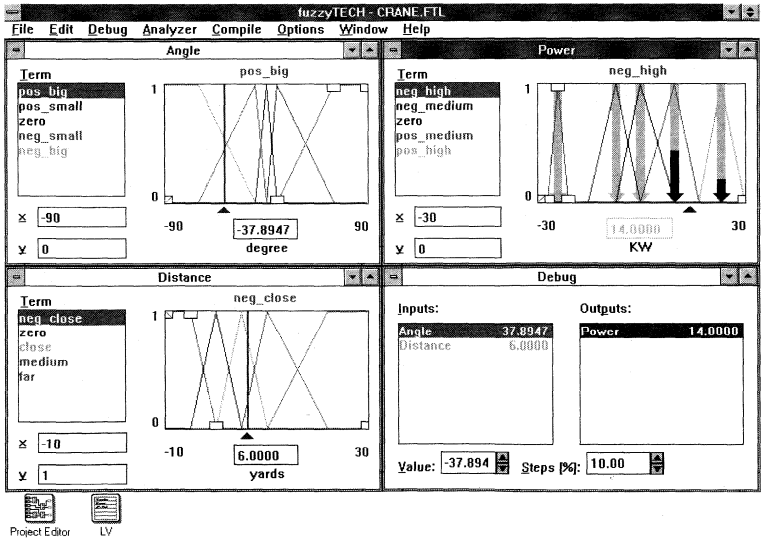
- Linguistic Variable Editor
  - Up to 7 labels per variable
  - Full 8-bit resolution
  - S.Z. Lambda and Pi-type membership functions
- Rules Editor
  - Full graphical input as matrices or spreadsheets
  - Supports standard Max-Min inference method
  - Allows up to 125 rules
- Structure Editor
  - Up to 4 Input variables per module
  - 1 output variables per module
  - Center-of-Maximum defuzzication method (includes Center-of-Area with singletons and Center-of-Area with overlap approximation) method

REAL-TIME CODE GENERATOR

- ST6 Code Generator
  - Emits the fuzzy functions as optimized assembly code
  - No license fee for runtime code

OFFLINE SIMULATOR

- Interactive Debugging
  - Full graphical testing of system performance
  - Visualization of entire inference flow
  - Interactive optimization of system parameters
- Real Data Simulation
  - Uses prerecorded example data for a graphic simulation
  - Timeplot features for real-time analysis
  - Generates input/output files for interfacing with other simulation systems
- Model Simulation
  - Connects to built-in simulation model
  - Any programming language which runs under MS-Windows can be used
  - to program the simulation model
  - Animation of the running controller
- Graphic Analyzer Tools
  - Control surface analysis
  - Rule tracing
  - Membership function tracing



DESCRIPTION

The *fuzzyTECH* ST6 Explorer Edition is a development software for fuzzy logic based systems on the entire ST6 microcontroller family.

Full Graphical Development

The *fuzzyTECH* ST6 Explorer Edition has graphical tools for all developments steps, such as design, optimization and verification. At the push of a button, the built-in code generator implements the developed system in ST6 assembly language code. Based on FTL, the hardware-independent fuzzy technologies language the designed system is compatible with all other *fuzzyTECH* Editions.

Get A Hands-on Experience With Fuzzy Technology

The *fuzzyTECH* ST6 Explorer Edition contains everything you need for a comprehensive working knowledge about designing fuzzy logic systems. Its easy-to- use, all-graphics editors and tools guide you step-by-step throught the development phases of fuzzy systems.

Experiment With The Prefabricated Graphic Simulation

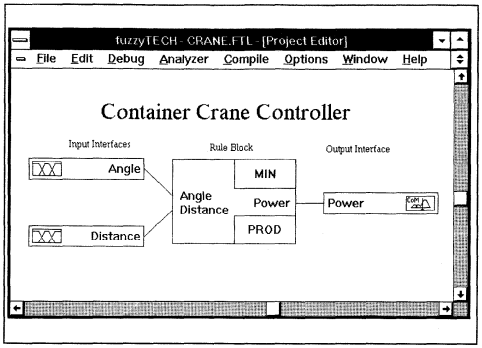
To get you started right away, the *fuzzyTECH* ST6 Explorer Edition comes with an animated simulation of a container crane controller. By experimenting with the fuzzy rules and system structure — and watching how your modifications affect the crane performance — you gain valuable insight into how fuzzy systems work.

Hardware/Software Requirements

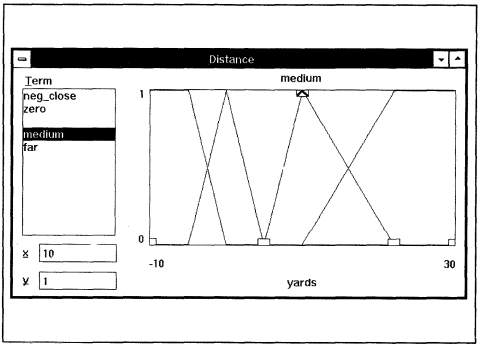
- A 80386 (higher) PC with at least 2MBytes memory
- MS-Windows 3.0 or higher and MS-DOS 3.3 or higher
- Hard disk with 5MB of free disk space and a 3.5" floppy
- VGA monitor supported by Windows

The generated ST6 assembly code runs on every member of the ST6 family. For the implementation, an ST6 assembler is required.

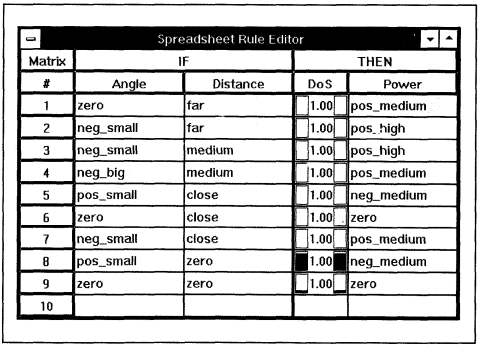
Example of System Stucture



Example of Linguistic Variables



Example of Rules Generation



Spreadsheet Rule Editor

Matrix	IF		THEN	
#	Angle	Distance	DoS	Power
1	zero	far	1.00	pos_medium
2	neg_small	far	1.00	pos_high
3	neg_small	medium	1.00	pos_high
4	neg_big	medium	1.00	pos_medium
5	pos_small	close	1.00	neg_medium
6	zero	close	1.00	zero
7	neg_small	close	1.00	pos_medium
8	pos_small	zero	1.00	neg_medium
9	zero	zero	1.00	zero
10				

DEVELOPMENT TOOLS

SOFTWARE DEVELOPMENT TOOLS FOR ST6 MCU FAMILY

- Includes:
  - Macro assembler
  - Linker
  - Software simulator
- Runs on MS-DOS systems
- Window based graphic interface
- Extensive symbol manipulation

GENERAL DESCRIPTION

Full software development tools is achieved using the ST6 Software Development Tools consisting of a powerful macro assembler, a linker and a software simulator.

The ST6 Macro assembler accepts a source file written in ST6 assembly language using any text editor package and transforms it in an ST6 executable file.

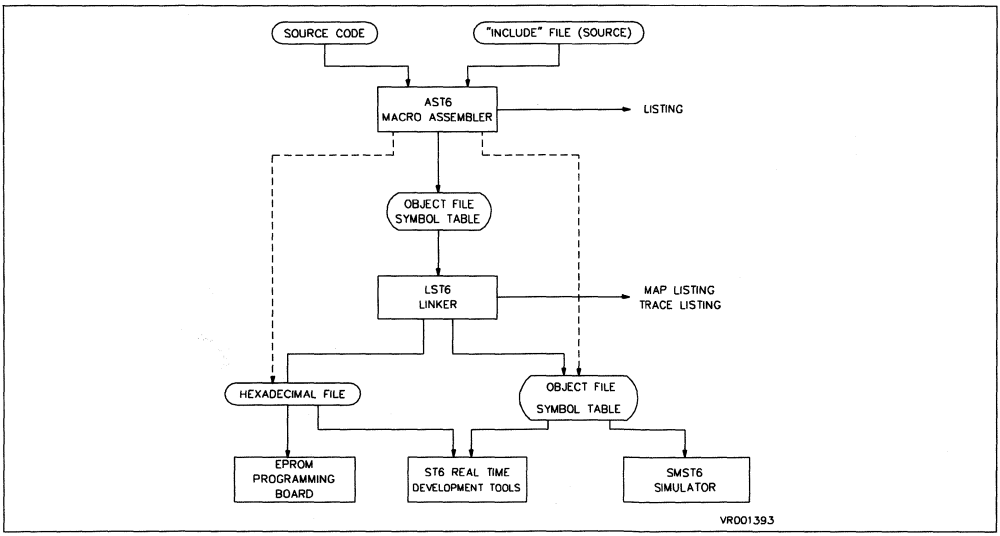
To enable good testability and fast debugging, many application software are made up of several modules, each of them performing an elementary task. Each module is assembled independently of the others, thus producing a number of object files. The ST6 Linker combines these object files into a single executable program. Both object format and hexadecimal format are produced. The hexadecimal file is used to program an EPROM while the object file is used to run the simulator or the debugger.

The ST6 Software Simulator allows the user to debug and execute any executable program written for any member of the ST62/ST63 family of microcontrollers without the aid of additional hardware.

Once debugged with the simulator, the program can be programmed into an EPROM device by using the hexadecimal file and the ST6 programming board. By plugging the EPROM device into the application hardware, simple applications can be debugged without the need of an emulator.

The ST6 Hardware Development Tools are required where high performance debugging is needed.

Figure 1. Development Flow Chart



ST6 ASSEMBLER

- Macro call and conditional assembly
- Extensive symbol manipulation
- Error diagnostics

General Description

The ST6 Macro assembler accepts a source file written in ST6 assembly language and transforms it into an executable file in relocatable object code format. When the whole program is in one file only, the assembler also generates an hexadecimal file (INTEL hex format) ready to be programmed into an EPROM device.

The assembler recognizes the use of section, symbols, macros and conditional assembly directives. In addition, the ST6 Assembler is able to produce detailed assembly listing and symbol cross reference file.

Figure 2. AST6 Directives

.ASCII	Stores in program space a string as a sequence of ASCII codes
.ASCIZ	Same as .ASCII followed by a null character
.BLOCK	Reserves a block of contiguous memory location
.BYTE	Stores successive bytes of data in program space
.DEF	Defines the characteristics of a data space location
.DISPLAY	Displays a string during assembly process
.DP_ON	Segments the data space
.EJECT	Starts a new listing page
.ELSE	Beginning of the alternative part in conditional assembly block
.END	End of source file
.ENDC	End of conditional assembly block
.ENDM	End of a macro definition
.EQU	Assigns the value of an expression to a label
.ERROR	User defined assembly error
.EXTERN	Defines a symbol as external
.IFC	Beginning of conditional assembly block
.INPUT	Includes an additional source file in the present one
.GLOBAL	Defines a symbol as global
.LABEL.W	Initializes Data ROM Window Register
.LABEL.D	Gains access to a label in a Data ROM Window
.LINESIZE	Set listing line length
.LIST	Enables the listing of specified fields of the source file
.MACRO	Beginning of a macro definition
.MEXIT	End of a macro expansion
.NOTRANSMIT	Inhibits symbol transmission to the linker
.ORG	Set current location counter
.PAGE_D	Specifies the page number in data space
.PL	Set listing page length
.PP_ON	Segments the program space in 2K pages
.ROMSIZE	Defines the available ROM size
.SECTION	Provides a logical partitioning of program space
.SET	Same as .EQU, but can be redefined in the source file
.TITLE	Assigns title to the document
.TRANSMIT	Transmits symbol definitions to the linker
.VERS	Defines the target ST6 device
.WARNING	User defined assembly warning
.WINDOW	Defines a continuous relocatable block of program code
.W_ON	Enables the use of the .WINDOW directive
.WORD	Stores successive words of data in program space

ST6 LINKER

- Links up to 32 modules
- Extensive symbol manipulation
- 33 sections (including interrupt vectors)
- Error diagnostics

The ST6 Linker is responsible for combining a number of object files into a single program, associating an absolute address to each section of code, and resolving any external references.

The ST6 Linker produces an hexadecimal file in INTEL format to be down loaded into an EPROM device and an object code file to be used with the simulator. The linker also produces a map file which gives information about the sections, pages, modules and labels. Finally, listing files are produced which update the assembler listings with real addresses of symbols and statements.

This software program allows the user to develop modular programs, which may then be combined and addressed as defined by the user. The flexibility of the ST6 Linker is greatly increased by the use of sections allowing the user to group pieces of software from different modules. The location and the size of each section is user selectable.

ST6 SIMULATOR

- Window based graphic interface
- On line assembler/disassembler
- Supports symbolic debugging
- 128 breakpoints and 128 software traps
- TRACE mode
- I/O and CLOCK simulation

SIMST6 allows the user to debug and execute any program written for any of the current and future members of the ST6 family of microcontrollers, without the aid of additional hardware.

The user specifies the target device, its mapping and the object code file to be used. The simulator functionally duplicates the operation of the ST6 and completely supports the instruction set. I/O channels may be opened, read, and written, in order to simulate the I/O functions of peripherals, while interrupts may be set, and then set pending, in order to simulate the handling of interrupts. The simulator uses the clock frequency assigned by the user, along with the number of clock cycles needed by each instruction to keep track of the real time execution speed.

The ST6 Simulator accepts command lines in both interactive and batch mode.

ORDERING INFORMATION

Sales Type	Description
ST6-SW	ST6 software development tools (includes assembler, linker and emulator)

Note : The ST6 software package is included in all ST6xxx-EMU real time develoment tools.

SGS-THOMSON MICROELECTRONICS

ST6xxx-EMU

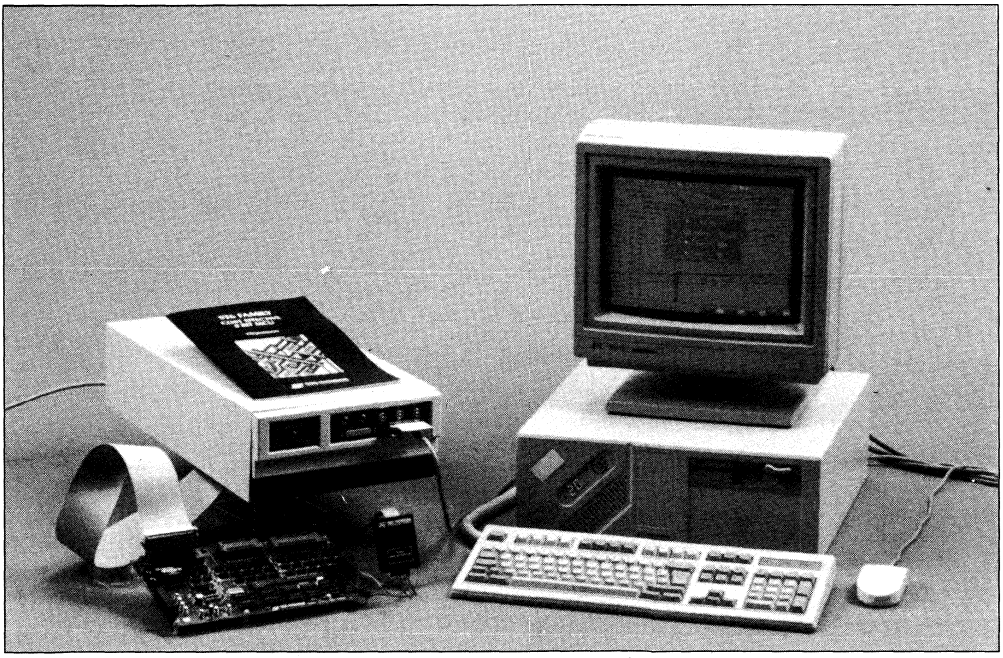
REAL TIME DEVELOPMENT TOOLS FOR ST6 MCU FAMILY

HARDWARE FEATURES

- Supports ST62xx and ST63xx family
- Real time emulation
- 32 KBytes of emulation memory
- Breakpoints on up to 256 events
- Events can be defined on program space, data space and on up to 4 external signals
- 1K of real trace memory
- Tracing of up to 32 bits including 4 external signals

SOFTWARE FEATURES

- Symbolic debugger
- Window based interface
- On line assembler/disassembler
- Log files capable of storing any displayed screen
- Command files able to execute a set of debugger commands



GENERAL DESCRIPTION

The ST6 Real Time Development System is an advanced hardware development system designed and configured to provide comprehensive support for the ST6 family of MCU's.

The mainframe consists of a basic part, common to all ST6 devices, and one (ST62 sub family) or two (ST63 sub family) dedicated board depending of specific device to emulate. Only the dedicated boards have to be changed to emulate a new device within the ST62/ST63 subfamilies.

The software part of the real time emulation tool is the symbolic debugger. It can be run on a PC compatible system and is common to all ST62/ST63 devices. It drives the emulator mainframe through an RS232 channel. The debugger uses a windowed menu driven user interface and enables the user to set the configuration of the emulator.

Once assembled and eventually linked and debugged by using the simulator, the application software is ready to be down loaded into the ST6-EMU. The device probe is connected into the application hardware. The development station will perform a real time emulation of the target device, thus allowing high performance test and debugging of both application hardware and software.

The breakpoints allow user to stop the MCU when the application software reaches selected addresses and/or addresses within a selected ranges and/or on data fetch (or read or write or both) cycles. The user is then able to read and modify any register and memory location. An on line assembler/disassembler is also available to ease the debugging.

The logic analyser can be used when real time emulation is needed. It allows to display the last 1024 cycles. The displayed cycles are either fetch cycles only or fetch cycles and data space accesses. Addresses, data, control/status bits and 4 user signals are displayed using mnemonic and user symbols.

Such a powerful tool enables the user to detect and trap any pattern and thus quickly debug the application. The trapping of random patterns is greatly improved by the capability to quit the emulation session while the emulator continue to run the application software. When the user re-enters the debugger, the emulation session resumes and information about any events of interest will be flashed to the screen in the form of a message.

Log files offer the ability to send any screen display to a text file. In particular, log files are very useful to save the contents of the logic analyser and/or the contents of data registers to be analysed or printed.

Command files can be used to execute a set of debugger commands in order to ease and speed up the emulation session.

A powerful help facility can be called at any time to give additional information about the commands, the processor or the emulator.

When the program is fully debugged, the ST6 EPROM remote programming board can be used to program the emulation device with the INTEL hex format file produced by the linker.

Figure 1. SDBST6 Command Summary

ALL	One Line Assembler
BASE	Change base of numbers
BREAK	Display/set breakpoint
CB	Clear breakpoints
CMP	Compare memory
DL	Display memory in listing ASM form
DM	Display/change memory
DOS	Branch to DOS
DR	Display/change registers
DS	Display symbol table
FM	Fill memory with pattern
GO	Start user program
GRAPH	Return to GRAPHIC interface
HELP	Call HELP utility
HWTEST	Execute diagnostic test
LOAD	Load memory from a file
LCONF	Load data pages configuration
MOVE	Move memory block
NEXT	Single/multi step mode
PM	Display/change paged Data ROM locations
QUIT	Abandon the program
RESET	Reset ST6 core dedications
SAVE	Save memory into a file
SB	Set address breakpoints
SCONF	Save data pages configuration
SEARCH	Search pattern in memory
REM	Put comment in a log file
SET	Set system options
SR	Set register
TRACE	Display traced execution
USE	Execute command file
WR	Display current Working Register set
UPLOAD	Copy ROMulator into HOST

ORDERING INFORMATION

Sales Type	Description
ST621X-EMU	Complete emulator package for ST621X/2X devices (including dedicated board and ST6-SW software package)
ST6240-EMU	Complete emulator package for ST6240 devices (including dedicated board and ST6-SW software package)
ST6242-EMU	Complete emulator package for ST6242 devices (including dedicated board and ST6-SW software package)
ST6245-EMU	Complete emulator package for ST6245 devices (including dedicated board and ST6-SW software package)
ST626X-EMU	Complete emulator package for ST626X and ST629X devices (including dedicated board and ST6-SW software package)
ST621X-DBE	Separate dedicated board for ST621X devices
ST624X-DBE	Separate dedicated board for ST624X devices
ST626X-DBE	Separate dedicated board for ST626X and ST629X devices
ST6240-PQFP	Probe for ST6240
ST6242-PQFP	Probe for ST6242
ST6245-PQFP	Probe for ST6245

Note : The emulator power supply can be adjusted to 220V or 110V.

SGS-THOMSON MICROELECTRONICS

ST62Exx-EPB

EPROM PROGRAMMING BOARD FOR ST62 MCU FAMILY

HARDWARE FEATURES

- Programs the ST62Exx EPROM and OTP MCUs
- Standalone and PC driven modes
- All ST62Exx packages are supported

SOFTWARE FEATURES

- Menu driven software
- S19 or INTEL hex file formats

DESCRIPTION

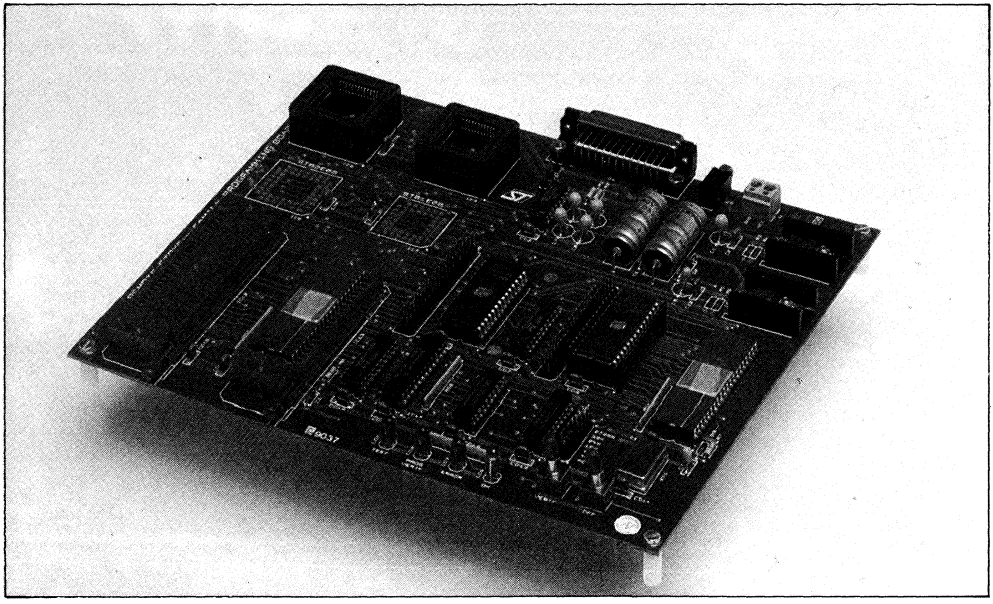
Different programming boards are designed for programming of the various EPROM and OTP devices of the ST62 sub-family. For a particular device, all available packages are supported by the same programming board.

It can run either in standalone or remote mode under control of a DOS compatible PC.

In standalone mode, the microcontrollers can be programmed with a simple key operation directly from a master EPROM device or a master microcontroller. Two colour LEDs indicate the operational pass or fail.

In standalone mode an EPROM memory or a master MCU is plugged into the programming board. The code from the EPROM or the master MCU is read and programmed into the ST62 EPROM or OTP device. Both VERIFY and BLANK CHECK functions are provided.

In remote mode, the programming board is connected to a DOS compatible PC through an RS232 serial channel. Object code in either S19 or INTEL HEX format is read from disk file to program the ST62 EPROM or OTP device. The menu driven software also offers VERIFY, BLANK CHECK, READ MASTER and other utility functions.



ORDERING INFORMATION

Sales Types [1]	Supported Devices	Supported Packages
ST62E1X- EPB/xxx	ST62E10 [2] ST62T10 [2] ST62E15 [2] ST62T15 [2] ST62E20 [2] ST62T20 [2] ST62E25 [2] ST62T25 [2]	DIP20 DIP28 SO20 SO28
ST62E4X-EPB/xxx	ST62E40 ST62E42 ST62E45 ST62T40 ST62T42 ST62T45	QFP52 QFP64 QFP80
ST62E6X-EPB/xxx	ST62E60 ST62E65 ST62T60 ST62T65 ST62E94 ST62T94	DIP20 SO20 DIP28 SO28 DIP20 DIP28

Notes :
1. ST62Exx-EPB/110 : 110V Power Supply
   ST62Exx-EPB/220 : 220V Power Supply
2. Both /HWD and /SWD options are supported

SGS-THOMSON MICROELECTRONICS

ST62Exx-GP

GANG PROGRAMMER FOR ST62 MCU FAMILY

HARDWARE FEATURES

- Programs simultaneously up to 10 ST62Exx EPROM and OTP MCUs
- Standalone and PC driven modes
- DIP and SO packages supported

SOFTWARE FEATURES

- Menu driven software
- S19 or INTEL hex file format

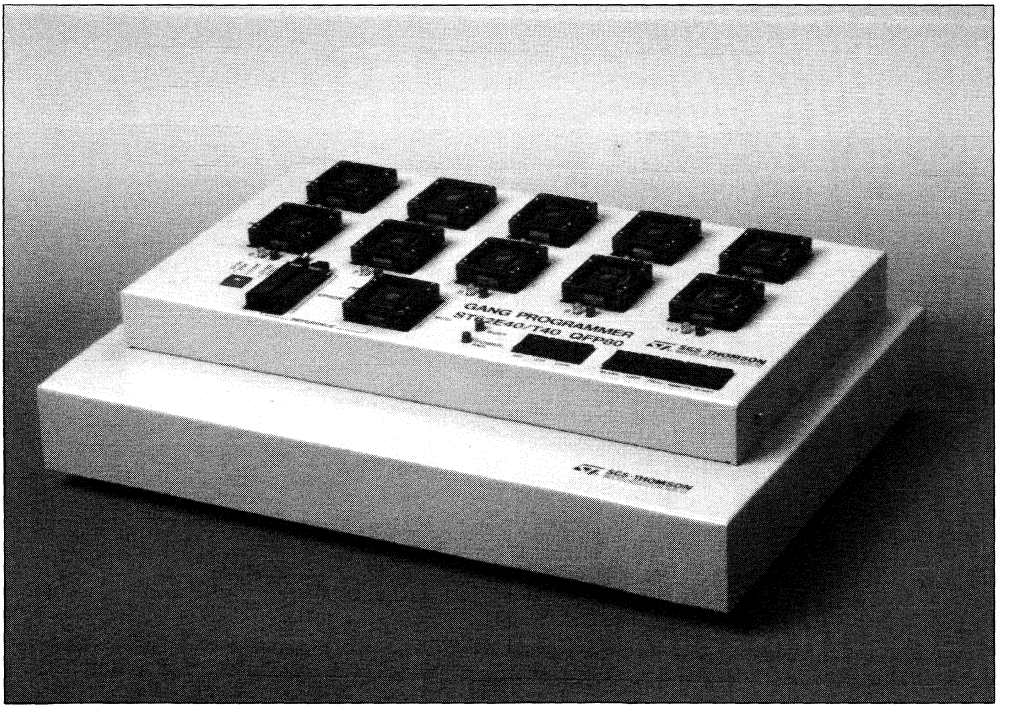
DESCRIPTION

The ST62 gang programmers are designed for programming up to 10 EPROM or OTP devices. It can run either in standalone or remote mode under control of a DOS compatible PC.

In standalone mode, the target ST62 MCUs are programmed with a simple key operation directly from a master EPROM memory or from a master EPROM MCU. Two color LEDs indicate for each target device the operational pass or fail. Both VERIFY and BLANK CHECK functions are provided.

In Remote mode, the gang programmer is connected to a DOS compatible PC through an RS232 serial channel. Object code in either S19 or INTEL HEX format is read from disk files to program the target devices. The menu driven software also offers VERIFY, BLANK CHECK, READ master and other utility functions.

The gang programmer is made up of a two parts, a base unit common to all ST62XX devices and a dedicated package adaptator.



ORDERING INFORMATION

Sales Types	Description	Supported Devices [1]	Supported Packages
ST62E10-GP/DIP	Gang Programmer	ST62E10 ST62T10 ST62E20 ST62T20	DIP20
ST62E10-GP/SO	Gang Programmer	ST62E10 ST62T10 ST62E20 ST62T20	SO20
ST62E15-GP/DIP	Gang Programmer	ST62E15 ST62T15 ST62E25 ST62T25	DIP28
ST62E15-GP/SO	Gang Programmer	ST62E15 ST62T15 ST62E25 ST62T25	SO28
ST62E40-GP/QFP	Gang Programmer	ST62E45 ST62T45	QFP52
ST62E42-GP/QFP	Gang Programmer	ST62E45 ST62T45	QFP52
ST62E45-GP/QFP	Gang Programmer	ST62E45 ST62T45	QFP52
ST62E65-GP/QFP	Gang Programmer	ST62E45 ST62T45	QFP52
ST62E65-GP/QFP	Gang Programmer	ST62E45 ST62T45	QFP52

Note 1. Both /HWD and /SWD options are supported.

Dedicated Gang Programmers (suffix /GP) are delivered including one dedication module (Suffix /GPA). Alternative device dedication modules are available under the sales types given above, but with /GP replaced by /GPA, allowing the use of another device type or package with the same Programmer.

PROGRAMMING MANUAL

PROGRAMMING MANUAL

INTRODUCTION

This manual deals with the description of the instruction set and addressing modes of ST62,63 microcontroller series. The manual is divided in two main sections. The first one includes, after a general family description, the addressing modes description. The second section includes the detailed description of ST62,63 instruction set. Each instruction is described in detail with the differences between each ST6 series.
ST6 software has been designed to fully use the hardware in the most efficient way possible while keeping byte usage to a minimum; in short to provide byte efficient programming capability.

PROGRAMMING MODEL

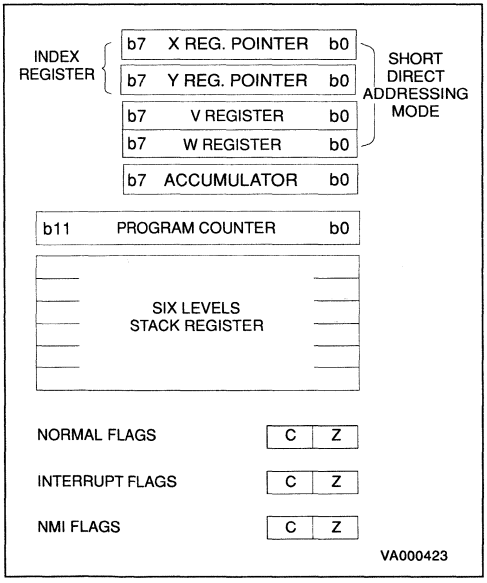
It is useful at this stage to outline the programming model of the ST62,63 series, by which we mean the available memory spaces, their relation to one another, the interrupt philosophy and so on.

Memory Spaces. The ST6 devices have three different memory spaces: data, program and stack. All addressing modes are memory space specific so there is no need for the user to specify which space is being used as in more complex systems. The stack space, which is used automatically with subroutine and interrupt management for program counter storage, is not accessible to the user.

Table 1. ST62,63 Series Core Characteristics

	ST62,63 Series
Stack Levels	6
Interrupt Vectors	5
NMI	YES
Flags Sets	3
Program ROM	2K + 2K• n 20K Max
Data RAM	64 byte • m
Data ROM	64 byte pages in ROM
Carry Flag SUB Instruction	Reset if A > Source
Carry Flag CP Instruction	Set if A < Source

Figure 1. ST6 Family Programming Model



PROGRAMMING MODEL (Continued)

Figure 2. ST62 Data Space Example

b7 b0	
NOT IMPLEMENTED	000h 03Fh
DATA ROM/EPROM WINDOW 64 BYTE	040h 07Fh
X REGISTER	080h
Y REGISTER	081h
V REGISTER	082h
W REGISTER	083h
DATA RAM 60 BYTES	084h 0BFh
PORT A DATA REGISTER	0C0h
PORT B DATA REGISTER	0C1h
PORT C DATA REGISTER	0C2h
RESERVED	0C3h
PORT A DIRECTION REGISTER	0C4h
PORT B DIRECTION REGISTER	0C5h
PORT C DIRECTION REGISTER	0C6h
RESERVED	0C7h
INTERRUPT OPTION REGISTER	0C8h
DATA ROM WINDOW REGISTER	0C9h
RESERVED	0CAh 0CBh
PORT A OPTION REGISTER	0CCh
PORT B OPTION REGISTER	0CDh
PORT C OPTION REGISTER	0CEh
RESERVED	0CFh
A/D DATA REGISTER	0D0h
A/D CONTROL REGISTER	0D1h
TIMER PSC REGISTER	0D2h
TIMER DATA REGISTER	0D3h
TIMER TSCR REGISTER	0D4h
RESERVED	0D5h 0D7h
WATCHDOG REGISTER	0D8h
RESERVED	0D9h 0FEh
ACCUMULATOR	0FFh

Figure 3. ST62 Program Memory Example

b7 b0	
NOT IMPLEMENTED	0000h 07FFh
RESERVED	0800h 087Fh
USER PROGRAM ROM 1828 BYTES	0880h 0F9Fh
RESERVED	0FA0h 0FEFh
INTERRUPT VECTOR #4 A/D INTERRUPT	0FF0h 0FF1h
INTERRUPT VECTOR #3 TIMER INTERRUPT	0FF2h 0FF3h
INTERRUPT VECTOR #2 PORT B & C INTERRUPT	0FF4h 0FF5h
INTERRUPT VECTOR #1 PORT A INTERRUPT	0FF6h 0FF7h
RESERVED	0FF8h 0FFBh
INTERRUPT VECTOR #0 NMI INTERRUPT	0FFCh 0FFDh
USER RESET VECTOR	0FFEh 0FFFh

On EPROM versions there are no reserved areas. These reserved bytes are present on ROM/OTP versions.

Data Memory Space. The following registers in the data space have fixed addresses which are hardware selected so as to decrease access times and reduce addressing requirements and hence program length. The Accumulator is an 8 bit register in location 0FFh. The X, Y, V & W registers have the addresses 80h-83h respectively. These are used for short direct addressing, reducing byte requirements in the program while the first two, X & Y, can also be used as index registers in the indirect addressing mode. These registers are part of the data RAM space. In the ST62 and ST63 for data space ROM a 6 bit (64 bytes addressing) window multiplexing in program ROM is available through a dedicated data ROM banking register.

PROGRAMMING MODEL (Continued)

For data RAM and I/O expansion the lowest 64 bytes of data space (00h-03Fh) are paged through a data RAM banking register.

Self-check Interrupt Vector FF8h & FF9h:
jp (self-check interrupt routine)

A jump instruction to the reset and interrupt routines must be written into these locations.

ST62 & ST63 Program Memory Space. The ST62 and ST63 devices can directly address up to 4K bytes (program counter is 12-bit wide). A greater ROM size is obtained by paging the lower 2K of the program ROM through a dedicated banking register located in the data space. The higher 2K of the program ROM can be seen as static and contains the reset, NMI and interrupt vectors at the following fixed locations:

Reset Vector FFEh & FFFh:
jp (reset routine)

NMI Interrupt Vector FFCh & FFDh:
jp (NMI routine)

Non user Vector FFAh & FFBh

Non user Vector FF8h & FF9h

Interrupt #1 Vector FF6h & FF7h jp (Int 1 routine)

Interrupt #2 Vector FF4h & FF5h jp (Int 2 routine)

Interrupt #3 Vector FF2h & FF3h jp (Int 3 routine)

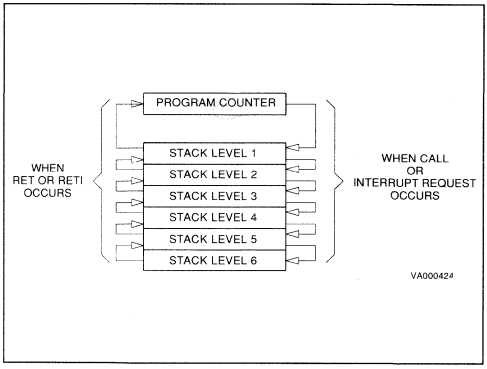
Interrupt #4 Vector FF0h & FF1h jp (Int 4 routine)

Program Counter & Stack Area. The program counter is a twelve bit counter register since it has to cover a direct addressing of 4K byte program memory space. When an interrupt or a subroutine occurs the current PC value is forward "pushed" into a deep LIFO stacking area. On the return from the routine the top (last in) PC value is "popped" out and becomes the current PC value. The ST60/61 series offer a 4-word deep stack for program counter storage during interrupt and sub-routines calls. In the ST62 and ST63 series the stack is 6-word deep.

Status Flags. Three pairs of status flags, each pair consisting of a Zero flag and a Carry flag, are available. In the ST62 and ST63 an additional third set is available. One pair monitors the normal status while the second monitors the state during interrupts; the third flags set monitors the status during Non Maskable interrupt servicing. The switching from one set to another one is automatic as the interrupt requests (or NMI request for ST62,ST63 only) are acknowledged and when the program returns after an interrupt service routine. After reset, NMI set is active, until the first RETI instruction is executed.

ST62 & ST63 Interrupt Description. The ST62 and ST63 devices have 5 user interrupt vectors (plus one vector for testing purposes). Interrupt vector #0 is connected to the not maskable interrupt input of the core. Interrupts from #1 to #4 can be connected to different on-chip and external sources (see individual datasheets for detailed information). All interrupts can be globally disabled through the interrupt option register. After the reset ST62 and ST63 devices are in NMI mode, so no other interrupts can be accepted and the NMI flags set is in use, until the RETI instruction is performed. If an interrupt is detected, a special cycle will be executed, during this cycle the program counter is loaded with the related interrupt vector address. NMI can interrupt other interrupt routines at any time while normal interrupt can't interrupt each other. If more then one interrupt is waiting service, they will be accepted according to their priority. Interrupt #1 has the highest priority while interrupt #4 the lowest. This priority relationship is fixed.

Figure 4. ST62/ST63 Stack Area

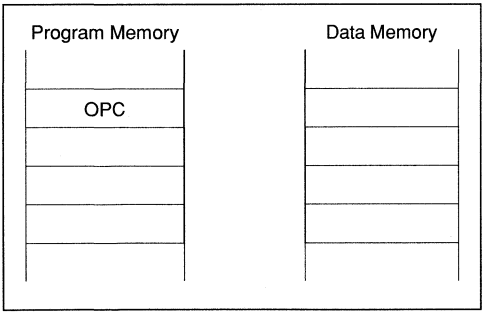


ADDRESSING MODES

The ST6 family gives the user nine addressing modes for access to data locations. Some of these are specifically tailored to particular instruction types or groups while others are designed to reduce program length and operating time by using the hardware facilities such as the X, Y, V & W registers. The data locations can be in either the program memory space or the data memory space when the ST6 is operating due to user software. In addition the ST6 has a stack space for the 12 bit program counter but this is controlled by internal programming and is not accessible by the user. This section will describe all the addressing modes which are provided to the user. The following is the complete list of the ST6 available addressing modes:

- Inherent
- Direct
- Short Direct
- Indirect
- Immediate
- Program Counter Relative
- Extended
- Bit Direct
- Bit Test & Branch

Inherent. For instructions using the inherent addressing mode the opcode contains all the information necessary for execution. All instructions using this mode are **One Byte** instructions.

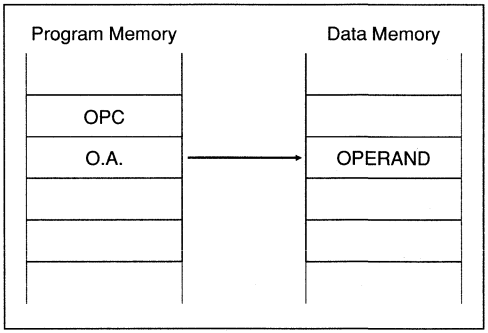


OPC = Opcode

Example:

Instruction	Comments
WAIT	Puts ST6 into the low power WAIT mode
STOP	Puts the ST6 into the lowest power mode
RETI	Returns from interrupt. Pops the PC from the PC stack.Sets the normal set of flags

Direct. In the direct addressing mode the address of the data is given by the program memory byte immediately following the opcode. This data location is in the data memory space. All instructions using this mode are **Two Bytes** instructions, lasting **Four Cycles**.



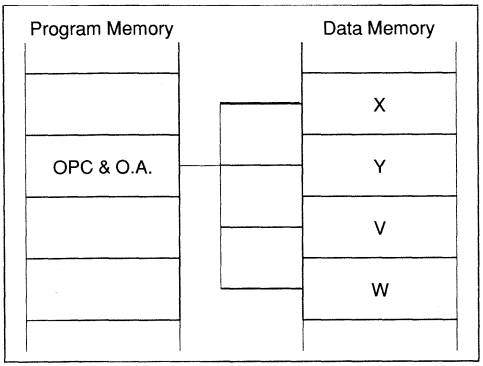
OPC = Opcode
O.A = Operand Address

Example:

Instruction	Comments
LD A,0A3h	Loads the accumulator with the value found in location A3h in the data space.
SUB A,11h	The value found in locations 11h in the data memory is subtracted from the value in the accumulator.

ADDRESSING MODES (Continued)

Short Direct. ST6 core has four fixed location registers in the data space which may be addressed in a short direct manner. The addresses and names of these registers are 80h (X), 81h (Y), 82h (V) and 83h (W). When using this addressing mode the data is in one of these registers and the address is a part of the opcode. All instructions using this mode are **One Byte** instructions, lasting **Four Cycles.**

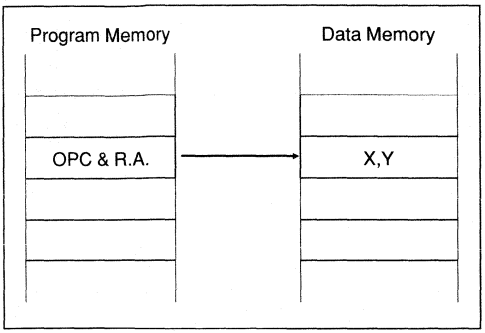


OPC = Opcode
O.A .= Operand Address

Example:

Instruction	Comments
LD A,X	The value of the X register (80h) is loaded into the accumulator.
INC X	The X register is incremented.

Indirect. The indirect mode must use either the X (80h) or Y (81h) register. This register contains the address of the data. The operand is at the data space address pointed to by the content of X or Y registers. All instructions using this mode are **One Byte** instructions, lasting **Four Cycles**.

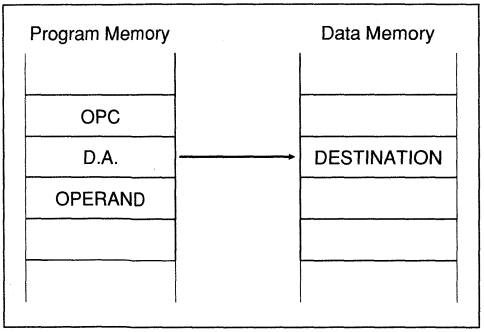


OPC = Opcode
R.A. = Register Address

Example:

Instruction	Comments
LD A,(X)	The value in the registers pointed to by the X register is loaded into the accumulator.
ADD A,(Y)	The value in the register pointed to by the Y register is added to the accumulator value.
INC (Y)	The value in the register pointed to by the Y register is incremented.

Immediate. In the immediate addressing mode the operand is found in the program ROM in a byte which is the last byte of the instruction. This addressing mode can be used for initializing data space registers and supplying constants. Instructions using this mode can be **Two** or **Three Bytes** instructions, lasting **Four Cycles**.



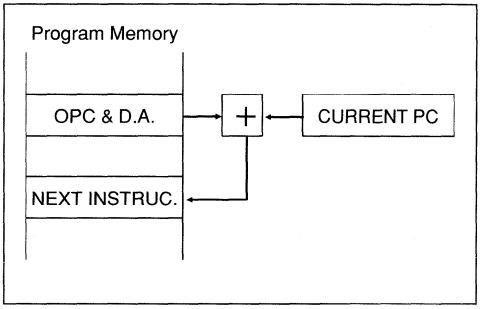
OPC = Opcode
D.A. = Destination Address

ADDRESSING MODES (Continued)

Example:

Instruction	Comments
LDI 34h,DFh	Loads immediate value DFh into data space location 34h.
SUBI A,22h	The immediate value 22h is substracted from the acc.

Program Counter Relative. This addressing mode is used only with conditional branches within the program. The opcode byte contains the data which is a fixed offset value. This offset is added to the program counter to give the address of the next instruction. The offset can have any value in the range -15 to +16. It is determined by the last five bits of the opcode. All instructions using this mode are **One Byte** Instructions, lasting **Two Cycles**.



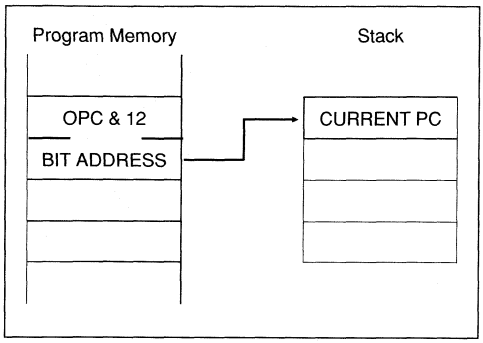
OPC = Opcode
D.A. = Destination Address

Example:

Instruction	Comments
JRC 3	If the carry flag is set then PC = PC+3
JRNZ -7	If the zero flag is not set (i.e the result of a previous instruction is not zero) then PC = PC-7

The relative jump address can be also a label that is automatically handled by the assembler.

Extended. The extended addressing mode is used to make long jumps within the program memory space (4K). The data requires 12 bits and is provided by half of the opcode byte and all of the second byte. All instructions using this mode are **Two Bytes** instructions, lasting **Four Cycles**.



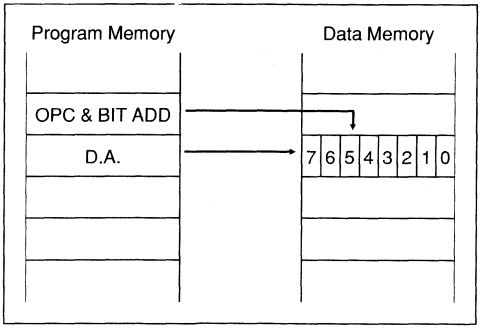
OPC = Opcode

Example:

Instruction	Comments
JP 3FAh	Loads 3FAh into program counter and continues with the instruction at 3FAh.
CALL ROU1	The current PC is pushed onto the stack and PC loaded with the value associated to the ROU1 label

The absolute jump address can be also a label that is automatically handled by the assembler.

Bit Direct. This addressing mode allows the user to set or clear any specified bit in a data memory register. The address of the bit is given in the form: "b,R" where b is the number of the bit and R is the address of the register. The bit is determined by three bits in the opcode and the register address is given by the second byte. All instructions using this mode are **Two Byte** instructions, lasting **Four Cycles**.



OPC = Opcode
D.A. = Destination Address

ADDRESSING MODES (Continued)

Example:

Instruction	Comments
SET 4,A	Sets bit 4 of the accumulator to 1.
RES 0,PORT	Clears bit 0 of PORT register

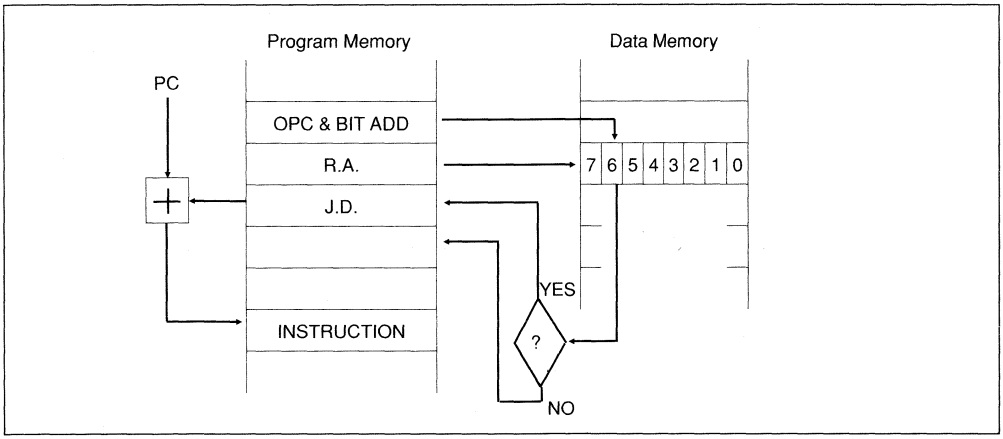
The register address can be associated to a label that is automatically handled by the assembler.

Bit Test & Branch. The bit test addressing mode is used in conditional jump instructions in which the jump depends on the result of a bit test. The opcode specifies the bit to be tested, the byte following the opcode in the register address in data space, and the third byte is the jump displacement, which is in the range -126 to +129. This displacement can be determined using a label, which is converted by the assembler. The state of the tested bit is also copied into the carry flag. All instructions using this mode are **Three Byte** instructions, lasting **Five Cycles**.

Example:

Instruction	Comments
JRS 3,PORT,LAB1	If bit three of data memory register associated to PORT label is set then PC=PC+LAB1 (where LAB1 is the jump displacement associated to a label
JRR 0,0Ah,-72	If bit 0 of data memory register OAh is reset to 0 then PC=PC-72.

The register address and the jump displacement can be associated to labels that are automatically handled by the assembler.



OPC = Opcode
R.A. = Relative Address
J.D. = Jump Displacement

ST62 & ST63 INSTRUCTION SET

The ST62,63 instructions can be divided functionally into the following seven groups.

- LOAD AND STORE
- ARITHMETIC AND LOGIC
- CONDITIONAL BRANCH
- JUMP AND CALL
- BIT MANIPULATION
- CONTROL
- IMPLIED

The following summary shows the instructions belonging to each group, the number of operands required for each instructions and the number of machine cycles. The flag behaviour is usually the same for both ST62 and ST63. The only difference is present for CP and SUB instructions as specified in the detailed description.

Note: For the following tables:

Δ: Affected

*: Not Affected

Table 2. Load & Store Instructions

Instruction	Bytes	Cycles	Flags	
			Z	C
LD	1	4	Δ	*
LD rr	2	4	Δ	*
LDI A	2	4	Δ	*
LDI	3	4	*	*

Table 3. Arithmetic & Logic Instructions

Instruction	Bytes	Cycles	Flags	
			Z	C
ADD	2	4	Δ	Δ
ADD (X,Y)	1	4	Δ	Δ
ADDI	2	4	Δ	Δ
AND	2	4	Δ	*
AND (X,Y)	1	4	Δ	*
ANDI	2	4	Δ	*
CLR A	2	4	Δ	Δ
CLR	3	4	*	*
COM	1	4	Δ	Δ
CP	2	4	Δ	Δ
CP (X,Y)	1	4	Δ	Δ
CPI	2	4	Δ	Δ
DEC	1	4	Δ	*
DEC A/rr	2	4	Δ	*
INC	1	4	Δ	*
INC A/rr	2	4	Δ	*
RLC	1	4	Δ	Δ
SLA	2	4	Δ	Δ
SUB	2	4	Δ	Δ
SUB (X,Y)	1	4	Δ	Δ
SUBI	2	4	Δ	Δ

Table 4. Conditional Branch Insructions

Instruction	Bytes	Cycles	Flags	
			Z	C
JRC	1	2	*	*
JRNC	1	2	*	*
JRR	3	5	*	Δ
JRS	3	5	*	Δ
JRZ	1	2	*	*
JRNZ	1	2	*	*

Table 5. Jump & Call Instructions

Instruction	Bytes	Cycles	Flags	
			Z	C
CALL	2	4	*	*
JP	2	4	*	*

ST62 & ST63 INSTRUCTION SET (Continued)

Table 6. Bit Manipulation Instructions

Instruction	Bytes	Cycles	Flags	
			Z	C
RES	2	4	*	*
SET	2	4	*	*

Table 7. Control Instructions

Instruction	Bytes	Cycles	Flags	
			Z	C
NOP	1	2	*	*
RET	1	2	*	*
RETI	1	2	Δ	Δ
STOP	1	2	*	*
WAIT	1	2	*	*

Table 8. Addressing Modes/Instruction Table

Instruction	Inh	Dir	Sh Dir	Ind	Imm	PCR	Ext	Bit Dir	Bit Test	Flags	
										Z	C
ADD		X	X	X						Δ	Δ
AND		X	X	X						Δ	*
CALL							X			*	*
CLR A		X								Δ	Δ
CLR		X								*	*
COM	X									Δ	Δ
CP		X		X	X					Δ	Δ
DEC		X	X	X						Δ	*
INC		X	X	X						Δ	*
JP							X			*	*
JRC, JRNC						X				*	*
JRZ, JRNZ						X				*	*
JRR, JRS									X	*	Δ
LD, LDI					X					Δ	*
NOP						X				*	*
RES, SET								X		*	*
RET	X									*	*
RETI	X									Δ	Δ
RLC	X									Δ	Δ
SLA	X									Δ	Δ
STOP, WAIT	X									*	*
SUB		X		X	X					Δ	Δ

Notes:

INH. Inherent, DIR: Direct, Sh.DIR: Short Direct,

IND. Indirect, IMM: Immediate, PCR: Program Counter Relative

EXT. Extended, BIT DIR: Bit Direct, BIT TEST.: Bit Test

Δ . Affected

* . Not Affected

ST62 & ST63 INSTRUCTION SET (Continued)

Table 9. Opcode Map

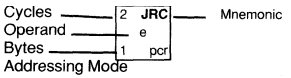
LOW / HI	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111	8 1000	9 1001	A 1010	B 1011	C 1100	D 1101	E 1110	F 1111	LOW / HI
0 0000	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b0,rr,ee 3 bt	2 JRZ e 1 pcr	#	2 JRC e 1 prc	4 LD a,(x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b0,rr 2 b.d	2 JRZ e 1 pcr	4 LDI rr,nn 3 imm	2 JRC e 1 pcr	4 LD a,(y) 1 ind	0 0000
1 0001	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b0,rr,ee 3 bt	2 JRZ e 1 pcr	4 INC x 1 sd	2 JRC e 1 prc	4 LDI a,nn 2 imm	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b0,rr 2 b.d	2 JRZ e 1 pcr	4 DEC x 1 sd	2 JRC e 1 pcr	4 LD a,rr 2 dir	1 0001
2 0010	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b4,rr,ee 3 bt	2 JRZ e 1 pcr	#	2 JRC e 1 prc	4 CP a,(x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b4,rr 2 b.d	2 JRZ e 1 pcr	4 COM a 1 inh	2 JRC e 1 pcr	4 CP a,(y) 1 ind	2 0010
3 0011	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b4,rr,ee 3 bt	2 JRZ e 1 pcr	4 LD a,x 1 sd	2 JRC e 1 prc	4 CPI a,nn 2 imm	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b4,rr 2 b.d.	2 JRZ e 1 pcr	4 LD x,a 1 sd	2 JRC e 1 pcr	4 CP a,rr 2 dir	3 0011
4 0100	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b2,rr,ee 3 bt	2 JRZ e 1 pcr	#	2 JRC e 1 prc	4 ADD a,(x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b2,rr 2 b.d	2 JRZ e 1 pcr	2 RETI 1 inh	2 JRC e 1 pcr	4 ADD a,(y) 1 ind	4 0100
5 0101	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b2,rr,ee 3 bt	2 JRZ e 1 pcr	4 INC y 1 sd	2 JRC e 1 prc	4 ADDI a,nn 2 imm	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b2,rr 2 b.d	2 JRZ e 1 pcr	4 DEC y 1 sd	2 JRC e 1 pcr	4 ADD a,rr 2 dir	5 0101
6 0110	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b6,rr,ee 3 bt	2 JRZ e 1 pcr	#	2 JRC e 1 prc	4 INC (x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b6,rr 2 b.d	2 JRZ e 1 pcr	2 STOP 1 inh	2 JRC e 1 pcr	4 INC (y) 1 ind	6 0110
7 0111	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b6,rr,ee 3 bt	2 JRZ e 1 pcr	4 LD a,y 1 sd	2 JRC e 1 prc	#	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b6,rr 2 b.d	2 JRZ e 1 pcr	4 LD y,a 1 sd	2 JRC e 1 pcr	4 INC rr 2 dir	7 0111
8 1000	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b1,rr,ee 3 bt	2 JRZ e 1 pcr	#	2 JRC e 1 prc	4 LD (x),a 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b1,rr 2 b.d	2 JRZ e 1 pcr	#	2 JRC e 1 pcr	4 LD (y),a 1 ind	8 1000
9 1001	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b1,rr,ee 3 bt	2 JRZ e 1 pcr	4 INC v 1 sd	2 JRC e 1 prc	#	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b1,rr 2 b.d	2 JRZ e 1 pcr	4 DEC v 1 sd	2 JRC e 1 pcr	4 LD rr,a 2 dir	9 1001
A 1010	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b5,rr,ee 3 bt	2 JRZ e 1 pcr	#	2 JRC e 1 prc	4 AND a,(x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b5,rr 2 b.d	2 JRZ e 1 pcr	4 RLC a 1 inh	2 JRC e 1 pcr	4 AND a,(y) 1 ind	A 1010
B 1011	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b5,rr,ee 3 bt	2 JRZ e 1 pcr	4 LD a,v 1 sd	2 JRC e 1 prc	4 ANDI a,nn 2 imm	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b5,rr 2 b.d	2 JRZ e 1 pcr	4 LD v,a 1 sd	2 JRC e 1 pcr	4 AND a,rr 2 dir	B 1011
C 1100	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b3,rr,ee 3 bt	2 JRZ e 1 pcr	#	2 JRC e 1 prc	4 SUB a,(x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b3,rr 2 b.d	2 JRZ e 1 pcr	2 RET 1 inh	2 JRC e 1 pcr	4 SUB a,(y) 1 ind	C 1100
D 1101	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b3,rr,ee 3 bt	2 JRZ e 1 pcr	4 INC w 1 sd	2 JRC e 1 prc	4 SUBI a,nn 2 imm	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b3,rr 2 b.d	2 JRZ e 1 pcr	4 DEC w 1 sd	2 JRC e 1 pcr	4 SUB a,rr 2 dir	D 1101
E 1110	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRR b7,rr,ee 3 bt	2 JRZ e 1 pcr	#	2 JRC e 1 prc	4 DEC (x) 1 ind	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b7,rr 2 b.d	2 JRZ e 1 pcr	2 WAIT 1 inh	2 JRC e 1 pcr	4 DEC (y) 1 ind	E 1110
F 1111	2 JRNZ e 1 pcr	4 CALL abc 2 ext	2 JRNC e 1 pcr	5 JRS b7,rr,ee 3 bt	2 JRZ e 1 pcr	4 LD a,w 1 sd	2 JRC e 1 prc	#	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 SET b7,rr 2 b.d	2 JRZ e 1 pcr	4 LD w,a 1 sd	2 JRC e 1 pcr	4 DEC rr 2 dir	F 1111

Abbreviations for Addressing Modes:

dir	Direct
sd	Short Direct
imm	Immediate
inh	Inherent
ext	Extended
b.d	Bit Direct
bt	Bit Test
pcr	Program Counter Relative
ind	Indirect

Legend:

#	Indicates Illegal Instructions
e	5 Bit Displacement
b	3 Bit Address
rr	1byte dataspace address
nn	1 byte immediate data
abc	12 bit address
ee	8 bit Displacement



ST62 & ST63 INSTRUCTION SET (Continued)

Table 10. Instruction Set Cycle-by-Cycle Summary

Instruction	Cycles	Cycles(#)	Address Bus	Data Bus	CPU Activity	Notes
Indirect Addressing Mode						
ADD, AND, CP, DEC, INC, LD, SUB	4	1 2 3 4	Opcode Address(*) Opcode Address +1 Opcode Address +1 Opcode Address +1	Opcode (*) Next Instruction Next Instruction Next Instruction	Decode Opcode Read Operand Address Read Operand Execute Instruction	ROM Data Space not Addressed
ADD, AND, CP, DEC, INC, LD, SUB	4	1 2 3 4	Opcode Address(*) Opcode Address +1 Opcode Address +1 Data Space Rom Add	Opcode (*) Next Instruction Next Instruction Rom Data (#)	Decode Opcode Read Operand Address Read Operand Execute Instruction	ROM Data Space Addressed
Direct Addressing Mode						
ADD, AND, CP, DEC, INC, LD, RES, SET, LSA, SUB, CLR	4	1 2 3 4	Opcode Address(*) Opcode Address +1 Opcode Address +1 (*) Opcode Address +2	Opcode (*) Operand Address Operand Address(*) Next Instruction	Decode Opcode Address Data Space Read Operand Execute Instruction	ROM Data Space not Addressed
ADD, AND, CP, DEC, INC, LD, RES, SET, LSA, SUB, CLR	4	1 2 3 4	Opcode Address(*) Opcode Address +1 Opcode Address +1 (*) Data Space Rom Add. (*)	Opcode (*) Operand Address Operand Address(#) Rom Data (#)	Decode Opcode Address Data Space Read Operand Execute Instruction	ROM Data Space Addressed
Immediate Addressing Mode						
ADDI, ANDI, CPI, LDI, SUBI	4	1 2 3 4	Opcode Address(*) Opcode Address +1 Opcode Address +1(*) Opcode Address +2(*)	Opcode (*) Immediate Operand Immediate Operand Next Instruction	Decode Opcode Idle Read Operand Execute Instruction	
LDI rr	4	1 2 3 4	Opcode Address(*) Opcode Address +1 Opcode Address +2 Opcode AdDress +3	Opcode (*) Register Address Immediate Operand Next Opcode	Decode Opcode Read Register Address Read Immediate Operand Write Operand To Reg.	ROM Data Space not Addressed
LDI rr	4	1 2 3 4	Opcode Address(*) Opcode Address +1 (*) Opcode Address +2 (#) Data Space Rom Add.	Opcode (*) Register Address Immediate Operand Rom Operand (#)	Decode Opcode Read Register Address Read Immediate Operand Write Operand To Reg.	ROM Data Space Addressed
Short Direct Addressing Mode						
DEC, INC, LD	4	1 2 3 4	Opcode Address(*) Opcode Address +1 Opcode Address +1 Opcode Address +1	Opcode (*) Next Opcode Next Opcode Next Opcode	Decode Opcode Define Data Space Add. Read Operand Execute Instruction	
Other Instructions						

Notes: *. Valid only at the beginning of the cycle

#. Valid only until t18 of the cycle

ST62 & ST63 INSTRUCTION SET (Continued)

Table 10. Instruction Set Cycle-by-Cycle Summary (Continued)

Instruction	Cycles	Cycles(#)	Address Bus	Data Bus	CPU Activity	Notes
CALL	4	1 2 3 4	Opcode Address(*) Opcode Address +1 Opcode Address +1 Opcode Address +2(*)	Opcode (*) Subroutine Address Subroutine Address Next Instruction	Decode Opcode Increment Stack Pointer Push Return Address Calculate Subroutine Add.	
COM	4	1 2 3 4	Opcode Address(*) Opcode Address +1 Opcode Address +1 Opcode Address +1	Opcode (*) Next Opcode Next Opcode Next Opcode	Decode Opcode Calculate Acc. Address Read Accumulator Complement Accumulator	
INTERRUPT	1	1	Next opcode address	Next Opcode (*)	Calculate Interrupt Add. Push Return Address Switch Flag Set	Note 1
JP	4	1 2 3 4	Opcode Address(*) Opcode Address +1 Opcode Address +1 Opcode Address +2	Opcode (*) Jump Address Following Instr. Following Instr. (*)	Decode Opcode Idle Read Jump Address Calculate Jump Address	
JRC, JRNC, JRZ, JRNZ	2	1 2	Opcode Address(*) Opcode Address +1	Opcode (*) Following Instr.	Decode Opcode Calculate Offset	
JRR, JRS	5	1 2 3 4 5	Opcode Address(*) Opcode Address +1(*) Opcode Address +2(*) Opcode Address +2(*) Opcode Address +3(*)	Opcode (*) Operand Address (*) Branch Value Branch Value (*) Following Instr.	Decode Opcode Read Operand Test Operand Fetch Branch Value Calculate New Address	ROM Data Space not Addressed
JRR, JRS	5	1 2 3 4 5	Opcode Address(*) Opcode Address +1(*) Data Space Rom Add.(#) Opcode Address +2(*) Data Space Rom Add.(#)	Opcode (*) Operand Address (*) Rom Data (#) Branch Value (*) Rom Data (#)	Decode Opcode Read Operand Test Operand Fetch Branch Value Calculate New Address	ROM Data Space Addressed
RET	2	1 2	Opcode Address(*) Return Address	Opcode (*) Next Opcode	Decode Opcode Pop Return Address	
RETI	2	1 2	Opcode Address(*) Return Address	Opcode (*) Next Opcode	Decode Opcode Pop Return Address Switch Flag Set	
RLC	4	1 2 3 4	Opcode Address(*) Opcode Address +1 Opcode Address +1 Opcode Address +1	Opcode (*) Next Opcode Next Opcode Next Opcode	Decode Opcode Calculate Acc. Address Read Accumulator Shifted	
STOP, WAIT	2	1 2	Opcode Address(*) Opcode Address +1	Opcode (*) Next Opcode	Decode Opcode Stop/Wait the Oscillator	

Notes: *. Valid only at the beginning of the cycle

#. Valid only until t18 of the cycle

1. Add oscillator build up time plus 16 oscillator clocks if a stop instruction has been executed before the interrupt occured

ADD
Addition

Mnemonic: ADD

Function: Addition

Description: The contents of the source byte is added to the accumulator leaving the result in the accumulator. The source register remains unaltered.

Operation: dst ← dst + src
The destination must be the accumulator.

Instruction Format	Opcode (Hex)	Bytes	Cycles	Flags	
ADD dst,src				Z	C
ADD A,A	5F FF	2	4	Δ	Δ
ADD A,X	5F 80	2	4	Δ	Δ
ADD A,Y	5F 81	2	4	Δ	Δ
ADD A,V	5F 82	2	4	Δ	Δ
ADD A,W	5F 83	2	4	Δ	Δ
ADD A,(X)	47	1	4	Δ	Δ
ADD A,(Y)	4F	1	4	Δ	Δ
ADD A,rr	5F rr	2	4	Δ	Δ

Notes:
rr.1 Byte dataspace address.
Δ:Z is set if the result is zero. Cleared otherwise.
C is cleared before the operation and than set if there is an overflow from the 8-bit result.

Example: If data space register 22h contains the value 33h and the accumulator holds the value 20h then the instruction,

ADD A,22h

will cause the accumulator to hold 53h (i.e. 33+20).

Addressing Modes: Source: Direct, Indirect
Destination: Accumulator

ADDI
Addition Immediate

Mnemonic: ADDI

Function: Addition Immediate

Description: The immediately addressed data (source) is added to the accumulator leaving the result in the accumulator.

Operation: dst ← dst + src
The destination must be the accumulator.

Instruction Format	Opcode (Hex)	Bytes	Cycles	Flags	
ADDI dst,src				Z	C
ADDI A,nn	57 nn	2	4	Δ	Δ

Notes:
nn.1 Byte immediate data
Δ: Z is set if result is zero. Cleared otherwise
C is cleared before the operation and than set if there is an overflow from the 8-bit result

Example: If the accumulator holds the value 20h then the instruction,

ADDI A,22h

will cause the accumulator to hold 42h (i.e. 22+20).

Addressing Modes: Source: Immediate
Destination: Accumulator

AND
Logical AND

Mnemonic: AND

Function: Logical AND

Description: This instruction logically ANDs the source register and the accumulator. The result is left in the destination register and the source is unaltered.

Operation: dst ←src AND dst
The destination must be the accumulator.

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
AND dst,src				Z	C
AND A,A	BF FF	2	4	Δ	*
AND A,X	BF 80	2	4	Δ	*
AND A,Y	BF 81	2	4	Δ	*
AND A,V	BF 82	2	4	Δ	*
AND A,W	BF 83	2	4	Δ	*
AND A,(X)	A7	1	4	Δ	*
AND A,(Y)	AF	1	4	Δ	*
AND A,rr	BF rr	2	4	Δ	*

Notes:
rr.1 Byte dataspace address

*.C is unaffected

Δ.Z is set if the result is zero. Cleared otherwise.

Example: If data space register 54h contains the binary value 11110000 and the accumulator contains the binary value 11001100 then the instruction,

AND A,54h

will cause the accumulator to be altered to 11000000.

Addressing Modes: Source: Direct, Indirect.
Destination: Accumulator

ANDI
Logical AND Immediate

Mnemonic: ANDI

Function: Logical AND Immediate

Description: This instruction logically ANDs the immediate data byte and the accumulator. The result is left in the accumulator.

Operation: dst ← src AND dst
The source is immediate data and the destination must be the accumulator.

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
ANDI dst,src				Z	C
ANDI A,nn	B7 nn	2	4	Δ	*

Notes:
nn.1 Byte immediate data

*.C is unaffected

Δ. Z is set if the result is zero. Cleared otherwise.

Example: If the accumulator contains the binary value 00001111 then the instruction,

ANDI A,33h

will cause the accumulator to hold the value 00000011.

Addressing Modes: Source: Immediate

Destination: Accumulator

CALL
Call Subroutine

Mnemonic: CALL

Function: Call Subroutine

Description: The CALL instruction is used to call a subroutine. It "pushes" the current contents of the program counter (PC) onto the top of the stack. The specified destination address is then loaded into the PC and points to the first instruction of a procedure. At the end of the procedure a RETurn instruction can be used to return to the original program flow. RET pops the top of the stack back into the PC. Because the ST6 stack is 4 levels deep (ST60) and 6 levels deep (ST62,ST63), a maximum of four/six calls or interrupts may be nested. If more calls are nested, the PC values stacked latest will be lost. In this case returns will return to the PC values stacked first.

Operation: PC ← dst; Top of stack ← PC

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
CALL dst				Z	C
CALL abc	c0001 ab	2	4	*	*

Notes:
abc.the three half bytes of a twelve bit address, the start location of the subroutine.

*. C,Z not affected

Example: If the current PC is 345h then the instruction,

CALL 8DCh

The current PC 345h is pushed onto the top of the stack and the PC will be loaded with the value 8DCh. The next instruction to be executed will be the instruction at 8DCh, the first instruction of the called subroutine.

Addressing Modes: Extended

CLR
Clear

Mnemonic: CLR

Function: Clear

Description: The destination register is cleared to 00h.

Operation: dst ← 0

Inst. Format	OPCDE (Hex)	Bytes	Cycles	Flags	
CLR dst				Z	C
CLR A	DF FF	2	4	Δ	Δ
CLR X	0D 80 00	3	4	*	*
CLR Y	0D 81 00	3	4	*	*
CLR V	0D 82 00	3	4	*	*
CLR W	0D 83 00	3	4	*	*
CLR rr	0D rr 00	3	4	*	*

Notes:
rr. 1 Byte dataspace address

Δ. C,Z set

*. C,Z unaffected

Example: If data space register 22h contains the value 33h,

CLR 22h

will cause register 22h to hold 00h.

Addressing Modes: Direct

COM
Complement

Mnemonic: COM

Function: Complement

Description: This instruction complements each bit of the accumulator; all bits which are set to 1 are cleared to 0 and vice-versa.

Operation: dst ← NOT dst
The destination must be the accumulator.

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
COM dst				Z	C
COM A	2D	1	4	Δ	Δ

Note :

Δ:Z is set if the result is zero. Cleared otherwise.

C will contain the value of the MSB before the operation.

Example: If the accumulator contains the binary value 10111001 then the instruction

COM A

will cause the accumulator to be changed to 01000110 and the carry flag to be set (since the original MSB was 1).

Addressing Modes: Inherent

CP
Compare

Mnemonic: CP

Function: Compare

Description: This instruction compares the source byte (subtracted from) with the destination byte, which must be the accumulator. The carry and zero flags record the result of this comparison.

Operation: dst - src
The destination must be the accumulator, but it will not be changed.

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
CP dst,src				Z	C
CP A,A	3F FF	2	4	Δ	Δ
CP A,X	3F 80	2	4	Δ	Δ
CP A,Y	3F 81	2	4	Δ	Δ
CP A,V	3F 82	2	4	Δ	Δ
CP A,W	3F 83	2	4	Δ	Δ
CP A,(X)	27	1	4	Δ	Δ
CP A,(Y)	2F	1	4	Δ	Δ
CP A,rr	3F rr	2	4	Δ	Δ

Note: rr. 1 Byte dataspace address

ST60 Δ: Z is set if the result is zero. Cleared otherwise.

C is set if Acc ≥ src, cleared if Acc < src.

ST62/63 Δ: Z is set if the result is zero. Cleared otehrwise.

C is set if Acc < src, cleared if Acc ≥ src.

Example: If the accumulator contains the value 11111000 and the register 34h contains the value 00011100 then the instruction,

CP A,34h

will clear the Zero flag Z and set the Carry flag C, indicating that Acc ≥ src (on ST60

CPI
Compare Immediate

Mnemonic: CPI

Function: Compare Immediate

Description: This instruction compares the immediately addressed source byte (subtracted from) with the destination byte, which must be the accumulator. The carry and zero flags record the result of this comparison.

Operation: dst-src

The source must be the immediately addressed data and the destination must be the accumulator, that will not be changed.

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
CPI dst,src				Z	C
CPI A,nn	37 nn	2	4	Δ	Δ

Note: nn.1 Byte immediate data.

ST60 Δ: Z is set if the result is zero. Cleared otherwise.

C is set if Acc ≥ src, cleared if Acc < src.

ST62/63 Δ: Z is set if the result is zero. Cleared otherwise.

C is set if Acc < src, cleared if Acc ≥ src.

Example: If the accumulator contains the value 11111000 then the instruction,

CPI A,00011100B

will clear the Zero flag Z and set the Carry flag C indicating that Acc ≥ src (on ST60).

Addressing Modes: Source: Immediate

Destination: Accumulator

DEC
Decrement

Mnemonic: DEC

Function: Decrement

Description: The destination register's contents are decremented by one.

Operation: dst ← dst-1

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
DEC dst				Z	C
DEC A	FF FF	2	4	Δ	*
DEC X	1D	1	4	Δ	*
DEC Y	5D	1	4	Δ	*
DEC V	9D	1	4	Δ	*
DEC W	DD	1	4	Δ	*
DEC (X)	E7	1	4	Δ	*
DEC (Y)	EF	1	4	Δ	*
DEC rr	FF rr	2	4	Δ	*

Notes:
rr.1 Byte dataspace address

*.C is unaffected

Δ.Z is set if the result is zero. Cleared otherwise.

Example: If the X register contains the value 45h and the data space register 45h contains the value 16h then the instruction,

DEC (X)

will cause data space register 45h to contain the value 15h.

Addressing Modes: Short direct, Direct, Indirect.

INC
Increment

Mnemonic: INC

Function: Increment

Description: The destination register's contents are incremented by one.

Operation: dst ← dst+1

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
INC dst				Z	C
INC A	7F FF	2	4	Δ	*
INC X	15	1	4	Δ	*
INC Y	55	1	4	Δ	*
INC V	95	1	4	Δ	*
INC W	D5	1	4	Δ	*
INC (X)	67	1	4	Δ	*
INC (Y)	6F	1	4	Δ	*
INC rr	7F rr	2	4	Δ	*

Notes:

rr.1 Byte dataspace address

*. C is unaffected

Δ.Z is set if the result is zero. Cleared otherwise.

Example: If the X register contains the value 45h and the data space register 45h contains the value 16h then the instruction

INC (X)

will cause data space register 45h to contain the value 17h.

Addressing Modes: Short direct, Direct, Indirect.

JP
Jump

Mnemonic: JP

Function: Jump (Unconditional)

Description: The JP instruction replaces the PC value with a twelve bit value thus causing a simple jump to another location in the program memory. The previous PC value is lost, not stacked.

Operation: PC ← dst

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
JP dst				Z	C
JP abc	c1001 ab	2	4	*	*

Notes:

abc.the three half bytes of a twelve bit address.

*. C,Z not affected

Example: The instruction,

JP 5CDh

will cause the PC to be loaded with 5CDh and the program will continue from that location.

Addressing Modes: Extended

JRC
Jump Relative on Carry Flag

Mnemonic: JRC

Function: Jump Relative on Carry Flag

Description: This instruction causes the carry (C) flag to be tested and if this flag is set then a jump is performed within the program memory. This jump is in the range -15 to +16 and is relative to the PC value. The displacemente is of five bits. If C=0 than the next instruction is executed.

Operation: If C=1, PC ← PC + e
where e= 5 bit displacement

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
				Z	C
JRC e	e110	1	2	*	*

Notes:

e.5 bit displacement in the range –15 to + 16

*.C,Z not affected

Example: If the carry flag is set then the instruction,

JRC + 8

will cause a branch forward to PC+8. The user can use labels as indentifiers and the assembler will automatically allow the jump if it is in the range -15 to +16.

Addressing Modes: Program Counter Relative

JRNC
Jump Relative on Non Carry Flag

Mnemonic: JRNC

Function: Jump Relative on Non Carry Flag

Description: This instruction causes the carry (C) flag to be tested and if this flag is cleared to zero then a jump is performed within the program memory. This jump is in the range -15 to +16 and is relative to the PC value. The dispacement is of five bits. If C=1 then the next instruction is executed.

Operation: If C=0, PC ← PC + e
where e= 5 bit displacement

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
				Z	C
JRNC e	e010	1	2	*	*

Notes:

e:5 bit displacement in the range -15 to +16

*:C,Z not affected

Example: If the carry flag is cleared then the instruction,

```
JRNC -5
```

will cause a branch backward to PC-5. The user can use labels as identifiers and the assembler will automatically allow the jump if it is in the range -15 to +16.

Addressing Modes: Program Counter Relative

JRNZ

Jump Relative on Non Zero Flag

Mnemonic: JRNZ

Function: Jump Relative on Non Zero Flag

Description: This instruction causes the zero (Z) flag to be tested and if this flag is cleared to zero then a jump is performed within the program memory. This jump is in the range -15 to +16 and is relative to the PC value. The displacement is of five bits. If Z=1 then the next instruction is executed.

Operation: If Z=0, PC ← PC + e
where e= 5 bit displacement

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
				Z	C
JRNZ e	e000	1	2	*	*

Notes:

e.5 bit displacement in the range -15 to +16.

*.C,Z not affected

Example: If the zero flag is cleared then the instruction,

```
JRNZ -5
```

will cause a branch backward to PC-5. The user can use labels as identifiers and the assembler will automatically allow the jump if it is in the range -15 to +16.

Addressing Modes: Program Counter Relative

JRR
Jump Relative if Reset

Mnemonic: JRR

Function: Jump Relative if RESET

Description: This instruction causes a specified bit in a given dataspace register to be tested. If this bit is reset (=0) then the PC value will be changed and a relative jump will be performed within the program. The relative jump range is -126 to +129. If the tested bit is not reset then the next instruction is executed.

Operation: If bit=0, PC ← PC + ee
where ee= 8 bit displacement

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
				Z	C
JRR b,rr,ee	b00011 rr ee	3	5	*	Δ

Notes:

b.3 bit-address

rr.1 Byte dataspace address

ee.8 bit displacement in the range -126 to +129

*.Z is not affected

Δ.The tested bit is shifted into carry.

Example: If bit 4 of dataspace register 70h is reset and the PC=110 then the instruction,

JRR 4, 70h, -20

will cause the PC to be changed to 90 (110-20) and the instruction starting at that address in the program memory to be the next instruction executed.

The user is advised to use labels for conditional jumps. The relative jump will be calculated by the assembler. The jump must be in the range -126 to +129.

Addressing Modes: Bit Test

JRS
Jump Relative if Set

Mnemonic: JRS

Function: Jump Relative if set

Description: This instruction causes a specified bit in a given dataspace register to be tested. If this bit is set (=1) then the PC value will be changed and a relative jump will be performed within the program. The relative jump range is -126 to +129. If the tested bit is not set then the next instruction is executed.

Operation: If bit=1, PC ← PC + ee
where ee= 8 bit displacement

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
				Z	C
JRS b,rr,ee	b10011 rr ee	3	5	*	Δ

Notes:

b.3 bit-address

rr.1 Byte dataspace address

ee.8 bit displacement in the range -126 to +129

*.Z is not affected

Δ.The tested bit is shifted into carry.

Example: If bit 7 of dataspace register AFh is set and the PC=123 then the instruction,

JRS 7,AFh,+25

will cause the PC to be changed to 148 (123+25) and the instruction starting at that address in the program memory to be the next instruction executed.

The user is advised to use labels for conditional jumps. The relative jump will be calculated by the assembler. The jump must be in the range -126 to +129.

Addressing Modes: Bit Test

JRZ
Jump Relative on Zero Flag

Mnemonic: JRZ

Function: Jump Relative on Zero Flag

Description: This instruction causes the zero (Z) flag to be tested and if this flag is set to one then a jump is performed within the program memory. This jump is in the range -15 to +16 and is relative to the PC value. The displacement is of five bits. If Z=0 then next instruction is executed.

Operation: If Z=1, PC ← PC + e
where e= 5 bit displacement

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
				Z	C
JRZ e	e100	1	2	*	*

Notes:

e.5 bit displacement in the range -15 to +16.

*.C,Z not affected

Example: If the zero flag is set then the instruction,

```
JRZ +8
```

will cause a branch forward to PC+8. The user can use labels as identifiers and the assembler will automatically allow the jump if it is in the range -15 to +16.

Addressing Modes: Program Counter Relative

LD
Load

Mnemonic: LD

Function: Load

Description: The contents of the source register are loaded into the destination register. The source register remains unaltered and the previous contents of the destination register are lost.

Operation: dst ← src
Either the source or the destination must be the accumulator.

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
LD dst,src				Z	C
LD A,X	35	1	4	Δ	*
LD A,Y	75	1	4	Δ	*
LD A,V	B5	1	4	Δ	*
LD A,W	F5	1	4	Δ	*
LD X,A	3D	1	4	Δ	*
LD Y,A	7D	1	4	Δ	*
LD V,A	BD	1	4	Δ	*
LD W,A	FD	1	4	Δ	*
LD A,(X)	07	1	4	Δ	*
LD (X), A	87	1	4	Δ	*
LD A,(Y)	0F	1	4	Δ	*
LD (Y),A	8F	1	4	Δ	*
LD A,rr	1F rr	2	4	Δ	*
LD rr,A	9F rr	2	4	Δ	*

Notes:

rr.1 Byte dataspace address

*.C not affected

Δ.Z is set if the result is zero. Cleared otherwise.

Example: If data space register 34h contains the value 45h then the instruction;

```
LD A,34h
```

will cause the accumulator to be loaded with the value 45h. Register 34h will keep the value 45h.

LDI
Load Immediate

Mnemonic: LDI

Function: Load Immediate

Description: The immediately addressed data (source) is loaded into the destination data space register.

Operation: dst ← src

The source is always an immediate data while the destination can be the accumulator, one of the X,Y,V,W registers or one of the available data space registers.

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
LDI dst,src				Z	C
LDI A,nn	17 nn	2	4	Δ	*
LDI X,nn	0D 80 nn	3	4	*	*
LDI Y,nn	0D 81 nn	3	4	*	*
LDI V,nn	0D 82 nn	3	4	*	*
LDI W,nn	0D 83 nn	3	4	*	*
LDI rr,nn	0D rr nn	3	4	*	*

Notes:

rr.1 Byte dataspace address

nn.1 Byte immediate value

*.Z, C not affected

Δ.Z is set if the result is zero. Cleared otherwise.

Example: The instruction

LDI 34h,45h

will cause the value 45h to be loaded into data register at location 34h.

Addressing Modes: Source: Immediate

Destination: Direct

NOP
No Operation

Mnemonic: NOP

Function: No Operation

Description: No action is performed by this instruction. It is typically used for timing delay.

Operation: No Operation

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
				Z	C
NOP	04	1	2	*	*

Note: *. C,Z not affected

Addressing Modes: Program Counter Relative

RES
Reset Bit

Mnemonic: RES

Function: Reset Bit

Description: The RESET instruction is used to reset a specified bit in a given register in the data space.

Operation: dst (n) ← 0, 0 ≤ n ≤ 7

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
RES bit,dst				Z	C
RES b,A	b01011 FF	2	4	*	*
RES b,rr	b01011 rr	2	4	*	*

Notes:

b.3 bit-address

rr.1 Byte dataspace address

*.C,Z not affected

Example: If register 23h of the dataspace contains 11111111 then the instruction,

RES 4,23h

will cause register 23h to hold 11101111.

Addressing Modes: Bit Direct

RET
Return from Subroutine

Mnemonic: RET

Function: Return From Subroutine

Description: This instruction is normally used at the end of a subroutine to return to the previously executed procedure. The previously stacked program counter (stacked during CALL) is popped back from the stack. The next statement executed is that addressed by the new contents of the PC. If the stack had already reached its highest level (no more PC stacked) before the RET is executed, program execution will be continued at the next instruction after the RET.

Operation: PC ← Stacked PC

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
				Z	C
RET	CD	1	2	*	*

Note: *. C,Z not affected

Example: If the current PC value is 456h and the PC value at the top of the stack is 3DFh then the instruction,

RET

will cause the PC value 456h to be lost and the current PC value to be 3DFh.

Addressing Modes: Inherent

RETI
Return from Interrupt

Mnemonic: RETI

Function: Return from Interrupt

Description: This instruction marks the end of the interrupt service routine and returns the ST60/62/63 to the state it was in before the interrupt. It "pops" the top (last in) PC value from the stack into the current PC. This instruction also causes the ST60/62/63 to switch from the interrupt flags to the normal flags. The RETI instruction also applies to the end of NMI routine for ST62/63 devices; in this case the instruction causes the switch from NMI flags to normal flags (if NMI was acknowledged inside a normal routine) or to standard interrupt flags (if NMI was acknowledged inside a standard interrupt service routine).

In addition the RETI instruction also clears the interrupt mask (also NMI mask for ST62/63) which was set when the interrupt occurred. If the stack had already reached its highest level (no more PC stacked) before the RETI is executed, program execution will be continued with the next instruction after the RETI. Because the ST60 is in interrupt mode after reset (NMI mode for ST62/63), RETI has to be executed to switch to normal flags and enable interrupts at the end of the starting routine. If no call was executed during the starting routine, program execution will continue with the instruction after the RETI (supposed no interrupt is active).

Operation: Actual Flags ← Normal Flags (1)
PC ← Stacked PC
IM ← 0
(1) Standard Interrupt flags if NMI was acknowledged inside a standard interrupt service (ST62/63 only).

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
				Z	C
RETI	4D	1	2	Δ	Δ

Note: Δ C,Z normal flag will be used from now on.

Example: If the current PC value is 456h and the PC value at the top of the stack is 3DFh then the instruction

RETI

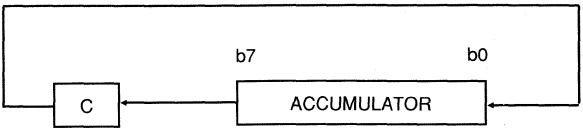
RLC
Rotate Left Through Carry

Mnemonic: RLC

Function: Rotate Left through Carry

Description: This instruction moves each bit in the accumulator one place to the left (i.e. towards the MSBit. The MSBit (bit 7) is moved into the carry flag and the carry flag is moved into the LSBit (bit0) of the accumulator.

Operation:



dst(0) ← C
C ←dst(7)
dst(n+1) ← dst(n), 0 ≤ n ≤ 6
This instruction can only be performed on the accumulator.

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
				Z	C
RLC A	AD	1	4	Δ	Δ

Note : Δ: Z is set if the result is zero. Cleared otherwise.
C will contain the value of the MSB before the operation.

Example: If the accumulator contains the binary value 10001001 and the carry flag is set to 0 then the instruction,

RLC A

will cause the accumulator to have the binary value 00010010 and the carry flag to be set to 1.

Addressing Modes: Inherent

SET
Set Bit

Mnemonic: SET

Function: Set Bit

Description: The SET instruction is used to set a specified bit in a given register in the data space.

Operation: dst (n) ← 1, $0 \leq n \leq 7$

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
SET bit,dst				Z	C
SET b,A	b11011 FF	2	4	*	*
SET b,rr	b11011 rr	2	4	*	*

Notes:

b. 3 bit-address

rr. 1 Byte dataspace address

*. C,Z not affected

Example: If register 23h of the dataspace contains 00000000 then the instruction,

SET 4,23h

will cause register 23h to hold 00010000.

Addressing Modes: Bit Direct

SLA
Shift Left Accumulator

Mnemonic: SLA

Function: Shift Left Accumulator

Description: This instruction implements an addition of the accumulator to itself (i.e a doubling of the accumulator) causing an arithmetic left shift of the value in the register.

Operation: ADD A,FFh
This instruction can only be performed on the accumulator.

Inst. Format	OPCPDE (Hex)	Bytes	Cycles	Flags	
				Z	C
SLA A	5F FF	2	4	Δ	Δ

Note: Δ: Z is set if the result is zero. Cleared otherwise.
C will contain the value of the MSB before the operation.

Example: If the accumulator contains the binary value 11001101 then the instruction,

SLA A

will cause the accumulator to have the binary value 10011010 and the carry flag to be set to 1.

Addressing Modes: Inherent

STOP
Stop Operation

Mnemonic: STOP

Function: Stop operation

Description: This instruction is used for putting the ST60/62/63 into a stand-by mode in which the power consumption is reduced to a minimum. All the on-chip peripherals and oscillator are stopped (for some peripherals,A/D for example, it is necessary to individually turn-off the macrocell before entering the STOP instruction). To restart the processor an external interrupt or a reset is needed.

Operation: Stop Processor

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
				Z	C
STOP	6D	1	2	*	*

Note : *: C,Z not affected

Addressing Mode: Inherent

SUB
Subtraction

Mnemonic: SUB

Function: Subtraction

Description: This instruction subtracts the source value from the destination value.

Operation: dst ←dst-src
The destination must be the accumulator.

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
SUB dst,src				Z	C
SUB A,A	DF FF	2	4	Δ	Δ
SUB A,X	DF 80	2	4	Δ	Δ
SUB A,Y	DF 81	2	4	Δ	Δ
SUB A,V	DF 82	2	4	Δ	Δ
SUB A,W	DF 83	2	4	Δ	Δ
SUB A,(X)	C7	1	4	Δ	Δ
SUB A,(Y)	CF	1	4	Δ	Δ
SUB A,rr	DF rr	2	4	Δ	Δ

Note: rr.1 Byte dataspace address

ST60 Δ: Z is set if the result is zero. Cleared otherwise.

C is set if Acc ≥ src, cleared if Acc < src.

ST62/63 Δ: Z is set if the result is zero. Cleared otherwise.

C is set if Acc < src, cleared if Acc ≥ src.

Example: If the Y register contains the value 23h, dataspace register 23h contains the value 53h and the accumulator contains the value 78h then the instruction,

SUB A,(Y)

will cause the accumulator to hold the value 25h (i.e. 78-53). The zero flag is cleared and the carry flag is set (on ST60), indicating that result is > 0.

Addressing Modes: Source: Indirect,Direct

SUBI
Subtraction Immediate

Mnemonic: SUBI

Function: Subtraction Immediate

Description: This instruction causes the immediately addressed source data to be subtracted from the accumulator.

Operation: dst ← dst - src
The destination must be the accumulator.

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
SUBI dst,src				Z	C
SUBI A,nn	D7 nn	2	4	Δ	Δ

Note: nn. 1 Byte of immediate data

ST60 Δ: Z is set if the result is zero. Cleared otherwise.

C is set if Acc ≥ src, cleared if Acc < src.

ST62/63 Δ: Z is set if the result is zero. Cleared otherwise.

C is set if Acc < src, cleared if Acc ≥ src.

Example: If the accumulator contains the value 56h then the instruction,

SUBI A,25

will cause the accumulator to contain the value 31h. The zero flag is cleared and the carry flag is set (on ST60), indicating that the result is > 0.

Addressing Modes: Source: Immediate

Destination: Accumulator

WAIT

Wait Processor

Mnemonic: WAIT

Function: Wait Processor

Description: This instruction is used for putting the ST60/62/63 into a stand-by mode in which the power consumption is reduced to a minimum. Instruction execution is stopped, but the oscillator and some on-chip peripherals continue to work. To restart the processor an interrupt from an active on-chip peripheral (eg. timer), an external interrupt or reset is needed. For on-chip peripherals active during wait, see ST60/62/63 data sheets.

Operation: Put ST6 in stand-by mode

Inst. Format	OPCODE (Hex)	Bytes	Cycles	Flags	
				Z	C
WAIT	ED	1	2	*	*

Note : *. C,Z not affected

Addressing Modes: Inherent

APPLICATION NOTES

SGS-THOMSON MICROELECTRONICS

APPLICATION NOTE

MICROCONTROLLER AND TRIACS ON THE 110/240V MAINS

Philippe RABIER/Laurent PERIER

INTRODUCTION

Today, electronics is used in home appliances for purposes as different as the motor regulation of a washing machine, the control of a vacuum cleaner, the light dimming of a lamp or the heating in a coffee machine. This pervasion increases rapidly because appliances require enhanced features, easy to built and modify while electronics based solutions become cheaper and more sophisticated.

Within this evolution, the microcontrollers (MCU) progressively replace analog controllers and discrete solutions even in low cost applications. They are more flexible, often need less components and provide faster time to market. With an analog IC, the designer is limited to a fixed function frozen inside the device. With a DIAC control, features like sensor feedback or enhanced motor drive can not be easily implemented. With the MCU proposed in this note (the ST6210), the designer can include his own ideas and test them directly using EPROM or One Time Programmable (OTP) versions.

The triac is the least expensive power switch to operate directly on the 110/240V mains. Thus it is the optimal switch for most of the low-cost power applications operating on-line. The LOGIC LEVEL or SNUBBERLESS triacs are a complement to the ST6210 MCU for such appliances. These triacs can operate with low gate current and can be directly triggered by the MCU, while still maintaining a high switching capability.

This application note describes three different MCU based applications: a universal motor drive, an AC switch and a light dimmer. They all operate with the same user interfaces and almost the same software and hardware.

UNIVERSAL MOTOR DRIVE

Basic function

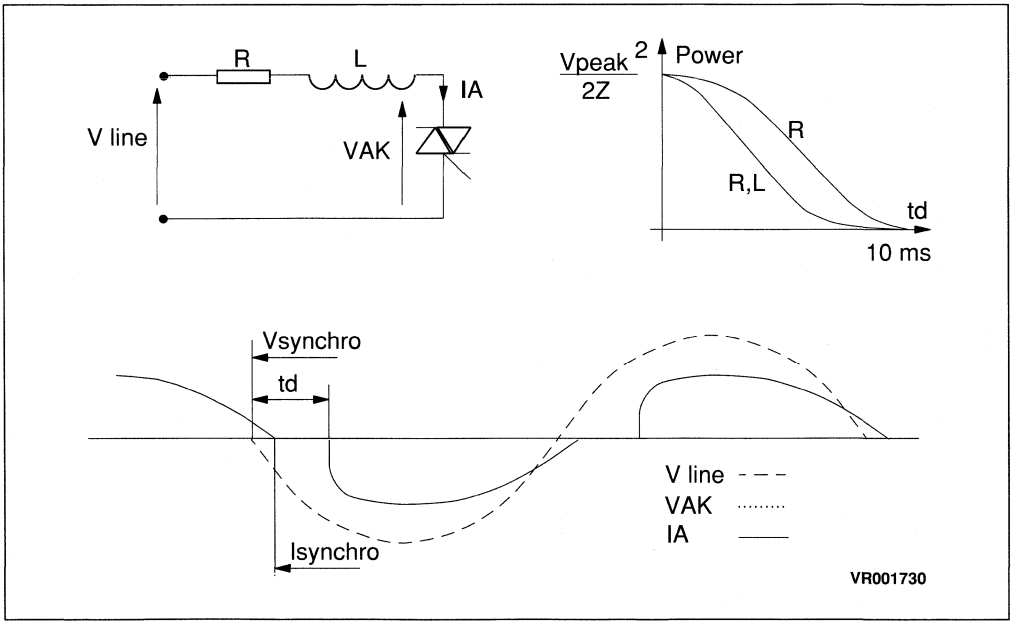
Universal motor drives with DIAC or analog controllers are currently used today. These circuits have the disadvantage of requiring many external components when controls include sophisticated features such as speed control with torque limiting or when parameters have to be easily changed from one design to an other. They are also limited in the choice of user interfaces.

A universal motor drive circuit, supplied directly from the 110V/240V mains has been realized using a MCU ST6210 and a SNUBBERLESS triac.The user interface is a touch sensor, a push button or a potentiometer. The board includes a minimum of components in order to save cost and size. The auxiliary supply is derived from the mains voltage.

Power control

The power device is a triac because it is the most economical on-line switch. The output power, and therefore the motor speed, are controlled by the phase delay of the triac drive. This delay is referred to the zero crossing of the line voltage which is detected by means of a connection to the mains neutral (fig.1). Changing operation from 60Hz to 50Hz can be achieved by making simple modifications to the MCU EPROM/ROM table defining the triac conduction angle versus power level. Automatic selection of the 50Hz/ 60Hz tables could be done.

Figure 1. Mains synchronisation



A universal motor is an inductive load which may generate very strong dynamic constraints on the circuit at turn-off. Because of the phase lag of current with respect to voltage, the reapplied voltage can be different from zero. So the leading edge of the reapplied voltage across the triac can be very sharp because it is limited only by parasitic capacitances (fig.8). A SNUBBERLESS triac is well adapted to this kind of loads because fast commutation characteristics can be obtained with low current rating devices. Otherwise, inrush current at motor startup is limited by the soft start feature included in the control.

Triac drive

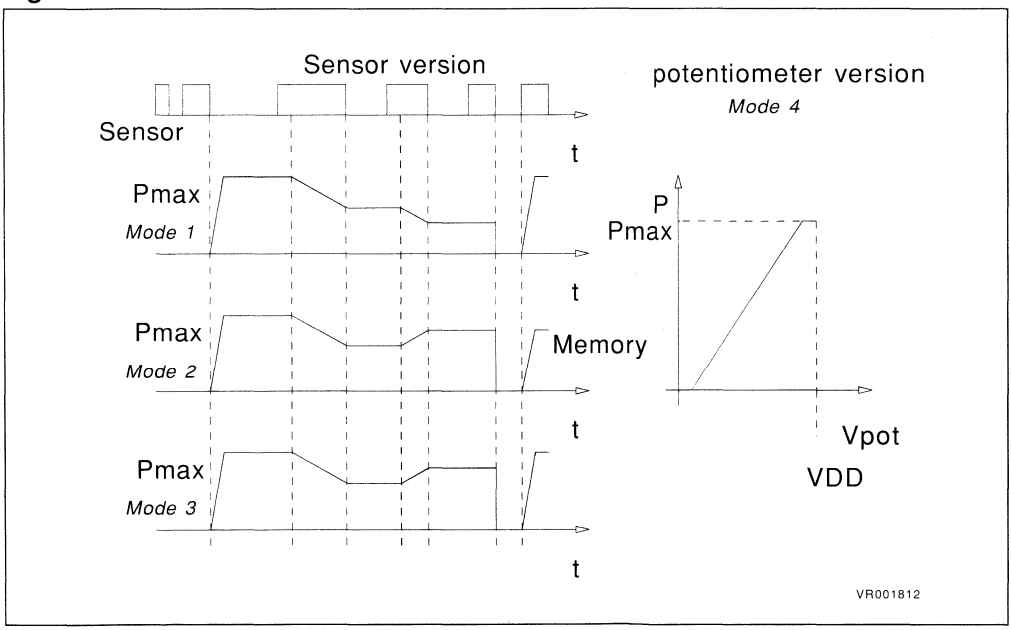
The triac is directly driven by the MCU. The pulse driving the triac is short (100s) in order to minimize the +5V supply circuit size. The SNUBBERLESS triac is driven in quadrants QII and QIII with 60 mA gate current provided by three I/O bits of the ST6210 in parallel. This pulse is sufficiently long to insure the triac is latched at the end of the pulse. Pulse length can be modified if another triac or motor is used.

User Interfaces

There are three different user interfaces: a touch control, a push button and a potentiometer. Four modes can be selected on the board in order to define how the transmitted power is related to the user interface.

Three modes operate with the touch sensor or the push button. Dimming is obtained when the sensor or the button is touched for more than 330ms. If the touch duration is between 50ms and 330ms, the circuit is switched on or off. A contact of less than 50ms causes no action. Modes 1,2,3 differ in the way the motor speed is changed by

Figure 2. User interfaces



sensor or button contact. (These values are given for the 60Hz version, for a 50Hz mains values are respectively 400ms and 50ms).

Mode 4 directly relates transmitted power to potentiometer position (fig.2). All modes include a soft start.

When operating together, the SNUBBERLESS triac and the ST6210 MCU save components on the drive, logic power supply, mains connection, and the power side (fig.3).

The MCU chosen (ST6210) includes an 8 bit accumulator, 2k ROM, 64 bytes RAM, an 8 bit A/D converter that can be connected to 8 different inputs, 4 I/O lines with 10mA sink current capability and a timer. Hysteresis protection is included in series with each I/O pin. The ST6210 is packaged in DIL or SMD packages. The ports, the timer and interrupts configurations can be chosen by software, providing high flexibility. The ST6210 has been designed to operate in very disturbed environments. Each I/O line contains internal diodes which clamp the input voltage between Vdd and Vss. These diodes are sized to withstand a continuous current of 1mA (typ.). With EPROM and OTP versions, the equipment development and preproduction can be done directly from the design lab providing a fast time to market.

The SNUBBERLESS triac (BTA 16-400CW for 110 V, BTA10-600CW for 220V) has been specially designed to drive loads which generate very strong dynamic con-

Figure 3. Motor drive circuit diagram

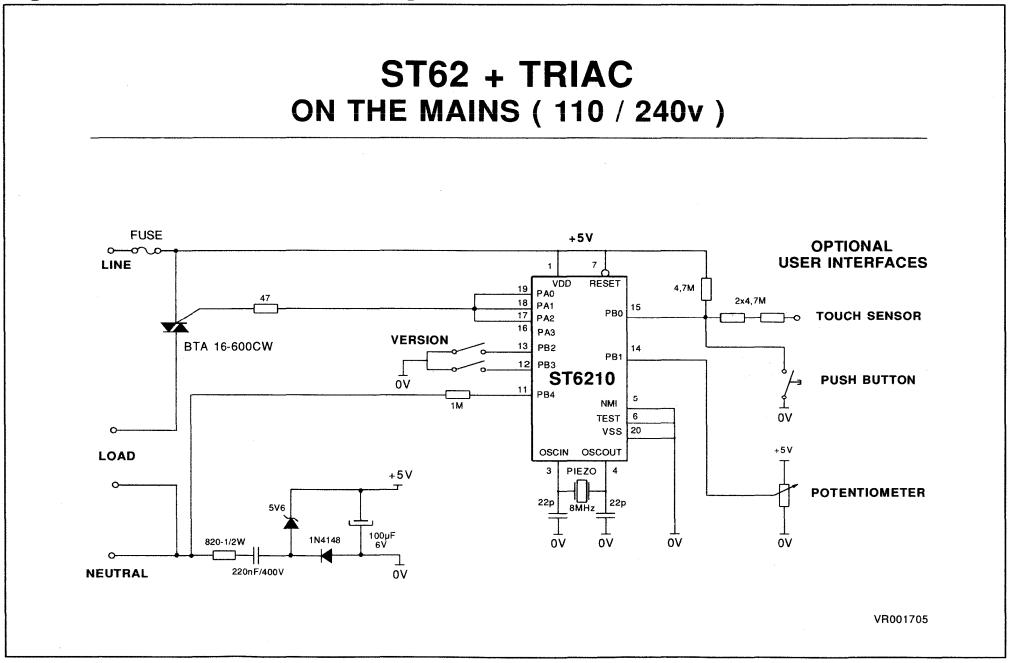
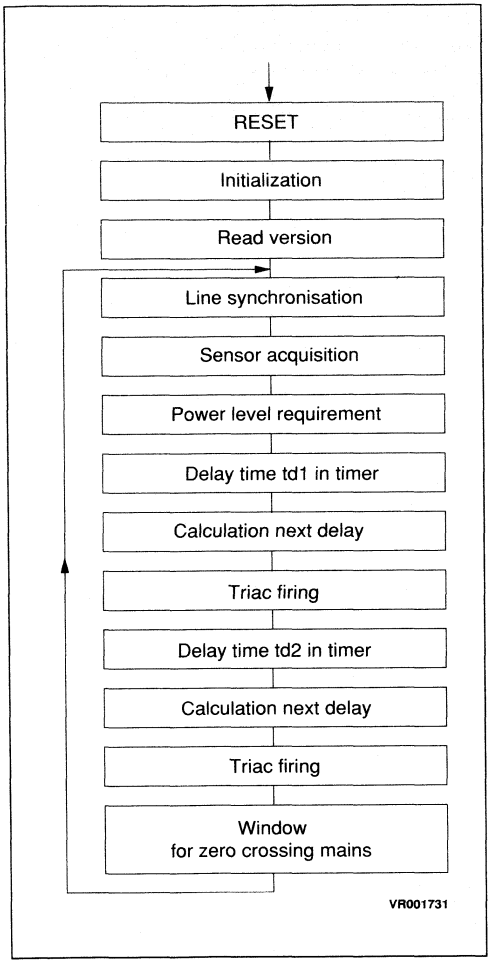


Figure 4. Major steps of the software



straints such as a vacuum cleaner motor. This triac can be triggered in quadrants QI, QII or QIII with gate and latching current of 35mA and 80mA respectively. In this application it is driven by three I/O lines of the ST6210 in parallel. This triac has high current switching capability ($[dI/dt]_c$>8.5A/ms and 5.5A/ms for BTA10- 600CW), and high static dv/dt ([dV/dt]>250V/µs). So, in this circuit, it can operate without a snubber.

Total consumption of the board is 3mA with an 8MHz oscillator. The board supply comes from the mains through a simple RCD circuit. The +5V is referred to anode 1 of the triac in order to provide the negative gate current necessary to drive the triac in quadrants QII and QIII. The 5V supply capacitance is mounted as near as possible to the MCU with very short interconnecting traces to maximize RFI immunity.

The touch sensor is a voltage divider between line and neutral. It operates when the +5V supply input of the circuit is connected to the line potential. This connection to the mains must be provided.

Software

All operating features are contained in a 700 byte program. More than 1byte of ROM is available for additional features. The architecture of the software is modular in order to provide maximum flexibility.

A lockup table relating delay time to the power requirement contains 64 different levels. The conduction time of the triac can vary from 1.7ms to 6.7ms for a 60Hz application and from 2ms to 8ms to a 50Hz application. The user can easily adjust the minimum and maximum power levels because the corresponding delay times are slowly changing at the top and bottom of the table. The table can

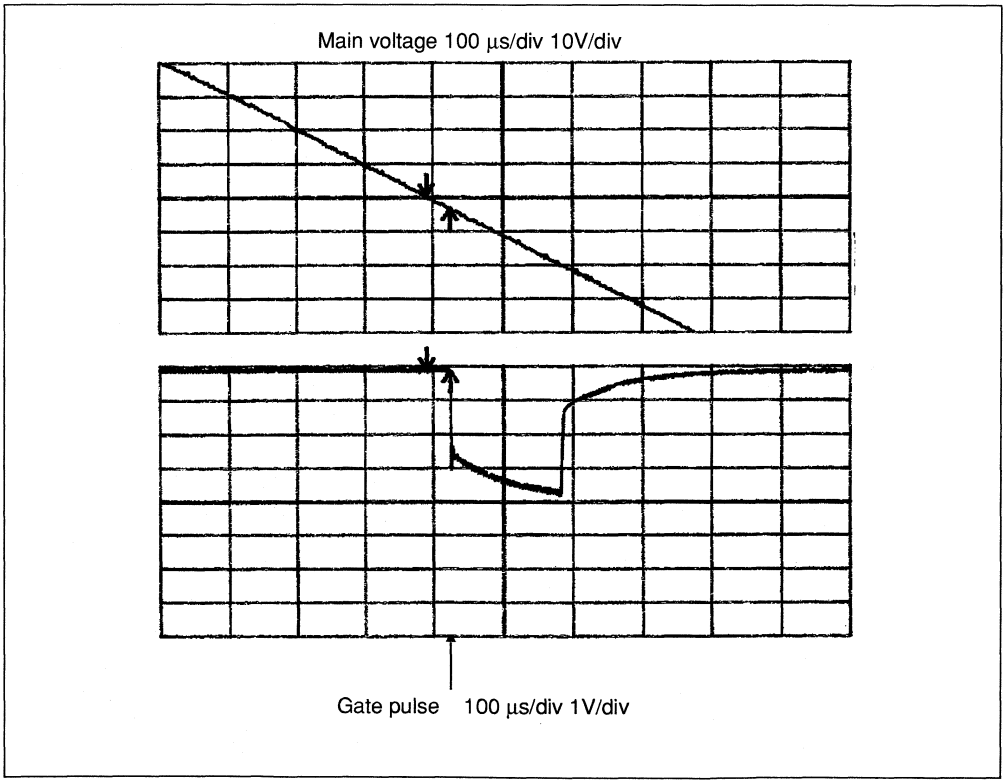
be modified in ROM/EPROM to meet different conditions e.g. 50Hz or 60Hz operation or varying loads.

One software version covers all four user interface modes without hardware change.

All inputs are digitally filtered so that an input is validated only if it remains constant for 15s or more. So, passive filter components can be saved. The mains supply carries disturbances (glitches, telecommand signals, ...) which could disturb the triac drive. For this reason, a mains voltage zero crossing is only validated if it occurs during a window of time (1.7ms each 16.6ms for 60Hz operation and 2ms each 200ms for 50Hz operation) selected by the internal timer of the MCU. This block acts as a filter and again eliminates external components (fig.4).

This circuit can be used as a basis for development of more sophisticated features such as vacuum regulation in a vacuum cleaner, speed control in a food processor, speed regulation with torque limiting in a drill, unbalance detection in a washing machine or door opener with remote control.

Figure 5. Delay time in drive of an AC switch



AC SWITCH

AC switches operate as relays. They have to turn-on as soon as possible after a mains voltage zero crossing in order to prevent the disturbances induced by a sharp leading edge of current. Their operating current has to be small in order to minimize the supply coming from the mains. Also, they are usually driven through isolated interfaces such as an optocoupler or transformer.

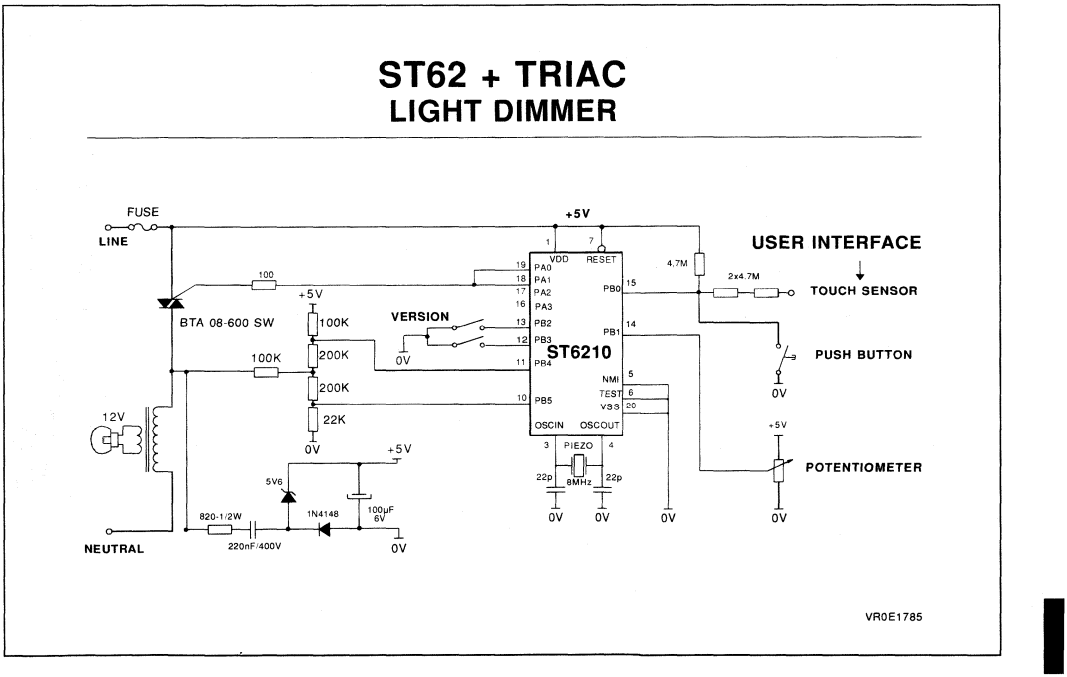
The motor drive circuit described previously can be used as a basis for such applications. When the zero crossing of the mains voltage is detected, this triac is turned-on.

A SNUBBERLESS triac such as the BTA 16-600BW is suitable for an AC switch. Its switching characteristics ($[dI/dt]_c$>14A/ms) allows it to control various types of resistive and inductive loads.

The figure 5 shows that a ST6210 can turn-on a triac 35µs after the mains voltage zero crossing. With such phase delay (0.75°/60Hz and 0.63°/50Hz), the voltage reapplied across the load is small (V) and the leading edge of current is minimal.

Such an AC switch can include additional features such as voltage and current monitoring or feedback features. This AC switch can also be used, for instance, in a refrigerator (with smaller triacs) with several compartments where the MCU controls different temperatures. The MCU can then interface the sensors, solve the priority conflicts and drive the AC switches with the optimal sequence.

Figure 6. Light dimmer circuit diagram



LIGHT DIMMER

Basic function

The motor drive board can also operate as a light dimmer. However, the board can be slightly modified such that the neutral connection is no longer necessary. Then, the board can be plugged in series with the line wire like a mechanical switch. The synchronization and the auxiliary supply are obtained from the voltage across the triac (Fig.6).

A light dimmer operating directly on the 110V or 240V mains has been realized using a MCU ST6210 and a LOGICLEVEL triac. This circuit drives halogen or incandescent lamps supplied directly from the mains or through a low voltage transformer. It includes softstart and protection against transformer saturation and against open load. The user interfaces are the same as with the motor drive.

Power control

Power is controlled by the phase delay (td) of the triac drive. In the previous design, td is referred to the zero crossing of the line voltage. In order to avoid a connection to the mains neutral and connect the circuit directly in series with the load, the trigger delay is referred to the previous zero crossing of the current (fig.1). When the current in the triac is zero, the mains voltage is reapplied across it. Synchronization is achieved by measuring this voltage. This main voltage is monitored over each halfwave with a network of resistances connected to two I/O lines of the ST6210. This allows detection of spurious open load and the retriggering of the triac with multipulse operation if it is not latched after the first gate pulse.

Operation with a transformer

Low power halogen spots use low voltage lamps (12V typ.) usually supplied through a low voltage transformer. Dimming these lamps is simple with this circuit thanks to the program features included in the ST6210 :

* At the start, the delay time between the first gate pulse and the synchronization instant is greater than 5ms. This limits induction in the transformer and the risk of saturation.

* The circuit starts on a positive line halfwave and stops on a negative one. Thus it starts with positive induction and stops after negative induction has been applied. This helps to minimize the size of the magnetic core material, and the current rating of the triac.

* The timer is precisely tuned in order to obtain 8.3ms (for 60Hz) or 10ms (for 50Hz) delay between two gate pulses. As a result, the triac is driven symmetrically in both phases so that continuous voltage in the transformer is avoided and noise in the transformer is reduced. Otherwise, the voltage across the triac is monitored to detect a spurious open load condition at the secondary of the transformer.

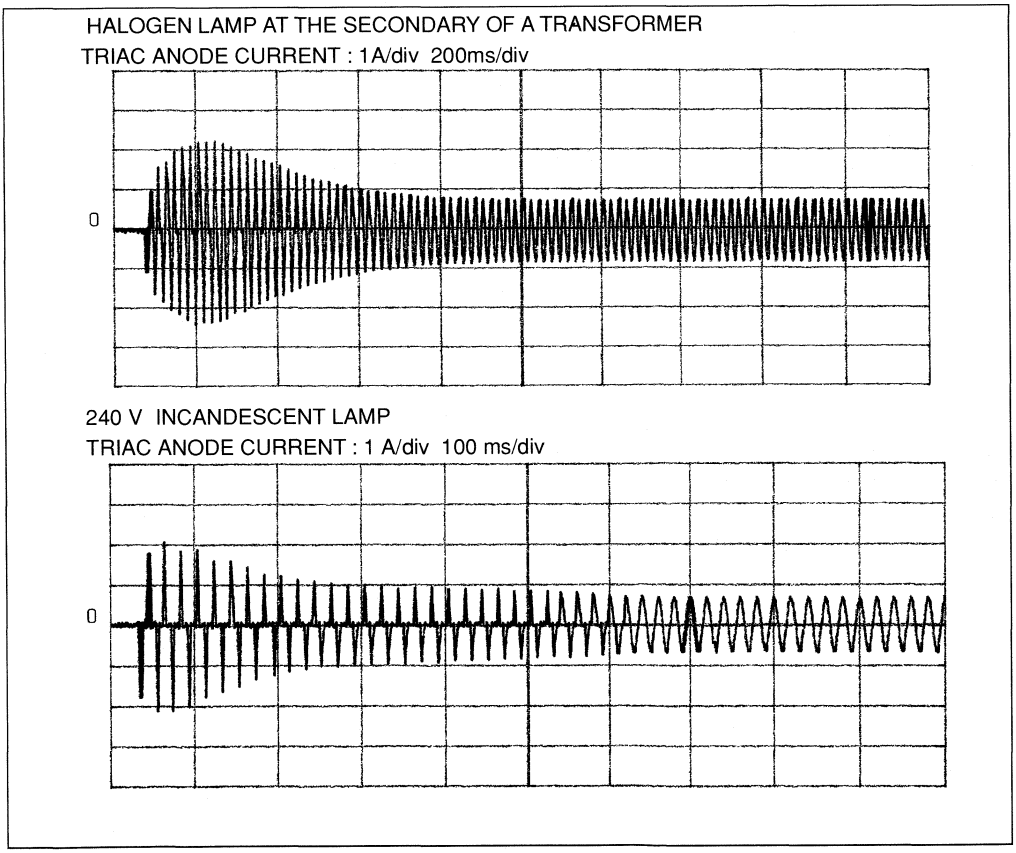
The inrush current at the turn-on of a lamp (halogen or incandescent) is also reduced due to the soft start feature of the circuit (fig.7).

Triac drive

The triac is directly driven by the MCU. The pulse driving the triac is 50µs long. The LOGIC LEVEL triac is driven in quadrants QII and QIII with a gate current of 20mA provided by two I/O lines of the ST6210 in parallel. The LOGICLEVEL triac has a maximum specified gate triggering current of 10mA at 25°C.

The triac is multi-pulse driven. Therefore, inductive loads can be driven without the use of long pulse drives. As a result, the consumption on the +5V supply can be minimized and the supply circuit becomes very small. Before supplying the first drive pulse, the triac voltage is tested. If no voltage is detected, a spurious open load or a supply disconnection is assumed to have occurred and the circuit is stopped. After the first driving pulse, the triac voltage is monitored. If the triac is not ON, another pulse is sent. The same process can be repeated up to four times. Then, if the triac is still not ON, the circuit is switched off.

Figure 7. Soft start with lamps



Hardware

The light dimmer board is almost the same as the motor drive board (fig.6). The major differences concern the position where the voltage is measured and the triac choice. When the board is dimming a resistive load, an RFI filter should be added in order to the RFI standards (i.e. VDE 875).

In a dimmer, because of the resistive load, dynamic constraints are lower than in a motor control, so a LOGIC LEVEL triac (BTA08-400SW or BTA08-600SW) is used for safe gate current. This triac has been especially designed to operate with a MCU. It is a sensitive triac (I_{GT}<10mA) which can be triggered in quadrants I, II and III. This triac has high switching capabilities ($[dI/dt]_c$>3.5A/ms), ($[dV/dt]_c$>20V/µs). Thus it can also operate without a snubber in this circuit.

This board is supplied when the triac is off. A minimum off-time of the triac (1.7ms/60Hz and 2ms/50 Hz) is necessary to ensure its supply. The RCD circuit is the same as the one used for the motor drive board.

Software

The light dimmer software is practically the same as with the motor drive. The major difference concerns the mains disturbances rejection in order to prevent lamp flickering. The timing is carried out internally by the MCU timer. The period of operation can be modified to follow the variations of the mains frequency but not the spurious disturbances. The mains synchronisation signal is received every cycle. The corresponding mains period is measured and compared to the internal timer period. If a difference remains after many cycles, the timer period is modified to follow the mains. This block acts like a low band filter which eliminates external filtering components.

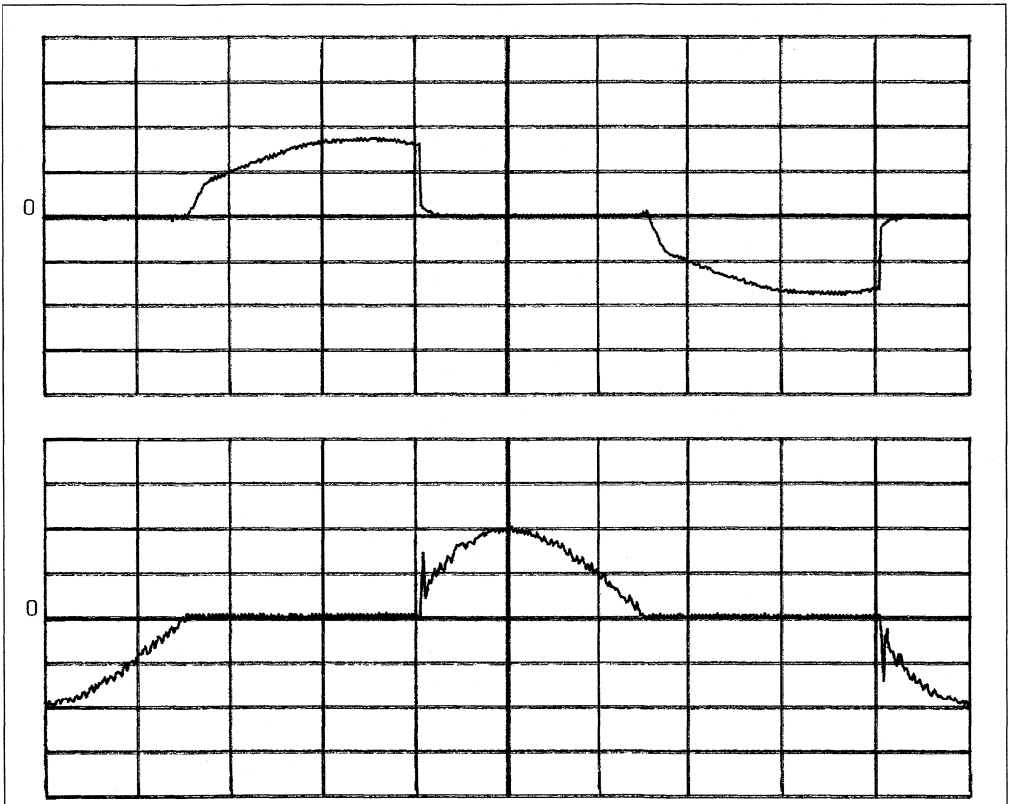
The user interface can be modified to fit other applications such as IR presence detection or alarm, remote control, etc .

PRACTICAL RESULTS

Figure 7 presents the soft start operation with a halogen lamp operating from the secondary of a low voltage transformer and with a high voltage tungsten filament lamp.

With soft start, the peak in-rush current is about 3 times the nominal current compared with 10 to 15 times otherwise. Therefore, the lamp life time is maximized, blowing the input fuse is prevented and the size of the triac is minimized. The figure 8 presents the current and voltage in a triac driving a universal motor.

Figure 8. Universal motor drive: Current and Voltage in the Triac



TRIAC ANODE CURRENT : 2A/div 2ms/div
TRIAC VOLTAGE VAK : 200 V/div 2ms/DIV

SUMMARY

Microcontrollers (MCU) are in common use in most areas of electronics. They are now set to penetrate the very cost sensitive area of home appliances. The applications described in this paper show that enhanced appliance circuits can be designed with fast prototyping using a ST6210 MCU and a SNUBBERLESS or LOGIC LEVEL triac. These circuits are low cost and they can provide more features with less components than classical solutions.

The presented circuits are a universal motor drive, a AC switch and a light dimmer operating from the 110/240V mains. The light dimmer drives incandescent and halogen lamps supplied either directly from the mains or through a low voltage transformer. The motor drive can be adapted, for instance, to vacuum cleaners, food processors, drills or washing machines. Those circuits include soft start and protection features. Different user interfaces can be chosen: touch sensor, push button or potentiometer.

Such features are obtained with only few components: a ST6210 MCU in 20pin DIL/SMD package with a LOGIC LEVEL or SNUBBERLESS triac in TO220 package and some passive components.

Additional features like motor speed regulation, torque limitation, vacuum or unbalance control, IR presence detection, remote control, alarm, homebus interface or electronic shortcircuit protection with IGBT/Mosfet can be implemented from these circuits.

Bibliography

Thyristors and triacs application manual 1986 / SGS-THOMSON Microelectronics
Universal Motor Speed Control / P.Rault + Y.Bahout

Thyristors and triacs application manual 1989 / SGS-THOMSON Microelectronics
Microcontroller based universal motor speed control / M.Queyrol

ST62 user manual 1991 / SGS-THOMSON Microelectronics

Power control with ST6210MCU and triac / Ph.Rabier + L.Perier

SGS-THOMSON MICROELECTRONICS

APPLICATION NOTE

USING ST6 ANALOG INPUTS FOR MULTIPLE KEY DECODING

J.Stockinger

INTRODUCTION

The ST6 on-chip Analog to Digital Converter (ADC) is a useful peripheral integrated into the silicon of the ST6 family members. The flexibility of the I/O port structure allows the multiplexing of up to 13/8 Analog Inputs into the converter in a 28/20 pin device for the ST6210/15 2k ROM and ST6220/25 4k ROM families, enabling full freedom in circuit layout. Many other members of the ST6 family also offer the Analog to Digital converter.

One of the more novel and practical applications of this converter, is to decode a number of keys. The technique is to connect the keys by resistive voltage dividers to the converter inputs. An example of key detection using 10 keys is illustrated in this note.

Using the Analog to Digital converter in this fashion does not require a static current and avoids false key detection.

BASIC CIRCUIT

The basic circuit of the key decoder consists of a pull-up resistor connected to the ST6 Analog to Digital converter input with the first key directly switching to ground. The following keys are then connected in sequence to the ADC input through serial resistors. The number of keys which may be detected depends on the tolerance of the resistors used. It can be seen that if more than one key is pressed at the same time, the key detected will be the next key in the chain closest to the ADC input. This also allows the keys in the keyboard to be prioritized.

PRINCIPLE OF OPERATION

The combination of the pull-up resistor, the serial resistors and the pressed key form a resistive voltage divider, generating a different voltage at the ADC input for each key pressed. The serial resistors are selected in order to give an equal distribution of voltage between V_{DD} and V_{SS} for each switch combination to give the best noise margin between keys.

When a key is pressed, the voltage at the ADC input is given by the activated voltage divider. This analog voltage is converted by the ADC and the digital value is used to determine which switch is closed. Two successive conversions may be made to avoid the influence of key bounce.

Figure 1. Analog Keyboard resistor key matrix

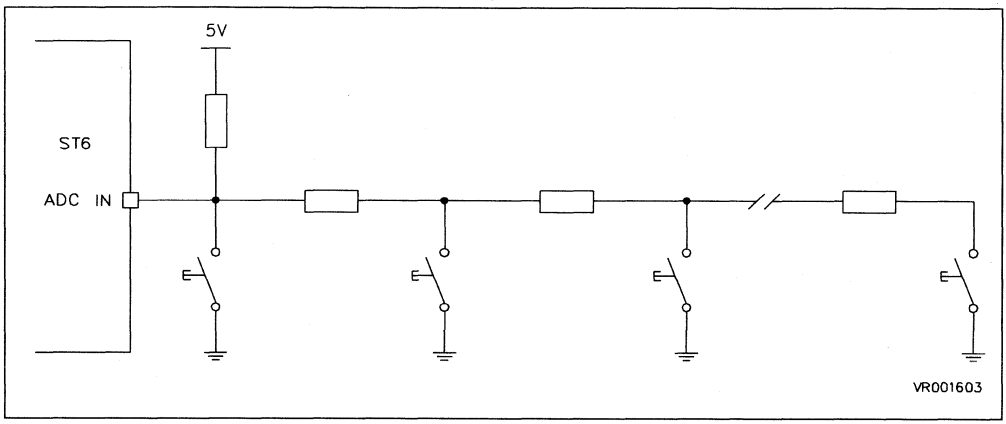


Figure 2. Multiple key press

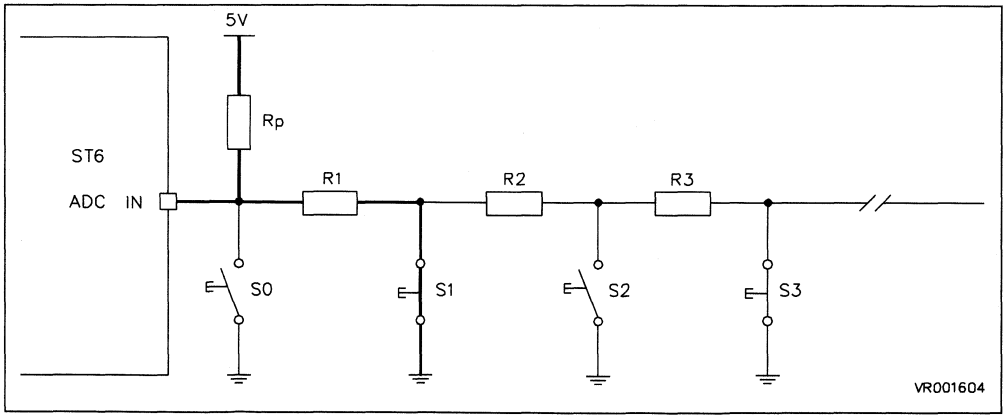


Table 1. Key code ranges

Key Nr	Valid Code Range	Distance to next key
1	0	24
2	18-1A	22
3	30-33	22
4	49-4E	21
5	63-68	20
6	7C-81	22
7	97-9B	21
8	B0-B4	22
9	CA-CD	24
10	E5-E6	25

If the top key is pressed, the voltage measured is always zero. For n keys, the resistor values should be selected such that the voltage for the second key from top is V_{DD}/n, for the 3rd - $2xV_{DD}/n$, for the 4th - $3xV_{DD}/n$ and for the nth - $(n-1)xV_{DD}/n$. Resistor values from the tolerance set used must be selected to meet this requirement.

The recommended resistor values for a 10-key keyboard with 2% resistors from the E24 series, used with a 10kΩ pull-up resistor, are shown in table 2. If more current can be allowed, then a 1kΩ resistor can be used in which case the serial resistor values should be divided by 10.

Table 2. Used resistors and Tolerance

Resistor	Value (Ω)	-2% (Ω)	+2% (Ω)
Rp	10000	9800	10200
R1	1100	1078	1122
R2	1300	1274	1326
R3	1800	1764	1836
R4	2400	2352	2448
R5	3300	3234	3366
R6	5100	4998	5202
R7	8200	8036	8364
R8	16000	15680	16320
R9	51000	49980	52020

PRACTICAL LIMITATIONS

Theoretically, for an ideal power supply, ADC and resistors, 255 keys could be detected. Practically however, it is necessary to take into account potential errors coming from:

- the power supply - the key resistivity - the resistor tolerance - the ADC error

The power supply tolerance can normally be neglected providing noise is not present at a frequency within or above the frequency range of the RC delay of the resistive divider, as the ADC reference is normally provided by the power supply of the ST6. For ST6 family members with external ADC reference voltage inputs, AV_{DD} and AV_{SS} may be used instead of V_{DD} and V_{SS}.

The sensitivity of the key can normally be neglected, as the resistance of the divider is high in comparison to it. If the key resistivity is significant, it should be added to the "serial" pull-down resistance of the different dividers. The key resistivity variation must also be added to the tolerance of the serial pull-down resistor (see resistor tolerance following).

The resistor tolerance affects the tolerance of the dividers. Two situations must be taken into account:

a) minimum value of pull-up combined with maximum values of pull-down = maximum voltage of the divider at the ADC input.

b) maximum value of the pull-up combined with the minimum values of pull-down = minimum voltage at the ADC input. These two cases give the maximum voltage variation of each divider (see Table 3). The voltage variation ranges of two dividers must not overlap otherwise the key cannot be decoded, even with an ideal converter.

Table 3. Effective Divider Resistors

Active Key	R -2% (Ω)	R +2% (Ω)
S0	0	0
S1	1078	1122
S2	2352	2448
S3	4116	4284
S4	6468	6732
S5	9702	10098
S6	14700	15300
S7	22736	23664
S8	38416	39984
S9	88396	92004

Realistic converters require a margin between the range of variation. In the case of a significant variation in the key resistivity, the maximum resistivity of the key has to be added to the value of the pull-down resistor in case a). For case b) no error needs to be added as the resistivity cannot be less than 0 Ω.

The linearity of the ADC converter of the ST6 is normally specified for ±2 LSB, therefore a minimum distance of 4 LSB is needed between the edges of the resistance tolerance ranges. For the best results, a minimum of 8 LSB should be used (see Table 4).

Table 4. Voltage at the ADC-Input,Converter Results (5V supply)

Active Key	V (Rxmin-Rpmax)			V (Rxmax-Rpmin)		
	V	hex.	dec.	V	hex.	dec.
S0	0.00	00	0	0.00	00	0
S1	0.48	18	24	0.51	1A	26
S2	0.94	30	48	1.00	33	51
S3	1.44	49	73	1.52	4E	78
S4	1.94	63	99	2.04	68	104
S5	2.44	7C	124	2.54	81	129
S6	2.95	97	151	3.05	9B	155
S7	3.45	B0	176	3.54	B4	180
S8	3.95	C9	201	4.02	CD	205
S9	4.48	E5	229	4.52	E6	230

Table 5. AD-Converter Results

Active Key	R Error Range (LSB)	Distance to next Key	Valid Key Range
S0	0	24	0-0
S1	2	22	18-1A
S2	3	22	30-33
S3	4	21	49-4E
S4	5	20	63-68
S5	5	22	7C-81
S6	5	21	97-9B
S7	4	22	B0-B4
S8	3	24	C9-CD
S9	2	25	E5-E6

EXTENSION FOR WAKE UP

ST6 family members with the Analog input capacity can also generate a wake-up operation (from WAIT or STOP modes) on the pressing of a key. This can be achieved by a modification of the circuit shown in figure 1. The pull-up resistor is not connected to V_{DD} but to an additional I/O port bit. During key polling, this additional port bit is set to output mode active high, thus effectively switching V_{DD} to the pull-up resistor. The resistance of the pull-up resistor must be high enough to give no significant voltage drop, or the resulting error must be calculated and taken into account. The other I/O bit is used as the Analog input to the ADC as in the original circuit.

During the wait for the key press, the first I/O pin, used to pull the pull-up resistor high to V_{DD} while polling, is switched into a high impedance state (e.g. open drain output mode). The second I/O pin, used as the ADC input while polling, is switched to the interrupt input with pull-up mode. The internal pull-up is in the range of 100k, in comparison to the 1k - 10k of the external resistor used during polling. If any key is now pressed an interrupt will be generated if the voltage at the second I/O pin is below the Schmitt trigger low level threshold. The serial resistors in the keyboard chain must not be too high in this case, therefore the maximum number of keys is reduced in comparison to the normal mode.

Figure 3. Keyboard wake-up circuit

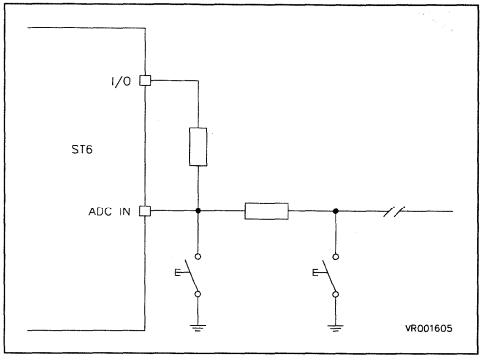


Figure 4. Keyboard reading

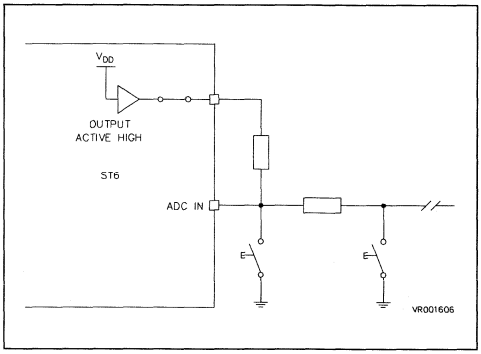
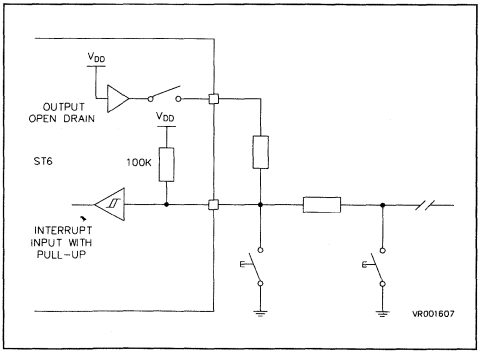


Figure 5. Interrupt configuration



APPENDIX A: Key Input by Polling

```
;*************************************************************************
;*                                                                       *
;*                        SGS-THOMSON GRAFING                            *
;*                                                                       *
;*                   APPLICATION NOTE 431  -  ST6                        *
;*                                                                       *
;*            Use of ADC inputs for multiple key decoding                *
;*                                                                       *
;*          With the inbuilt A/D converter of any ST6 it is easy to      *
;*          implement a small routine which enables ONE port pin, con-   *
;*          figured as an ADC input, to decode up to ten different switches*
;*          All that is necessary is to set one port pin as an ADC input *
;*          Then the program runs in an endless loop until one of the    *
;*          connected keys is pushed.                                    *
;*          The value from the ADC data register is then used to decide  *
;*          how the program will continue,on reaction to the key-push.   *
;*                                                                       *
;*************************************************************************

                        ;***REGISTERS***

ddrpb       .def    0c5h   ;port B data direction register
orpb        .def    0cdh   ;port B option register
drpb        .def    0c1h   ;port B data register
adr         .def    0d0h   ;A/D data register
adcr        .def    0d1h   ;A/D control register
a           .def    0ffh   ;accumulator

                        ;***CONSTANTS***

inpall      .equ    000h   ;used for setting all pins input
peg1_2      .equ    00ch   ;border to distinguish between switch1 and switch2
peg2_3      .equ    025h   ;border to distinguish between switch2 and switch3
peg3_4      .equ    03eh   ;border to distinguish between switch3 and switch4
peg4_5      .equ    058h   ;border to distinguish between switch4 and switch5
peg5_6      .equ    072h   ;border to distinguish between switch5 and switch6
peg6_7      .equ    08ch   ;border to distinguish between switch6 and switch7
peg7_8      .equ    0a5h   ;border to distinguish between switch7 and switch8
peg8_9      .equ    0beh   ;border to distinguish between switch8 and switch9
peg9_10     .equ    0d9h   ;border to distinguish between switch9 and switch10
```

```
        ldi     ddrpb,inpall  ;sets all port B pins low -- all input
        ldi     orpb,01h      ;option register:
                              ;sets bit b0 high, the rest low
        ldi     drpb,01h      ;direction register:
                              ;sets bit b0 high, the rest low
                              ;-- pb0 becomes analog input
                              ;   pb1-7 become input with pull-up, but
                              ;   are not used here (only one pin may be
                              ;   analog input for A/D at the same time)
        ldi     adcr,30h      ;A/D control register:
                              ; 0011 0000 -- -activate A/D converter
                              ;              -start conversion
                              ;              -disable A/D interrupt
loop:   jrr     6,adcr,loop   ;loop until the End Of Conversion bit is
                              ;set (indicator that a conversion has
                              ;been completed)
        ld      a,adr ;load acc with the result of the A/D
                              ;conversion
                              ;now the result is compared with the
                              ;switches
sw1:    cpi     a,peg1_2      ;compare with peg1_2
        jrnz    sw2           ;A/D result was smaller than peg1_2
        jp      s1            ; -- switch1 was pressed: jump to s1

sw2:    cpi     a,peg2_3      ;compare with peg2_3
        jrnz    sw3           ;A/D result was smaller than peg2_3
        jp      s2            ; -- switch2 was pressed: jump to s2

sw3:    cpi     a,peg3_4      ;compare with peg3_4
        jrnz    sw4           ;A/D result was smaller than peg3_4
        jp      s3            ; -- switch3 was pressed: jump to s3

sw4:    cpi     a,peg4_5      ;compare with peg4_5
        jrnz    sw5           ;A/D result was smaller than peg4_5
        jp      s4            ; -- switch4 was pressed: jump to s4

sw5:    cpi     a,peg5_6      ;compare with peg5_6
        jrnz    sw6           ;A/D result was smaller than peg5_6
        jp      s5            ; -- switch5 was pressed: jump to s5
```

```
sw6:        cpi     a,peg6_7        ;compare with peg6_7
            jrnz    sw7             ;A/D result was smaller than peg6_7
            jp      s6              ; -- switch6 was pressed: jump to s6

sw7:        cpi     a,peg7_8        ;compare with peg7_8
            jrnz    sw8             ;A/D result was smaller than peg7_8
            jp      s7              ; -- switch7 was pressed: jump to s7

sw8:        cpi     a,peg8_9        ;compare with peg8_9
            jrnz    sw9             ;A/D result was smaller than peg8_9
            jp      s8              ; -- switch8 was pressed: jump to s8

sw9:        cpi     a,peg9_10       ;compare with peg9_10
            jrnz    sw10            ;A/D result was smaller than peg9_10
            jp      s9              ; --> switch9 was pressed: jump to s9

sw10:       jp      s10             ;A/D result was greater than peg9_10
                                    ; -- switch10 was pressed: 0
                                    ;

;*** the routines handling to the reaction to the individual key presses
;*** are to be included here.

s1:
s2:
s3:
s4:
s5:
s6:
s7:
s8:
s9:
s10:
```

APPENDIX B: Key Input by Interrupt

```
;*************************************************************************
;*                                                                       *
;*                        SGS-THOMSON GRAFING                            *
;*                                                                       *
;*                   APPLICATION NOTE 431 -    ST6                       *
;*                                                                       *
;*             Use of ADC inputs for multiple key decoding               *
;*                                                                       *
;*           With the inbuilt A/D converter of any ST6 it is easy to     *
;*           implement a small routine with which you can recognize      *
;*           if one of nine connected keys is pushed by creating an      *
;*           interrupt. The program can then decide how it will react    *
;*           to the key pushed.                                          *
;*                                                                       *
;*                                                                       *
;*************************************************************************

                        ;***REGISTERS***

 ddrpb      .def    0c5h  ;port B data direction register
 orpb       .def    0cdh  ;port B option register
 drpb       .def    0c1h  ;port B data register
 ior        .def    0c8h  ;interrupt option register
 adr        .def    0d0h  ;A/D data register
 adcr       .def    0d1h  ;A/D control register
 a          .def    0ffh  ;accumulator

                        ;***CONSTANTS***

inpall      .equ    000h  ;used for setting all pins input
peg1_2      .equ    00ch  ;border to distinguish between switch1 and switch2
peg2_3      .equ    025h  ;border to distinguish between switch2 and switch3
peg3_4      .equ    03eh  ;border to distinguish between switch3 and switch4
peg4_5      .equ    058h  ;border to distinguish between switch4 and switch5
peg5_6      .equ    072h  ;border to distinguish between switch5 and switch6
peg6_7      .equ    08ch  ;border to distinguish between switch6 and switch7
peg7_8      .equ    0a5h  ;border to distinguish between switch7 and switch8
peg8_9      .equ    0beh  ;border to distinguish between switch8 and switch9

; en_kint (enable key-interrupt) sets the registers in a way that pushing
; any key will cause an interrupt. This subroutine must be called to
; re-enable the key interrupt (e.g. after handling the key service routine)
```

```
en_kint:
          ldi     ddrpb,inpall   ;sets all port B pins low -- all input
          ldi     orpb,02h       ;option register:
                                 ; sets bit b1 high, the rest low
          ldi     drpb,01h       ;data register:
                                 ; sets bit b0 high, the rest low
                                 ;-- pb0 becomes input, no pull-up, no int
                                 ;   pb1 becomes input with pull-up and int.
                                 ;   pb2-7 become input with pull-up, but
                                 ;   are not used here
          ldi     ior,10h        ;interrupt option register:
                                 ;-- set D4: enable all interrupts
                                 ;   reset D5: falling edge on int.input(#2)
          ret                    ;return to the calling address

;*** hd_kint (handle key interrupt) interrupt service routine
;*** evaluates the data resulting in pushing a key.
;*** Interrupt vector #2 (0ff4h and 0ff5h) must point (jump) to hd_kint.

hd_kint:  ldi     drpb,03h       ;data register:
                                 ; 0000 0011
          ldi     ddrpb,01h      ;data direction register:
                                 ; 0000 0001
                                 ; -- pb0 becomes output
          ldi     orpb,03h       ;option register:
                                 ; 0000 0011
                                 ; -- pb0: push-pull output
                                 ; -- pb1: ADC-input
                                 ;    pb2-7 become input with pull-up, but
                                 ;    are not used here
          ldi     adcr,30h       ;A/D control register:
                                 ; 0011 0000 -- -activate A/D converter
                                 ;              -start conversion
                                 ;              -disable A/D interrupt
loop:     jrr     6,adcr,loop    ;waits until the End Of Conversion
                                 ; bit is set (indicator that a conversion
                                 ; has been completed)
          ld      a,adr          ;load acc with the result of the A/D
                                 ; conversion
                                 ;now the result is compared with the
                                 ; values which represent the different
                                 ; switches
```

```
sw1:        cpi     a,peg1_2        ;compare with peg1_2
            jrnz    sw2             ;A/D result was smaller than peg1_2
            jp      s1              ; -- switch1 was pressed: jump to s1

sw2:        cpi     a,peg2_3        ;compare with peg2_3
            jrnz    sw3             ;A/D result was smaller than peg2_3
            jp      s2              ; -- switch2 was pressed: jump to s2

sw3:        cpi     a,peg3_4        ;compare with peg3_4
            jrnz    sw4             ;A/D result was smaller than peg3_4
            jp      s3              ; -- switch3 was pressed: jump to s3

sw4:        cpi     a,peg4_5        ;compare with peg4_5
            jrnz    sw5             ;A/D result was smaller than peg4_5
            jp      s4              ; -- switch4 was pressed: jump to s4

sw5:        cpi     a,peg5_6        ;compare with peg5_6
            jrnz    sw6             ;A/D result was smaller than peg5_6
            jp      s5              ; -- switch5 was pressed: jump to s5

sw6:        cpi     a,peg6_7        ;compare with peg6_7
            jrnz    sw7             ;A/D result was smaller than peg6_7
            jp      s6              ; -- switch6 was pressed: jump to s6

sw7:        cpi     a,peg7_8        ;compare with peg7_8
            jrnz    sw8             ;A/D result was smaller than peg7_8
            jp      s7              ; -- switch7 was pressed: jump to s7

sw8:        cpi     a,peg8_9        ;compare with peg8_9
            jrnz    sw9             ;A/D result was smaller than peg8_9
            jp      s8              ; -- switch8 was pressed: jump to s8

sw9:        jp      s9              ;A/D result was bigger than peg8_9
                                    ; -- switch9 was pressed: jump to s9
                                    ;

;*** The routines handling the reaction to the individual key presses
;*** are to be included here
```

```
s1:
s2:
s3:
s4:
s5:
s6:
s7:
s8:
s9:

;*** Each routine must end with the following lines in order to enable
;*** another interrupt when the next key is pressed.

            call en_kint  ; enable another interrupt
return:     reti
```

SGS-THOMSON MICROELECTRONICS

APPLICATION NOTE

DESIGNING WITH MICROCONTROLLERS IN NOISY ENVIRONMENTS

INTRODUCTION

Microcontrollers (MCU) make possible the design of integrated and flexible controls for a constantly decreasing cost. As a result, they are spreading rapidly among most electronic applications and especially noise sensitive equipments such as for power control or automotive use.

An MCU operates with sequential logic, so the control of an application can be lost during a disturbance, as with analog control, but also after a power glitch in the system.

In addition, a modern MCU includes several tens of thousands of transistors switching in the MHz range, potentially radiating interference of high magnitude in a large frequency spectrum. Consequently, noise sensitivity and generation have to be considered as early as possible in MCU based designs.

This Application note presents numerous methods to effectively reduce noise problems. The first part presents a short overview on noise and proposes hardware solutions to increase the equipment immunity to noise. The second part concerns the writing of software more immune to disturbances. The behaviour versus disturbances of an MCU designed for noisy environments, the ST6210, is presented and practical examples and results are shown.

NOISE PREVENTION

The major noise receptors and generators are the tracks and wiring on the Printed Circuit Board (PCB), especially those near the MCU. The first actions to prevent noise problems thus concern the PCB layout and the design of the power supply.

Optimized PCB layout

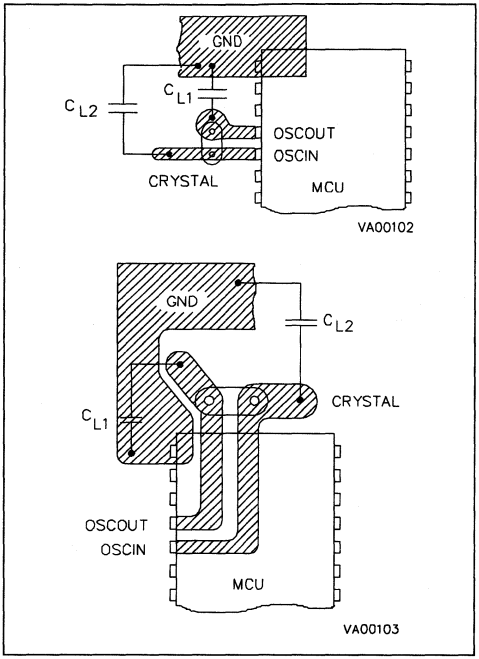
Noise is basically received and transmitted through tracks and components which, once excited, act as antennas. Each loop and track includes parasitic inductances and capacitances which radiate and absorb energy once submitted to a variation of current, voltage or electromagnetic flux.

An MCU chip itself presents high immunity to and low generation of EMI since its dimensions are small versus the wave lengths of EMI signals (typically mm versus 10's of cm for EMI signals in the GHz range). So a single chip solution with small loops and short wires reduces noise problems.

The initial action at the PCB level is to reduce the number of possible antennas. The loops and wires connected to the MCU such as supply, oscillator and I/O should be considered with a special attention (Figure 1). The oscillator loop has to be especially small since it operates at high frequency.

A reduction of both the inductance and the capacitance of a track is generally difficult. Practical experience suggests that in most cases the inductance is the first parameter to be minimized.

Figure 1. PCB Board Oscillator Layout Examples



The reduction of inductance can be obtained by making the lengths and surfaces of the track smaller. This can be obtained by placing the track loops closer on the same PCB layer or on top of one another (Figure 2). The resulting loop area is small and the electromagnetic fields reduce one another.

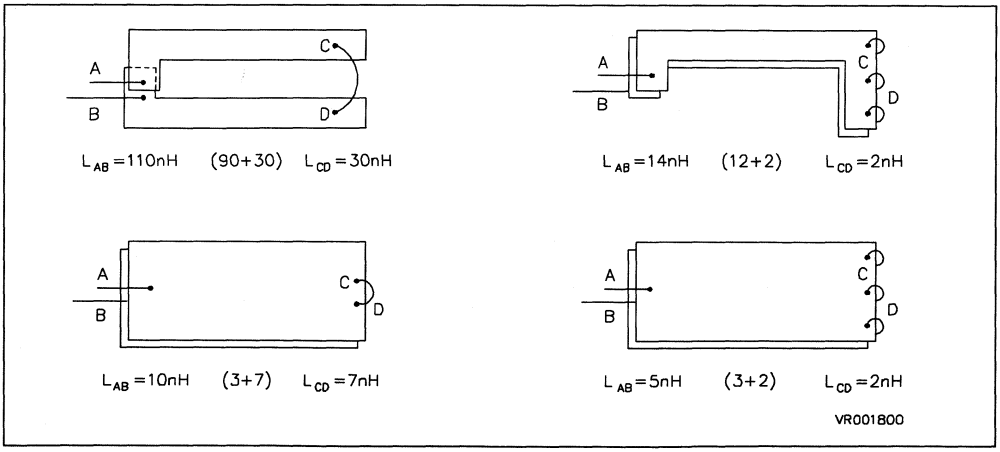
The ratio in order of magnitude relating to the inductance value and the area defined by the wire loop is around 10nH/cm^2. Typical examples of low inductivity wires are coaxial, twisted pair cables or multiple layer PCBs with one ground and one supply layers. The current density in the track can also be smaller due to track enlargement or the paralleling of several small capacitances mounted in the current flow.

In critical cases, the distance between the MCU and the PCB, and therefore the surfaces of the loops between an MCU and its environment, has also to be minimized. This can be achieved by removing any socket between the MCU package and the PCB, by replacing a ceramic MCU package by a plastic one or by using Surface Mounting instead of Dual In Line packages.

Power supply filtering

The power supply is used by all parts of the circuit, so it has to be considered with special attention. The supply loops have to be decoupled to make sure that signal levels and power currents do not interfere. These loops can be separated using star wiring with one node designated as common for the circuit (Figure 3).

Figure 2. Reduction of PCB Tracks Loop Surfaces



Note: This test is done with a double sided PCB. Insulator thickness is 1.5mm, copper thickness is 0.13mm. The overall board size is 65 x 200mm.

The decoupling capacitance should be placed very close to the MCU supply pins to minimize the resultant loop. It should be also large enough to absorb, without significant voltage increase, parasitic currents coming from the MCU via the input protection diodes. The decoupling of the board can be done with electrolytic capacitors (typ. 10µF to 100µF) since the dielectric used in such capacitors provides a high volumic capacitance. However these capacitors behave like inductances at high frequency (typ. above 10MHz) while ceramic or plastic capacitances keep a capacitive behaviour at higher frequency. A ceramic capacitance of, for instance, 0.1µF to 1µF should be used as high frequency supply decoupling for critical chips operating at high frequency.

The supply circuit must be sized in such way that its components can absorb energy peaks during supply overvoltages. For instance, in a power supply done with a capacitor in series between the mains supply and the MCU supply (typically +5V), this capacitance is a short circuit when a voltage spike occurs. The corresponding short circuit current has to be absorbed by a protection zener diode. Depending on the maximum energy to withstand, a standard 0.5W Zener diode (e.g. BZX55C) may have to be replaced by a 1.3W (BZX85C) or 2W (BZV47C) Zener diode (Figure 15). Additional filtering with serial resistance or inductance can be included to reduce the influence of voltage spikes and to absorb transients coming from the input supply line.

I/O configuration

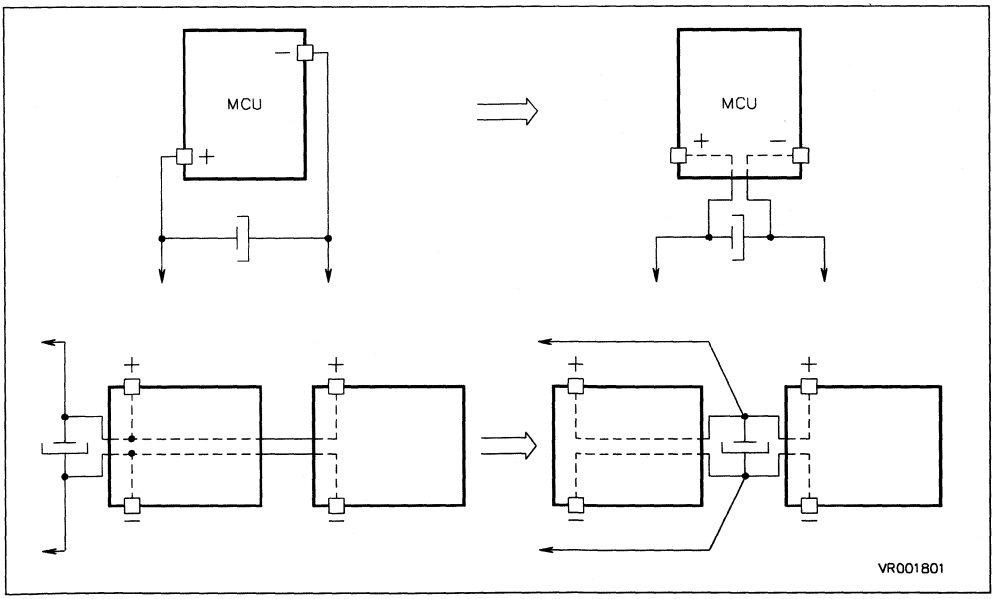
In general, the smaller the number of components surrounding the MCU, the better the immunity versus noise. A ROMless solution, for instance, is typically more sensitive to and a bigger generator of noise than an embedded circuit.

If the output buffers are embedded in the MCU, their switching speed has to be controlled in order to avoid parasitic oscillations when they are switching. A trade-off between noise and speed has to be found by the MCU designers.

I/O pins which are not used in the application should be preferably grounded or connected via a large impedance (i.e. 100kΩ) to a fixed potential, depending on the MCU reset configuration. Here, the trade-off is between immunity and consumption.

If a current can be forced in an input pin, clamping protection with diodes has to be included in the circuit connected to the pin to divert the current

Figure 3. Supply Lay-out Examples



from the MCU structure and to avoid the risk of latch-up (Figure 4). In an MCU such as the ST62, these diodes are integrated inside the chip.

Shielding

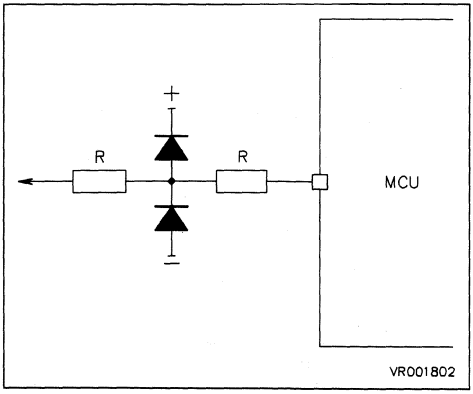
Shielding can help in reducing noise reception and emission, but its success depends directly on the material chosen as shield (high permeability, low resistivity) and on its connection to a stable voltage source including a decoupling capacitance via a low serial impedance (low inductance, low resistance).

If the generator of major disturbances is near to the MCU board and can be identified as a strong dV/dt generator (i.e. a transformer or Klystron), the noise is carried mainly by the electrostatic field. The critical coupling between the noise generator and the control board is capacitive. A highly conductive shield (i.e. copper) creating a Faraday cage around the control board may strongly increase the immunity.

If the strongest source of perturbations is a dI/dt generator (i.e. a relay), it is a high source of electromagnetic fields. Therefore, the permeability of the shielding material (i.e. alloy) is crucial to increase the immunity of the board. In addition, the number and size of the holes on the shield should be reduced as much as possible to increase its efficiency.

In critical cases, the implantation of a ground plane below the MCU and the removal of sockets between the device and the PCB can reduce the MCU noise sensitivity. Indeed, both actions lead to a reduction of the apparent surface and loop between the MCU, its supply, its I/O and the PCB.

Figure 4. Standard MCU I/O Block Protection



WRITING SAFER SOFTWARE

The hardware solutions described help in reducing the noise received and radiated by the MCU. Nevertheless if the MCU is disturbed, a modification of a register, for instance the program counter, can occur. In this case, control of the application should not be lost.

Writing safe software can prevent most of the problems due to parasitic modifications of registers or program counter. Many register errors can be quickly identified and masked in the program flow without influence on the environment.

The examples and indications given in this section are written for the ST62 MCU family.

The basic precepts for the writing of a safe software are:

-Test only configurations clearly defined in the flow chart

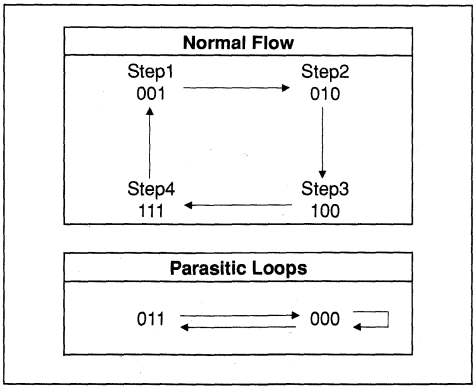
-Regularly check vital data stored in RAM

-Control the program flow

-Fill the unused memory

It is useful for the program to identify to itself that it is following the correct program flow. This can be implemented by using trace points held in several bits of a specific register (Figure 5). If more bit flags are included in the decision than strictly necessary, the program may enter in a mode which it never leaves. Spikes may set an unused bit, so an error can be generated and an exit should be defined for every trace point condition.

Figure 5. Trace Point of a Program Flow



The trace points can be used also to control the flow of the tasks. In such a case, a "called" task can be only called by a "calling" task. Such checks can be done at the beginning and at the end of each task.

The program flow can be also monitored by controlling the duration of a subroutine, for instance, by reading a timer value at the beginning and at the end of the task.

The contents of the data RAM may be changed by noise, therefore it is good practice to check regularly the consistency of vital data. For example, the fact that the RAM value is inside a predefined interval, its coherence with a value previously stored, a checksum or a comparison with a copy (inverted or not) can be checked. Constant values can be stored in non-used RAM addresses and regularly checked to make sure that the RAM data are not disturbed. The control registers of the MCU peripherals (used or not) can also be reinitialized regularly inside the main routine.

Unused parts of the program memory can be filled with the NOP instruction plus a reset of the chip at the end of the unused area (LDI WDT,01h). If the program counter is modified after a glitch and sends the program to the unused area, the program will not hang in an endless loop and the core is finally reset.

Example:

unused area

```
04h             ; NOP
04h             ; NOP
....
04h             ; NOP
LDI WDT,01h     ; generate reset
                ; of chip
```

end of unused area

The unused part of the program memory can also be filled with the ST62 JP X9X instruction since this instruction has its two hexadecimal bytes identical (hex instruction code X9X9). If an unforseeen disturbance to the program counter sends the program to the unused area, the program immediately jumps to address X9X. This address can be at the beginning of the program (090 for 4kROM, 898 for 2kROM versions) or a reset instruction (LDI WDT,01h).

The Watchdog should be refreshed a minimum of times and in the main loop. Its refresh value can be calculated in order to minimize the reload value and therefore the duration without a potential watchdog reset. In addition, the user has to make sure that the main routine is executed from time to time. The Watchdog should never be used for tests in the main routines, especially not as a timer.

Additional flags and trace points in the subroutines can be used to check that the program path is correct before reloading the watchdog. Instead of refreshing the watchdog with a constant value using a LDI instruction (i.e. LDI WDT,#0FFh), the refresh value can be preselected or calculated depending on the trace point TP, using the accumulator A (i.e. LD A,TP and LD WDT,A).

The program presented in annex 1 has been written for the ST621x/2x. It checks the flags, the trace points and adjusts the watchdog refresh value. It is written in such way that the watchdog is reloaded only in the main loop and not in a subroutine or an interrupt routine. If the watchdog has to be reloaded out of the main loop, the application safety is reduced and this example has to be modified. It can be implemented in applications which can start again after a reset and where the reset configuration of the MCU I/O pins may occur without damage in any step of operation of the equipment.

The ROM content has also to be checked in order to avoid data combinations where the watchdog register may be written unintentionally. This can occur if a byte follows another byte which, read as an instruction, can modify the watchdog, and if the program counter is corrupted. For instance in the ST62, the watchdog address byte (D8) is the same as the JRNZ instruction.

Example:

Initial version:

```
CPI A,#0D     =370D
JRNZ OUTloop =D8nn  (0DD8 sequence
                     in program)
```

Modified version:

```
CPI A,#0D     =370D
NOP           =04
JRNZ OUTloop =D8nn
```

The following table lists the critical bytes not to be placed before this byte.

Two Successive Bytes	Equivalent Instructions
0D D8	LDI WDT
2B D8	RES 4,WDT
3B D8	SET 4,WDT
4B D8	RES 2,WDT
5B D8	SET 2,WDT
6B D8	RES 6,WDT
7B D8	SET 6,WDT
9B D8	SET 1,WDT
AB D8	RES 5,WDT
BB D8	SET 5,WDT
CB D8	RES 3,WDT
DB D8	SET 3,WDT
EB D8	RES 7,WDT
FB D8	SET 7,WDT
7F D8	INC WDT
9F D8	LD WDT,A

If the program flow is such that the watchdog register byte address follows one of the critical bytes listed, the watchdog contents can be corrupted. The solution to this problem can be either to modify the first byte i.e. by changing the data RAM location (if used) or to insert a NOP instruction between the 2 critical bytes.

In addition, if possible, the data in the X or Y index registers should never be identical to the WDT address.

Operation may also be disturbed due to noise on input lines. All inputs can be digitally filtered, so that an input (analog or digital) is valid only if it remains constant for a defined time. This reduces the number of passive components.

Example:

```
Main1 LDI loop,04h
Main2 JRR 4,PB,Main1; continue flow
                    ; if PB4=0
      DEC loop      ; for 4 successive
                    ; measurements
      JRNZ Main2
Main3 LDI loop,04h
Main4 JRS 4,PB,Main3; continue flow
                    ; if PB=1
      DEC loop      ; for 4 successive
                    ; measurements
      JRNZ Main4
```

ST62, AN MCU FAMILY DESIGNED FOR NOISE IMMUNITY

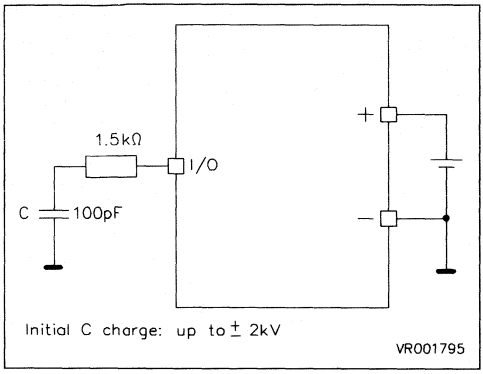
This section presents some technology and design solutions used in the ST62 MCU family to enable safe operation when used in products in disturbed or noise sensitive environments (Figure 6).

High destruction limits

Destruction of an MCU is usually due to Electrostatic discharge (ESD), a peak voltage or latchup which causes uncontrolled current to flow in the chip and to concentrate in some parts of the structure where a high voltage is applied. The common action of the current and voltage is the creation of hot spots which burn the silicon of the device.

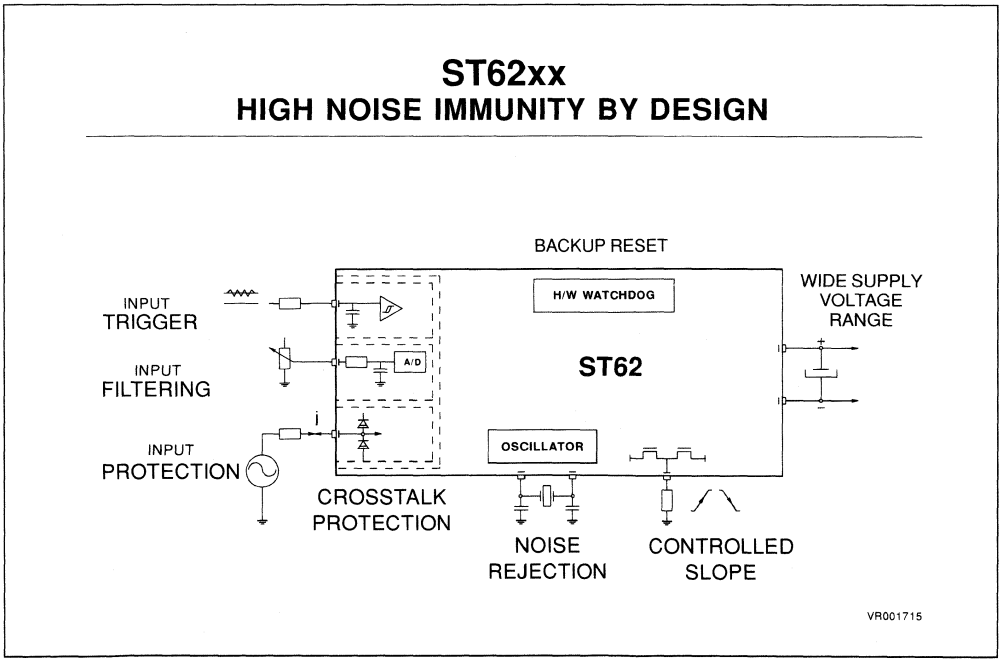
Such defects mechanisms are modelled and corresponding tests are applied on the chips. The ESD test simulates the action of electrostatic energy stored in the parasitic capacitance of a person, which is discharged in the chip. It is modelled by standards such as MIL STD 883.5 (Figure 7).

Figure 7. ESD Test Schematic



The latch-up test determines ruggedness of the device to overvoltage and current injection. An SGS-THOMSON corporate quality specification defines the test procedure. In the first test, an overvoltage higher than the maximum specified rating is applied on the supply pins. For the second test, a high current pulse is injected in I/O pins of a device supplied normally. In both tests, latch-up is

Figure 6. Major ST62 Features Increasing its Noise Immunity



observed by measuring the supply (and I/O pin) current and by making sure that no discontinuity occurs in the current growth (Figure 8).

When the MCU is used inside its specified characteristics with normal handling precautions, such defects mechanism should not occur.

High noise immunity

Several optimization technics have been implemented in the technology and design of the ST62 to minimize its sensitivity to external disturbances.

Voltage potential wells have been integrated between the I/O cells and between other I/O and logic cells, avoiding noisy line influences on other MCU blocks. Protection diodes are included inside each I/O pin, the timer and the NMI cells. If the current in these diodes is limited with external resistances, the diodes can be used functionally, providing that the total current in the supply is also limited. Typical values of the diode current for the ST6210 are 2.5mA per I/O, 0.5mA for NMI/timer and 25mA total.

Schmitt triggers are included in each input to filter noisy signals. The hysteresis levels of comparison on the digital inputs are typically 3.5V for level "1" and 1.5V for level "0" with a +5V supply.

Capacitances are included in the pads (typ. 5pF) to provide a minimum of filtering if an external resistance is connected. These capacitances are internally associated to resistances to avoid capacitive coupling. The A/D converter also includes its own filter to help stabilizing the input signal during the conversion (Figure 9).

The wide supply voltage range between 3V and 6V allows the ST6210 to operate safely inside these limits even if the voltage is not stable, providing that the oscillator frequency is compatible with the voltage (Figure 10).

The V_{DD}, V_{SS} and oscillator pins of the ST6210 are close to each other. In this way, the surface of the most critical loops is minimized.

Figure 8. Latch-up Test Schematics

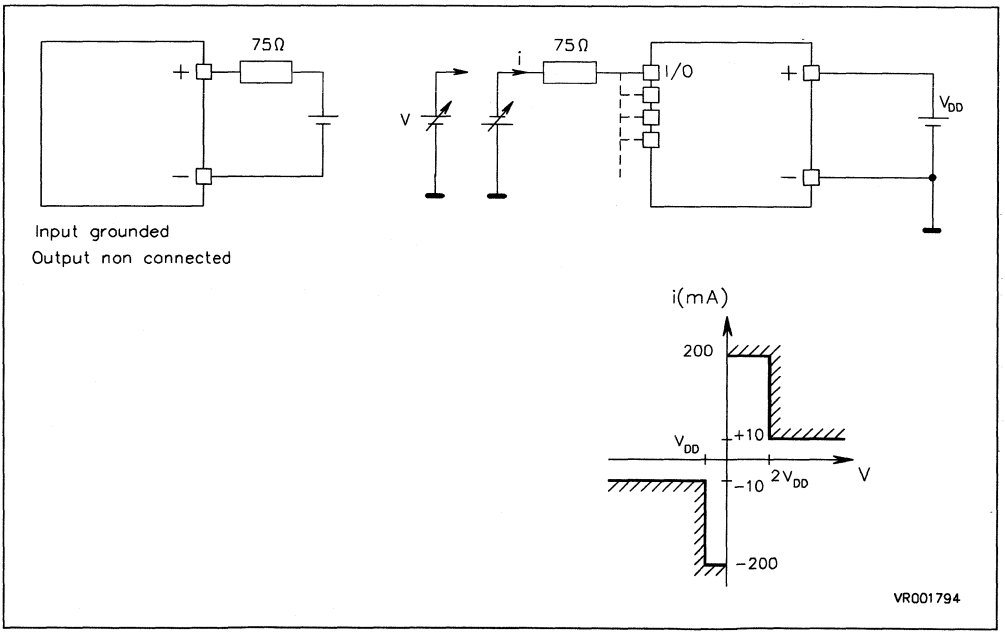


Figure 9. A/D Converter Input Schematic on ST6210

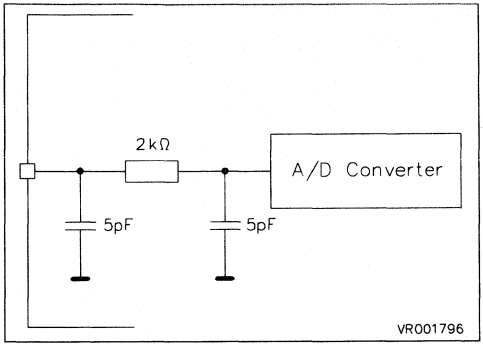


Figure 10. Relation Between Oscillator Frequency and Supply Voltage on ST6210

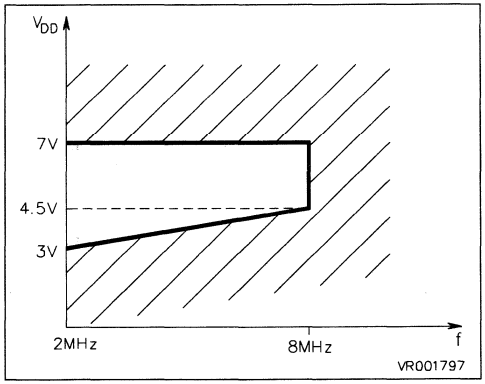
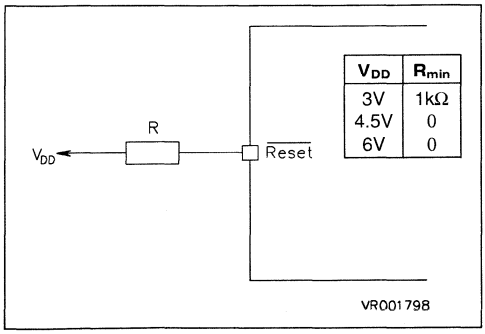


Figure 11. Reset Pin Connection of the ST6210



V_{DD}	R_{min}
3V	1kΩ
4.5V	0
6V	0

Low noise generation

High current buffers (20mA typ.) are included in the outputs of the ST62 MCU. The output edges are slowed down in order to avoid over-oscillation at the commutations (typ. 10ns switching time). This is especially useful when inductive loads are driven with the 20mA ports, in parallel or not.

The internal databus of the ST62 CPU is serial, meaning that only few transistors switch at the same time (typically 1/13th versus an MCU with a parallel bus). The radiated spectrum and the noise on the supply are reduced compared to a parallel architecture running at the same oscillator frequency.

Reset modes

A strong glitch or a power line failure may stop or strongly disturb the operation of the program. In such case, the MCU has also to recover safely. This is achieved via a hardware reset. With ST621X such a reset can come from an external pin, the watchdog or the power-on-reset (POR) block.

The reset pin allows the reset from an external component, i.e. a voltage regulator L4947. If this pin is not used and with a +5V supply, the reset pin of the ST6210 can be directly connected to Vdd, providing therefore a high noise immunity on this pin (Figure 11).

After the reset, the I/O configuration has to be checked in order to avoid problems such as short circuits or parasitic drive of external components before the software initialisation. In addition, a system status has to be made to make sure that the program will not restart in a bad step of the process. If necessary, the process can be forced to a clear configuration at the software initialisation phase.

Watchdog. If the program counter is disturbed and the program lost in a loop where the watchdog is not reloaded, the watchdog counts down to zero and resets the MCU in a similar way as the external reset pin. When the program comes out of a loop, an exit condition can be checked and if the condition is not met, the watchdog is activated. An example of reset by watchdog activation is given in Annex 1. In any case, a watchdog reset should never happen in normal operation.

For a safe operation in noisy environment, the user should use a "hardware" watchdog. This circuit is activated when the MCU is supplied with power and when the oscillator runs. Once activated, it can not be deactivated by any means. A "software" activated watchdog can be chosen when a low power consumption mode is required but it does not provide the same level of safety. This watchdog, once

initialized by the software, has the same behaviour as the hardware activated watchdog and can not be deactivated by the program. However, until it is activated there is no watchdog protection.

The two versions of the watchdog ("software" and "hardware" activated) are available on the ST62 MCU.

An embedded counting watchdog can be replaced or doubled with an external analog watchdog designed with a resistance and a capacitor connected to the reset pin. In normal operation, the capacitance is discharged through an I/O port of the MCU at a rhythm defined by the software. The reset occurs if the oscillator stops or the program does not go through the corresponding I/O port drive.

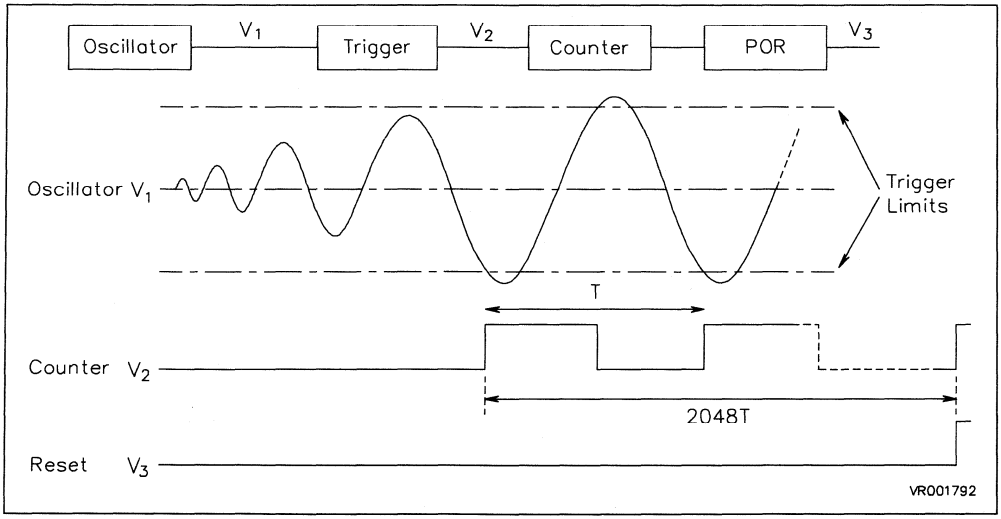
Power On Reset (POR). In the ST6210, both the watchdog and the POR blocks participate to a safe start. When the supply voltage grows above 0.7V to 1V, the oscillator starts. Depending on the type of oscillator (RC, crystal, ceramic resonator), its startup lasts around 2ms to 10ms. Once the oscillator voltage reaches the trigger limits, a clean signal is available and the counter counts 2048 clock periods to ensure a full and valid reset of the ST62. The POR then allows the CPU to exit from the reset state (Figure 12).

Since there is not a precise voltage source inside CMOS technology products and considering that the oscillator startup can vary strongly from one type of oscillator to another, the simplest approach for the user is to make sure that the supply has reached its nominal level 2048 clock periods after the start. In applications supplied directly from the mains, a capacitive supply enables a very fast voltage growth while a resistive supply slows it down.

Disturbances on the supply.

By design, the minimum voltage for watchdog operation is lower than for the CPU (typically 3.5V versus 4V at 8Mhz). So if the supply voltage does not decrease below, for instance 3.5V, the watchdog resets the CPU when it counts down to zero. If the supply goes down below 3.5V, both the CPU and the watchdog are stopped. The watchdog rest-

Figure 12. Power-On-Reset (POR) Timing



arts when the voltage increases up and resets the CPU when it counts down to zero.

If the supply voltage drops down below 0.7 to 1V, the POR acts when the supply rises again. Then two resets can occur, coming from the POR and the watchdog (Figure 13).

If the supply voltage changes, its speed of variation is normally limited by the decoupling capacitance. If the voltage variations remain inside the limits specified for the given oscillator frequency, the ST6210 CPU operation will not be disturbed.

The ST6210 includes an A/D converter, allowing additional supply voltage monitoring to be achieved using an external Zener diode (Figure 14). The circuit consumption is slightly increased due to the polarization of the diode. With the A/D converter, the supply level can be accurately measured and a back-up procedure can be decided if the converted value increases above a certain limit.

Figure 14. Supply Monitoring via The A/D Converter

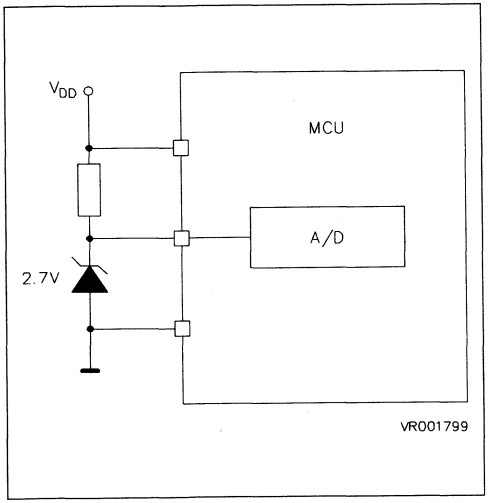
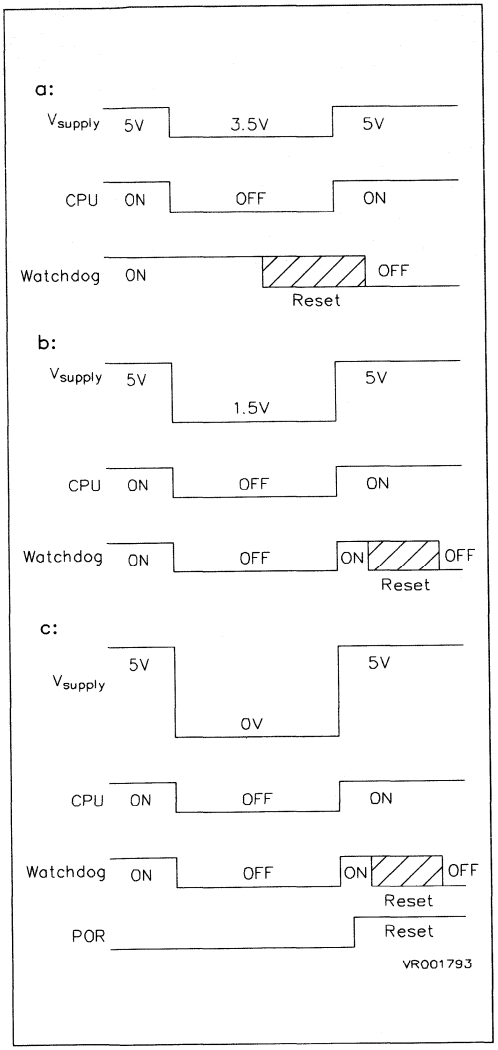


Figure 13. Reset Sequences after Power Disturbances



EXPERIMENTAL RESULTS

The noise immunity of a ST6210 can be tested in a functional manner. One input is forced with high current when the neighbouring input pin is connected to a potentiometer. All other pins are connected in output mode to LEDs. The program converts the analog value to a LED display. When a current is forced in a pin close to the functional pins, no defect appears on the display.

This clamping feature included in the I/O pins can be also used for detection of the mains zero crossing. The mains voltage is directly connected to the ST6210 via a high impedance which limits the current. The internal I/O diodes clamp the signal and the I/O works safely without external diode networks (Figure 15).

Noise generation can be tested using a TEM cell. Such an antenna is a type of coaxial cable with space available inside to put the equipment under test (Figure 16). It can be used either as a noise generator to check the ruggedness of an equipment versus EMI or as a receiver to measure the EMI generated by an equipment. Such test equipment is much less expensive than an anechoic chamber.

Figure 15. Power Control Using ST6210 and Snubberless Triac

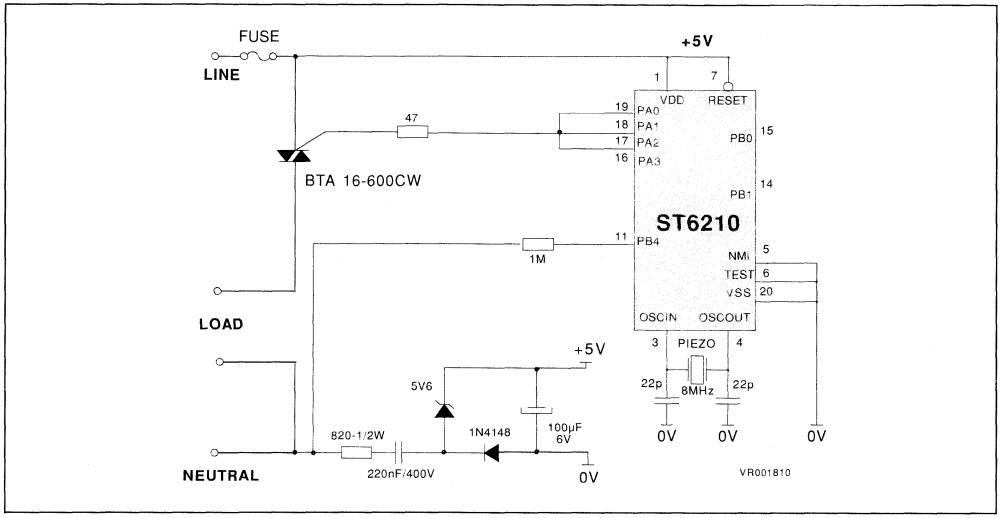
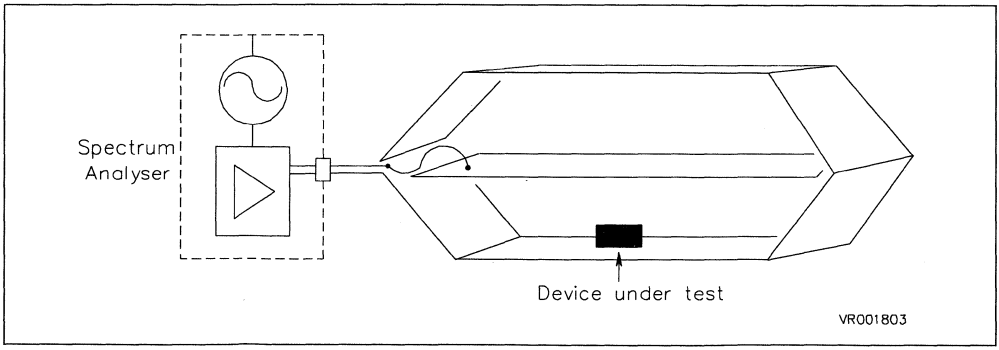


Figure 16. The TEM Cell in Receiver Mode



SUMMARY

Microcontrollers (MCU) are spreading out from applications well protected against noise such as telecom or computer, to noisy environment such as automotive and power control.

Protection against noise goes through the choice of an adapted MCU. The ST62 family, for instance, has been designed to operate safely in disturbed or noise sensitive environment. Its major design characteristics concern the I/O design, the reset modes and the supply voltage range.

Even when using an MCU designed for noisy environment, special care has to be taken during the design on the circuitry, on the PCB lay-out and on the writing of the software. This article presents some concrete solutions applicable to these fields.

With caution on these points, the designer can use MCU's for applications such as motor control, battery charger, light dimming or alarm. And here the real advantages of an MCU can be taken: fast time to market, high flexibility with a minimum of components on the board and treatment of relatively complex algorithms.

Bibliography:

US patent: An integrated Controlled FET switch (J.Stockinger/SGS-THOMSON Microelectronics)

Power semiconductors and EMI reduction (L.Perier/Powertechnics 1-91)

Environmental design rules for Mosfet (B.Maurice/Power transistor manual 91/ SGSTHOMSON)

ANNEX 1

This program checks the flags, the trace points and adjusts the watchdog refresh value. It is organized in such way that the watchdog is reloaded only in the main loop and not in a subroutine or an interrupt routine. It is written for the ST621x/2x MCU. It can be implemented in applications which can start again after a reset and where the reset configuration of the MCU I/O pins may occur without damage in any step of operation of the equipment.

Program Example

```
.def        WDT  0d8h      ; WDT = watchdog timer address
.def        WDM  090h      ; WDM = watchdog trace mask in RAM e.g. reg 090h
      ; WDM is also the WDT refresh value in normal mode main program loop:
            ...
            ...
            LD  A,WDM                 ; read last trace mask
            CPI A, last tracepoint    ; check last tracepoint value
            JRZ  cont1          ; continue if correct, else reset chip
            LDI WDT,01h         ; reset chip if the test fails
            set 1,WDM           ; set bit 1 on WDM (WDM=2d)
cont1:      RET                 ; RET will work like a NOP, if executed in
            RET                 ; the main loop it is used to be sure
            RET                 ; that stack is in the top position.
            RET                 ; The stack has 6 levels hardware stack
            RET                 ; so unnecessary RET are seen as NOP.
            RET
            RETI                ; switches normal flags back
                                ; RETI would cancel the interrupt
                                ; So to be sure not to be
                                ; in interrupt mode
            JRNZ contm          ; check that Zero flag still the same
                                ; JRZ is used in alternance with JRNZ for jump
                                ; to detect if Zero flag is stuck in one level
            LDI WDT,01h         ; reset chip if the test fails
contm:      JRNC contn          ; check that Carry flag still cleared
            LDI WDT,01h         ; reset chip if the test fails
contn:      LD A,WDM            ; load refresh value to WD
            LDI WDM,01h         ; set trace register to initial
                                ; reset position (WDM=1d)
            CPI A,WDMASK        ; WDMASK stored in ROM for double check
            JRZ conto           ; continue if stored and calculated
                                ; values identical
            LDI WDT,01h         ; reset chip if test fails
```

Program example (Continued)

```
conto:      LD  WDT,A          ; refresh the watchdog only here
                               ; the last tracepoint before the normal
                               ; watchdog refresh is the next value of
                               ; the watchdog timer itself
            ...                ; continue with normal program flow

            ...
            LD A,WDM           ; first tracepoint in the program flow
                               ; may be in a subroutine or
                               ; interrupt routine
            CPI A,01h          ; test initial value
            JRZ  cont1         ; reset if not valid
            LDI WDT,01h        ; reset chip if the test fails
cont1:      SET 2,WDM          ; set bit 2 (WDM=5d)
            ...                ; continue with normal program flow
            ...
            LD A,WDM           ; second tracepoint in the program flow
                               ; may be in a subroutine or
                               ; interrupt routine
            CPI A,05h          ; test preceding value
            JRZ cont2          ; reset if not valid
            LDI WDT,01h        ; reset chip
cont2:      SET 3,WDM          ; set next or other bit of WDM (WDM=13d)
                               ; SET,RES combinations for generation
                               ; of binary codes for more than 6 tracepoints
                               ; may be used
            ...                ; continue to normal program flow
```

ST62 IN-CIRCUIT PROGRAMMING

IN-CIRCUIT PROGRAMMING

This note provides information on the steps required in order to perform in-circuit programming of ST62Exx EPROM or OTP devices for both on-chip EPROM and EEPROM.

In-circuit EPROM programming is possible if the relevant pins of the programming socket located on the ST62 EPROM Programming tool (either the ST6 Starter kit, Remote Programming board or Gang programmer) are connected to a 16-pin connector (8x2 header HE10), which must be provided on the application board by the customer.

Note: In-circuit programming embedded in production test is not possible. If the EPROM programmer cable is connected to the application, the RESET signal for instance is tied to GND before and after programming.

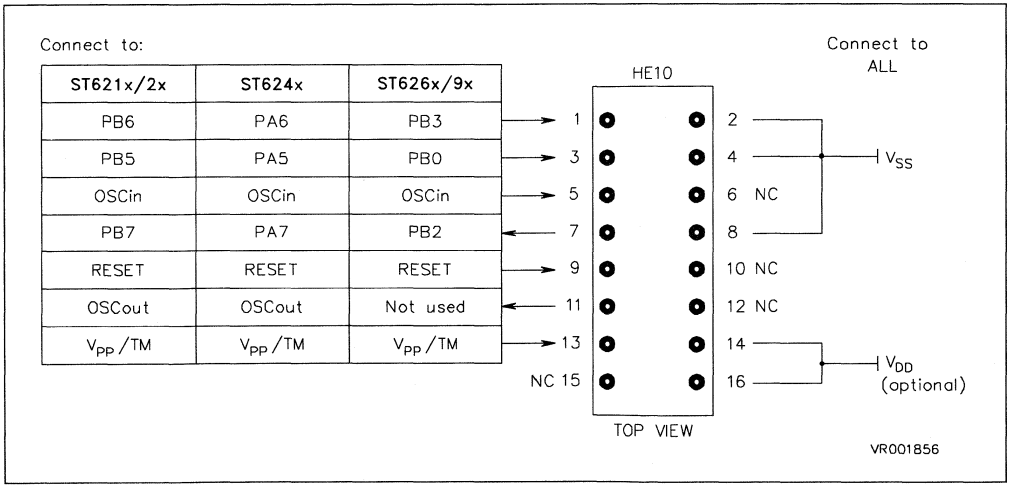
In Circuit Programming Procedure

The procedure for in-circuit programming is as follows:

- Power up the PC and invoke the ST6 EPROM programmer software.
- Connect the programmer to the application board which must be in the power off condition.
- Power on the application board (+5V) and use the ST6 EPROM programmer software in the usual way. The target chip may be supplied by the programmer (in the case where the power supply of the chip can be separated from the remaining parts of the application).

Only a few signals of the 16-pin cable are used. These are listed below including their functional characteristics, seen from the programming tool point of view, and the interconnection to ST62 family members:

Figure 1. 16-pin PCB Socket Connection



The signals shown in Figure 1 should be completely separated from the application circuitry for the time of programming and all signals must be connected to the ST62Exx on the application board. In addition it is mandatory to add a ceramic capacitor with a value of 100nF between V_{PP}/TM and V_{SS}!
Separation of V_{SS} to the application board GND is not necessary.

Programming Conditions

If separation between the ST62Exx and its application circuitry is not possible, certain conditions concerning the application circuitry must be fullfilled:

- V_{PP}/TM (TEST) on the ST62Exx application must not be connected directly to V_{SS}, instead it should be pulled down by a resistor with a minimum value of 10kΩ . It is mandatory to add a ceramic capacitor with a value of 100nF between V_{PP}/TM and V_{SS}!
- Both EPROM programmer pins for V_{SS} must be connected to the V_{SS} input of the ST62Exx to be programmed.
- Connection of the EPROM Programmmer pin V_{DD} is optional (and not recommended). If the ST62Exx chip is supplied by the application power circuit, the supplied V_{DD} voltage must be +5V to avoid excessive current through the ST62Exx CMOS input protection diodes. If the ST62Exx is supplied by the Programmer, the total load current should not exceed 100mA and the capacitive load must be lower than 50µF.
- The RESET pin on the application ST62Exx must be left open or pulled up by a resistor with a minimum value of 2kΩ . The capacitive load should not exceed 1µF.
- OSCin on the application ST62Exx must not be connected to a clock generator. A quartz crystal or a ceramic resonator is allowed.
- The Programmer cable header's Pin 1 and Pin 3 are applied to different members of the ST62Exx families as shown in Figure 1. These signals must not be connected to any other **output** on the application to prevent any voltage contention. Pullup resistors of a minimum value of 2kΩ and pulldown resistors of 10kΩ minimum are allowed.

Note: The connection of Pin 5 of the cable header is not necessary if a high voltage level on the ST62Exx pin is guaranteed. This pin is set to input with pullup mode during reset, meaning another pullup, or CMOS inputs, are allowed for the application.

- Pin 7 of the cable header is applied to the members of the ST6 family, as shown in Figure1. On the application board this signal must not be connected to any other **output**. A pullup resistor of a minimum value of 2kΩ and a pulldown resistor of 2kΩ minimum are allowed.
- ***For ST626X and ST629X only:*** Pin EXTAL = OSCout must be tied directly to V_{DD}. PB7 must not be connected to any other output. PB6 must be at a high voltage level. PB6 is set to input with pullup mode during reset, meaning another pullup, or CMOS inputs, may be connected.
- ***For ST624X only:*** Pin PB0 must be at a high voltage level. It is set to input with pullup mode during reset, meaning another pullup, or CMOS inputs, may be connected.

SGS-THOMSON MICROELECTRONICS

APPLICATION NOTE

DIRECT SOFTWARE LCD DRIVE WITH ST621X AND ST626X

T.Castagnet / J. Nicolai / N. Michel

INTRODUCTION

This note describes a technique for driving a Liquid Crystal Display (LCD) with a standard ST62 microcontroller (MCU), without any dedicated LCD driver. This technique offers a display capability for applications which require a small display at a low cost together with the versatile capabilities of the standard ST62xx MCU. Higher display requirements are easily handled by dedicated members of the ST62 MCU family, for example the ST6240.

The first part of this note describes the typical waveforms required to drive an LCD correctly with a multiplexing rate of 1 or 2 (duplex). The following parts present two solutions based on standard ST62 MCUs driving directly the LCD. The first is based on an ST6215 without using software interrupts and the second on an ST6265 where the LCD is controlled by timer interrupts.

In both examples the program size, the CPU time occupation due to the LCD drive and the number of surrounding components are minimized. Consequently many additional tasks can be added to the MCU program.

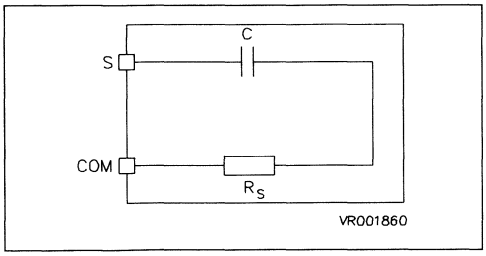
LCD requirements

With a zero Root Mean Square (RMS) voltage applied to it, an LCD is practically transparent. The LCD contrast, which makes the segments turn dark or opaque and thus "on", is caused by the difference between the RMS LCD voltage applied and the LCD threshold voltage, specific to each LCD type.

The applied LCD voltage must alternate to give a zero DC value in order to ensure a long life time of the LCD. The higher the multiplexing rate is, the lower the contrast, also the period of the signal has to be short enough to avoid visible flickering of the LCD display.

The LCD voltage for each segment equals to the difference between the S and COM voltages (see Figure 1).

Figure 1. Equivalent Electrical Schematic of an LCD Segment

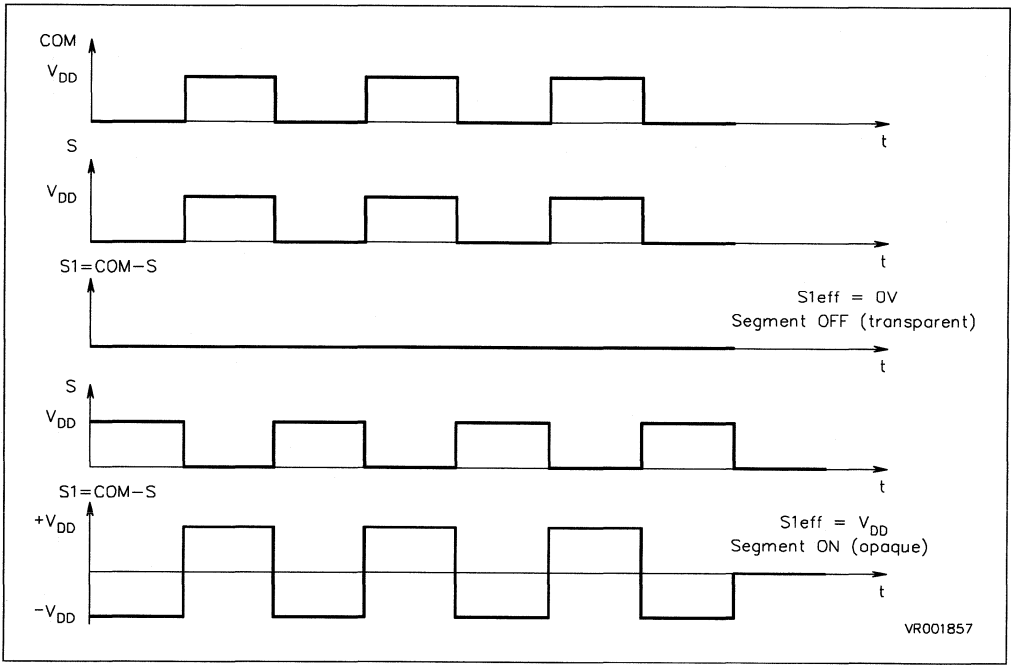


- DC value should never be more than 100mV. Else time life can be shorten.
- Frequency range is 30 - 2000Hz typically. Less, it flickers; more, consumption grows.

Direct LCD drive

Each LCD segment is connected to an I/O "Segment" and to one backplane common to all the segments. A display using S segments is driven with S+1 MCU output lines. The backplane is driven with a signal "COM" controlled between 0 and V_{DD} with a duty cycle of 50%.
When selecting a segment "ON", a signal with opposite polarity to "COM" is sent to the corresponding "Segment" pin. When the non-inverted signal "COM" is sent to the "Segment" pin, the segment is "OFF". Using an MCU the I/O operate in output mode either at the logic levels 0 or 1.

Figure 2. LCD Signals for Direct drive



Duplexed LCD drive

For duplexed drive, two backplanes are used instead of one. Each LCD pin is connected to two LCD segments, each one connected on the other side to one of the two backplanes. Thus, only (S/2)+2 MCU pins are necessary to drive an LCD with S segments.

Three different voltage levels have to be generated on the backplanes : 0, $V_{DD}/2$ and V_{DD}. The "Segment" voltage levels are 0 and Vdd only. The LCD segment is inactive if the RMS voltage is below the LCD threshold voltage and is active if the LCD RMS voltage is above the threshold voltage. Figure 4 shows typical Backplane, Segment and LCD waveforms.

The intermediate voltage $V_{DD}/2$ is only required for the "Backplane" voltages. The ST62 I/O pins selected as "Backplanes" are set by software to output mode for 0 or Vdd levels and to high impedance input mode for $V_{DD}/2$. This voltage Vdd/2 is defined by two equal valued resistors externally connected to the I/O pin.

By using an MCU with flexible I/O pin configuration such as the ST6215 or ST6265, duplexed LCD drive can be made with only 4 additional resistors.

Figure 3. Basic LCD Segment Connection in duplexed mode

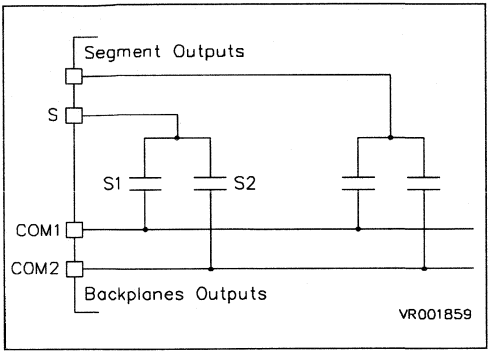
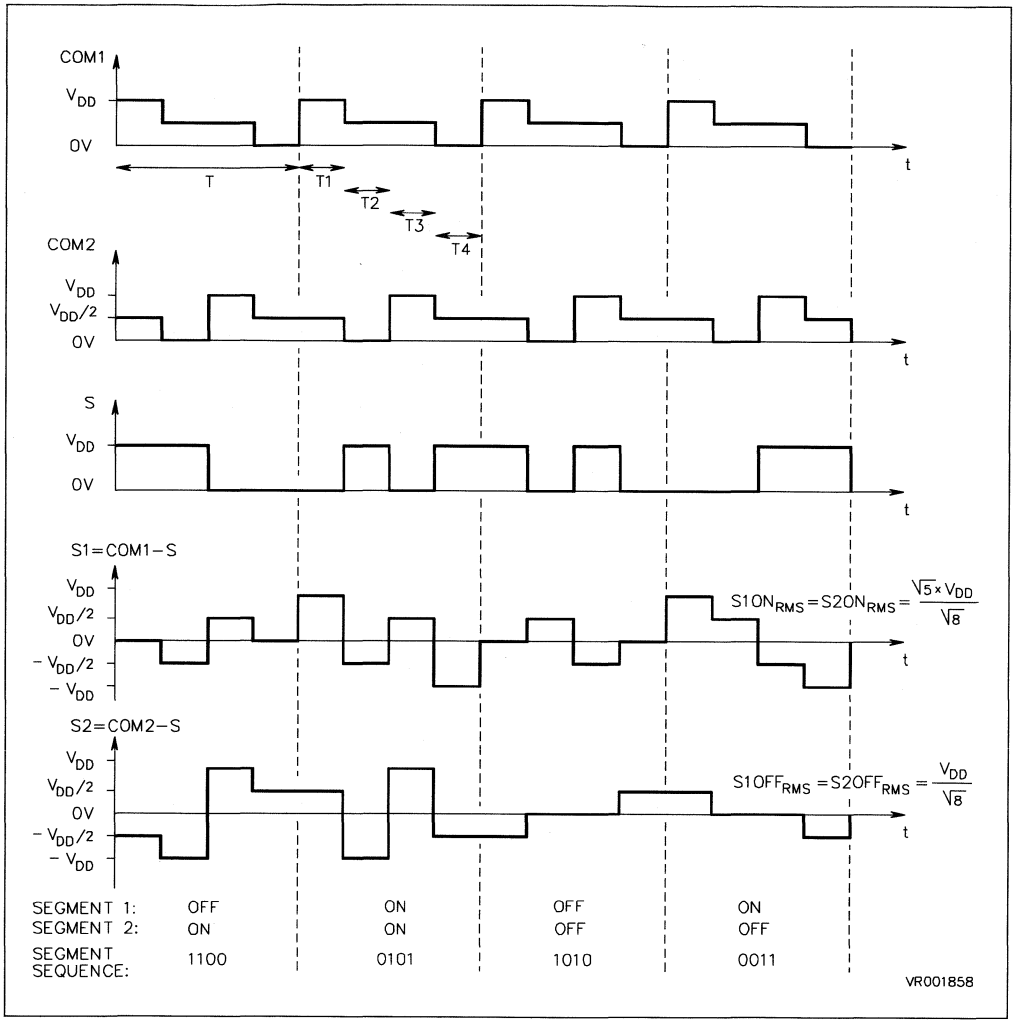


Figure 4. LCD Signals for Duplexed Mode (Used in ST6215 Example)



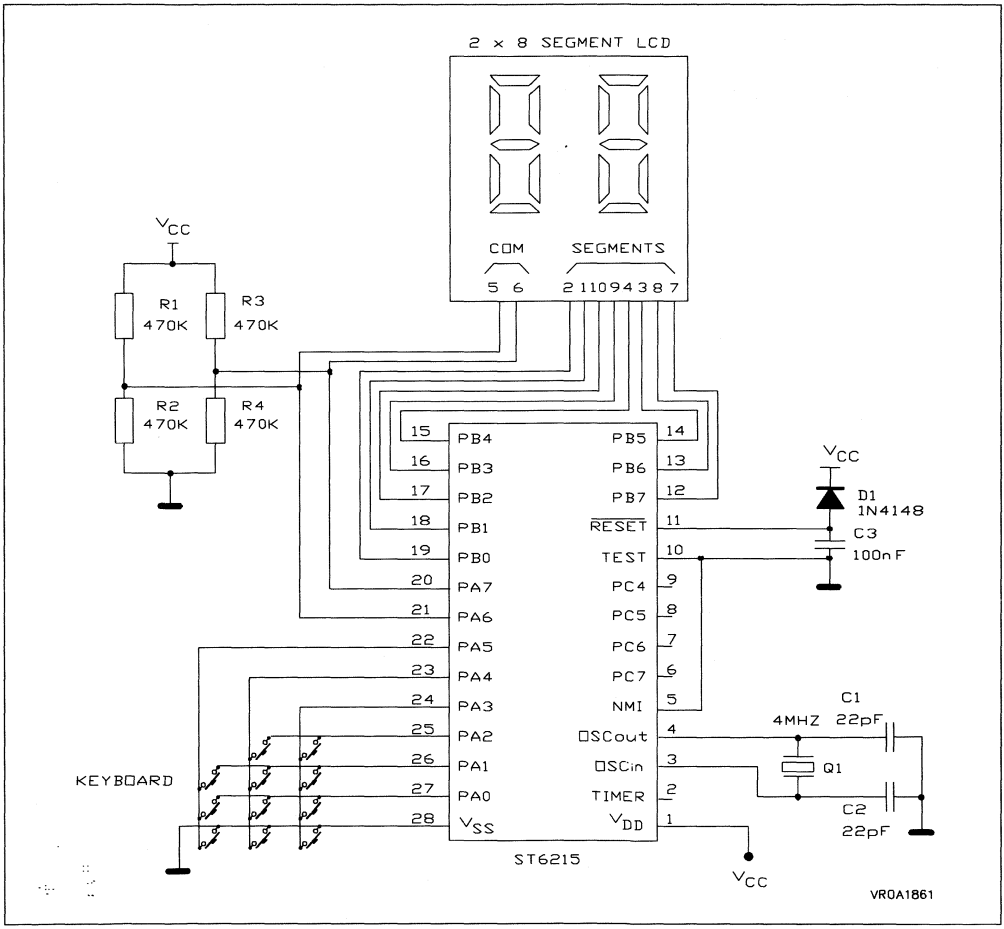
EXAMPLE OF DUPLEXED LCD DRIVE WITH ST6215

The following example describes the drive in duplexed mode of an LCD with an ST6215. The software is made in such way that no interrupt is generated and an OFF state for the LCD is possible. The only components necessary are those to drive the ST62 (oscillator, reset,...) and four resistors to generate the backplane intermediate voltages.

The ST6215 has 20 I/O pins, thus it is able to drive up to 36 LCD segments and 2 backplanes. One digit is defined with 8 segments connected to 2 backplanes. Each digit can display 11 values, from 0 to 9 and no display.

Each value to be displayed is associated to a certain LCD waveform. One LCD waveform period is separated in 4 steps corresponding to the 3 I/O configurations (1-0-input). A look-up table stores the bytes which relate the I/O configuration to the value to display.

Figure 5. ST6215 Based Example

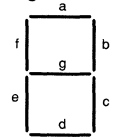


The following tables show an example of linking one LCD digit of the display with the relevant MCU port sequences. The second digit follows the same scheme. These examples are for the sample LCD display, please adapt these tables to your particular LCD.

The digit segments are connected to PB0-PB3 lines in this example. Schematic is shown in Figure 5. Each digit configuration defines a segment status. Each couple of segments defines a timing sequence to be output by MCU "Segment" lines (see Figure 4). These sequences are coded inside the ST6215.

Table 1: Example of drive of a digit with ST6215

Digit Segments to Display



Segment Connections

Segment Line	PB3	PB2	PB1	PB0
COM1 (PA6)	d	e	f	a
COM2 (PA7)	OFF	c	g	b

Digit Required

Segments Status

COM1	ON	ON	ON	ON
COM2	OFF	ON	OFF	ON

Timing Sequences

					MCU Output Sequences
T1	0	0	0	0	0h
T2	0	1	0	1	5h
T3	1	0	1	0	Ah
T4	1	1	1	1	Fh

Digit Required: Blank

Segments Status

COM1	OFF	OFF	OFF	OFF
COM2	OFF	OFF	OFF	OFF

Timing Sequences

					MCU Output Sequences
T1	1	1	1	1	Fh
T2	0	0	0	0	0h
T3	1	1	1	1	Fh
T4	0	0	0	0	0h

Digit Required

Segments Status

COM1	OFF	OFF	OFF	OFF
COM2	OFF	ON	OFF	ON

Timing Sequences

					MCU Output Sequences
T1	1	1	1	1	Fh
T2	1	0	1	0	5h
T3	0	1	0	1	Ah
T4	0	0	0	0	0h

Digit Required

Segments Status

COM1	OFF	OFF	ON	OFF
COM2	OFF	ON	ON	ON

Timing Sequences

					MCU Output Sequences
T1	1	1	0	1	Dh
T2	0	1	1	1	7h
T3	1	0	0	0	8h
T4	0	0	1	0	2h

Digit Required

Segments Status

COM1	ON	ON	OFF	ON
COM2	OFF	OFF	ON	ON

Timing Sequences

					MCU Output Sequences
T1	0	0	1	0	2h
T2	0	0	1	1	3h
T3	1	1	0	0	Ch
T4	1	1	0	1	Dh

Digit Required

Segments Status

COM1	ON	OFF	ON	ON
COM2	OFF	ON	ON	OFF

Timing Sequences

					MCU Output Sequences
T1	0	1	0	0	4h
T2	0	1	1	0	6h
T3	1	0	0	1	9h
T4	1	0	1	1	Bh

Digit Required

Segments Status

COM1	ON	OFF	OFF	ON
COM2	OFF	ON	ON	ON

Timing Sequences

					MCU Output Sequences
T1	0	1	1	0	6h
T2	0	1	1	1	7h
T3	1	0	0	0	8h
T4	1	0	0	1	9h

Digit Required

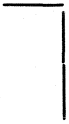
Segments Status

COM1	ON	ON	ON	ON
COM2	OFF	ON	ON	OFF

Timing Sequences

					MCU Output Sequences
T1	0	0	0	0	0h
T2	0	1	1	0	6h
T3	1	0	0	1	9h
T4	1	1	1	1	Fh

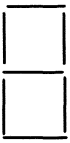
Digit Required



Segments Status				
COM1	OFF	OFF	OFF	ON
COM2	OFF	ON	OFF	ON

Timing Sequences					MCU Output Sequences
T1	1	1	1	0	Eh
T2	0	1	0	1	5h
T3	1	0	1	0	Ah
T4	0	0	0	1	1h

Digit Required



Segments Status				
COM1	ON	ON	ON	ON
COM2	OFF	ON	ON	ON

Timing Sequences					MCU Output Sequences
T1	0	0	0	0	0h
T2	0	1	1	1	7h
T3	1	0	0	0	8h
T4	1	1	1	1	Fh

Digit Required

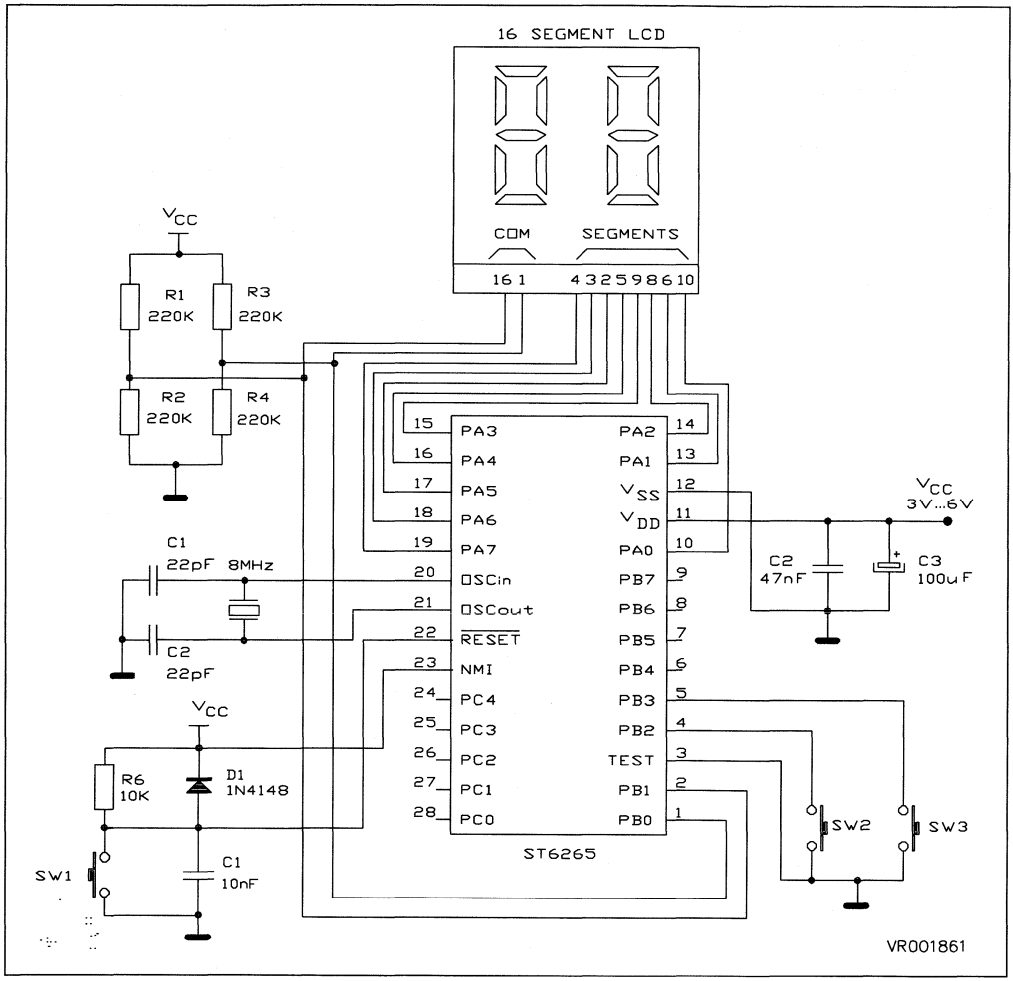
Segments Status				
COM1	ON	OFF	ON	ON
COM2	OFF	ON	ON	ON

Timing Sequences					MCU Output Sequences
T1	0	1	0	0	4h
T2	0	1	1	1	7h
T3	1	0	0	0	8h
T4	1	0	1	1	Bh

EXAMPLE OF DUPLEXED LCD DRIVE WITH ST6265

In this example the LCD drive tasks are controlled by interrupts from an on-chip timer. This software block can be easily included in a central task such as motor control, temperature measurement or heating. Figure 6 shows the circuit of the application. Only 4 resistors are added to drive the LCD in the duplexed mode.

Figure 6. ST6265 Based Example



The software for the LCD display operates in interrupt mode, driving I/O pins by the CPU at instants defined by the Timer1 interrupts. When the LCD drive tasks are finished, the main application program can continue.

The major tasks of the LCD display routine are:

- generation of the alternate signals which control the backplanes (PB0, PB1)
- drive of the LCD segments through PortA
- conversion of the hexadecimal data to decimal data
- program Timer1 for the next drive sequence

The backplane pattern sequences are different between ST6215 and ST6265 examples. The ST6215 software is based on non-symetrical signals (see Figure 4). The backplane patterns are symetrical in the ST6265 software. Both are equivalent on the LCD viewpoint.

The ROM code size used is 300 bytes for the program and 256 bytes for the tables. With an CPU frequency of 8MHz, the display task duration is 240µs and the full task duration including decimal conversion is 510µs. With an LCD period signal of 14ms, the CPU duty cycle of occupation is 2.5% for the LCD drive task. The program can be adjusted to other applications by modifying the timer duration (FASTIM) and the segment drive byte table. The LCD drive phase may also be synchronized to the mains zero crossing or a software loop duration, saving the timer for other tasks.

Summary

The examples presented in this note show that a simple LCD can be driven directly by standard ST621x/2x/6x/9x microcontrollers. The ST62's flexible I/O configuration and the large voltage range of operation of the ST62 family MCU allow an LCD driver to be achieved with very few surrounding components, small CPU time occupation and reduced ROM program size.

Such an approach is a very cost effective solution for simple LCD displays operating with a multiplexing rate of 1 or 2 and up to 36 segments. For LCDs requiring more segment drive capability and/or higher multiplexing rates ST624x and ST628x provide highly integrated and easy to implement solutions.

Such LCD drive features can easily be included in a larger application including keyboard interface, sensor display or motor control.

We thank the company Akotronic for the ST6215 application example they have developed for this note.

Annex 1: Software of the ST6215 based application

Annex 2: Flowchart and software of the ST6265 based application

ANNEX 1: Software of the ST6215 Based Application

```
;*************************************************************************
;*DEMONSTRATION SOFTWARE FOR MANAGEMENT OF A BIPLEXED LIQUID CRISTAL
;*DISPLAY ( LCD ) WITH ST621X OR ST622X SGS THOMSON MICROCONTROLLERS
;*************************************************************************
;*              This program has been developped by AKOTRONIC comp.
;*                    PARC DE LA MOTHE 03400 YZEURE FRANCE
;*************************************************************************
;*                          PROGRAM FOR LIQUID CRISTAL DISPLAY DRIVE
;*************************************************************************
;*                                REGISTER DECLARATION
;*************************************************************************
.ROMSIZE 4 .
VERS "ST6225"
; SWD, 4MHZ
x .DEF 80h!m    ; INDEX REGISTER
Y .DEF 81h!m    ; INDEX REGISTER
V .DEF 82h      ; SHORT DIRECT REGISTER
W .DEF 83h      ; SHORT DIRECT REGISTER
A .DEF 0FFh!m   ; ACCUMULATOR
DRA   .DEF 0C0h ; PORT A DATA REGISTER
DRB   .DEF 0C1h ; PORT B DATA REGISTER
DRC   .DEF 0C2h ; PORT C DATA REGISTER
DDRA  .DEF 0C4h ; PORT A DIRECTION REGISTER
DDRB  .DEF 0C5h ; PORT B DIRECTION REGISTER
DDRC  .DEF 0C6h ; PORT C DIRECTION REGISTER
IOR   .DEF 0C8h ; INTERRUPT OPTION REGISTER
DWR   .DEF 0C9h ; DATA ROM WINDOW REGISTER
ORA   .DEF 0CCh ; PORT A OPTION REGISTER
ORB   .DEF 0CDh ; PORT B OPTION REGISTER
ORC   .DEF 0CEh ; PORT C OPTION REGISTER
ADR   .DEF 0D0h ; A/D DATA REGISTER
ADCR  .DEF 0D1h ; A/D CONTROL REGISTER
PSC   .DEF 0D2h ; TIMER PRESCALER REGISTER
TCR   .DEF 0D3h ; TIMER COUNTER REGISTER
TSCR  .DEF 0D4h ; TIMER STATUS CONTROL REGISTER
WDR   .DEF 0D8h ; WATCHDOG REGISTER

;*************************************************************************
;*                              DATA DECLARATION
;*************************************************************************
TOUCHU .DEF 084h            ;Low Significant DIGIT button  ( L.S. DIGIT )
TOUCHD .DEF 085h            ;More significant DIGIT button ( M.S. DIGIT )
TOUCH  .DEF 086h            ;pushed button
COPYA  .DEF 087h            ;COPY of PORT A
COPYB  .DEF 088h            ;COPY of PORT B
COPYC  .DEF 089h            ;COPY of PORT C
TABD   .DEF 08Ah            ;data/ROM window address to display M.S. DIGIT
TABU   .DEF 08Bh            ;data/ROM window address to display L.S. DIGIT
LOOP   .DEF 08Ch            ;LOOP
RELACHE.DEF 08Dh            ;latch counter
TOUCHP .DEF 08Eh            ;previous valided button
FLAGS  .DEF 08Fh            ;FLAGS : 0/ push on/off
;*************************************************************************
```

ANNEX 1: Software of the ST6215 Based Application (Continued)

```
;*************************************************************************
;*           TABLE 1 of the Low Significant DIGIT ( L.S. DIGIT )
;*************************************************************************
       .ORG  0F00H
       .BYTE 00FH,09FH,06FH,0FFH,0FFH,09FH,06FH,00FH
       .BYTE 04FH,0CFH,03FH,0BFH,05FH,0DFH,02FH,0AFH
       .BYTE 0BFH,0DFH,02FH,04FH,01FH,05FH,0AFH,0EFH
       .BYTE 00FH,05FH,0AFH,0FFH,07FH,09FH,06FH,08FH
       .BYTE 00FH,0DFH,02FH,0FFH,01FH,0DFH,02FH,0EFH
       .BYTE 00H,00H,00H,00H,00H,00H,00H,00H
       .BYTE 00H,00H,00H,00H,00H,00H,00H,00H
       .BYTE 00H,00H,00H,00H,00H,00H,00H,00H
;*************************************************************************
;*************************************************************************
;*           TABLE 2 of the More Significant DIGIT ( M.S. DIGIT )
;*************************************************************************
       .ORG  0F40H
       .BYTE 0FFH,0F0H,0FFH,0F0H,0FFH,0F5H,0FAH,0F0H
       .BYTE 0F2H,0F3H,0FCH,0FDH,0F6H,0F7H,0F8H,0F9H
       .BYTE 0FDH,0F7H,0F8H,0F2H,0F4H,0F6H,0F9H,0FBH
       .BYTE 0F0H,0F6H,0F9H,0FFH,0FEH,0F5H,0FAH,0F1H
       .BYTE 0F0H,0F7H,0F8H,0FFH,0F4H,0F7H,0F8H,0FBH
       .BYTE 00H,00H,00H,00H,00H,00H,00H,00H
       .BYTE 00H,00H,00H,00H,00H,00H,00H,00H
       .BYTE 00H,00H,00H,00H,00H,00H,00H,00H
;*************************************************************************
;*************************************************************************
;*              INTERRUPT VECTORS
;*************************************************************************
.ORG          0FF0h
IT_ADC        NOP
              RETI
IT_TIMER      JP  T_IT_TIMER
IT_PORTBC     JP  T_ITPBC
IT_PORTA      JP   T_ITPA
              NOP
              NOP
              NOP
              NOP
NMI           NOP
              RETI
RES           JP   DEBUT
;*************************************************************************
```

ANNEX 1: Software of the ST6215 Based Application (Continued)

```
;************************************************************************
;*                    INITIALIZATION SUBROUTINE
;************************************************************************
.ORG 880h
DEBUT  RETI                    ; END OF RESET INTERRUPT
       LDI  DDRA,0C7H          ; A0 to A2 PUSH PULL OUTPUT = 0
       LDI  ORA,0C7H           ; A3 to A5 pull up input ; A6 & A7 output = 0
       LDI  DRA,00H            ; A3 to A5 for keyboard  ; A6 -> BP1, A7 -> BP2
       LDI  COPYA,00H
       LDI  DDRB,0FFH          ; B0 A B7 push pull output = 0
       LDI  ORB,0FFH           ; port B controls LC Display
       LDI  DRB,00H            ;
       LDI  COPYB,00H          ;
       LDI  DDRC,00H           ;
       LDI  ORC,00H            ; C4 C5 C6 C7 unused inputs
       LDI  DRC,00H            ;
       LDI  COPYC,00H          ;
       LDI  DWR,3CH            ; origin of the table
       LDI  FLAGS,00h          ; reset flags
;************************************************************************
;*                        END OF INITIALIZATION SUBROUTINE
;************************************************************************
;************************************************************************
;*                           MAIN LCD DRIVE SUBROUTINE
;*
;*  task : generate alternative signals to control LCD backplanes BP1/BP2
;*         drive the segments of the LCD
;*         calculate duration of each duration phase
;************************************************************************
SOMMEIL  RES  0,FLAGS
       LDI  ORA,0CFh           ; A3 becomes interrupt input
       LDI  IOR,10h            ; valid interrupt
       STOP                    ; wait at ON/OFF button activation
       LDI  ORA,0C7h           ;
       CLR  IOR                ; inihibit INTERRUPTS
       CALL CLAVIER            ; keyboard test
       JRR  0,FLAGS,SOMMEIL    ; IF no push on ON/OFF button , THEN stand by
I1BOUCLE RES  0,FLAGS          ; ELSE INITIALIZATION of MAIN LOOP
       LDI  TOUCHU,00h         ; ON/OFF FLAG is reset, UNITEE A 0
       LDI  TOUCHD,00h         ; reset M.S. DIGIT
       LDI  IOR,10h            ; valid interrupts
;************************************************************************
BOUCLE1  LDI  TCR,18           ; initialization of timer
       LDI  TSCR,7Fh           ; program it at 1,5 ms
       LD   A,TOUCHU           ; determine data/rom window address
       SLA  A                  ;
       SLA  A                  ; multiply TOUCHU by 4
       LD   TABU,A             ; initialize data/rom address
       LD   A,TOUCHD           ; of TABLES 1 & 2
       SLA  A                  ;
       SLA  A                  ;
       LD   TABD,A             ;
       CALL DATALCD            ; determine segments driver byte for PHASE 1
       LDI  ORA,47h            ; BP1 = Vdd ; BP2 =  Vdd/2 therefore
       LDI  DDRA,47h           ; A6 becomes push pull output at VDD
       LDI  DRA,0C0h           ; A7 becomes high impedance input
       LD   DRB,A              ; load segments driver byte on port
       CALL CLAVIER            ; test of keyboard
       WAIT
;************************************************************************
```

ANNEX 1: Software of the ST6215 Based Application (Continued)

```
BOUCLE2  LDI  TCR,18          ; timer initialization
         LDI  TSCR,7Fh        ; program it at 1.5 ms
         INC  TABU            ; determine data/rom window address
         INC  TABD            ; of tables 1 & 2 for phase 2
         CALL DATALCD         ; determine segments driver byte for PHASE 2
         LDI  ORA,07h         ; BP1 = Vdd/2 ; BP2 = Vss therefore
         LDI  DRA,40h
         LDI  DDRA,87h        ; A6 becomes high impedance input
         LDI  ORA,87h         ; & A7 becomes push pull output at Vss
         LD   DRB,A           ; load segments driver byte on port
         WAIT
;*****************************************************************************
BOUCLE3  LDI  TCR,18          ; timer initialization
         LDI  TSCR,7Fh        ; program it at 1.5 ms
         INC  TABU            ; determine data/rom window address
         INC  TABD            ; of tables 1 & 2 for phase 2
         CALL DATALCD         ; determine segments driver byte for PHASE 3
         LDI  DRA,0C0h        ; BP1 = Vdd/2 ; BP2 = Vdd therefore
                              ; A6 remains high impedance input
                              ; A7 becomes push pull output at Vdd
         LD   DRB,A           ; load segments driver byte on port
         WAIT
;*****************************************************************************
BOUCLE4  LDI  TCR,18          ; timer initialization
         LDI  TSCR,7Fh        ; program it at 1.5 ms
         INC  TABU            ; determine data/rom window address
         INC  TABD            ; of tables 1 & 2 for phase 2
         CALL DATALCD         ; determine segments driver byte for PHASE 4
         LDI  ORA,07h         ; BP1 = Vss ; BP2 =A Vdd/2 therefore
         LDI  DRA,80h
         LDI  DDRA,47h        ; A6 becomes output at Vss
         LDI  ORA,47h         ; A7 becomes high impedance input
         LD   DRB,A           ; load segments driver byte on port
         WAIT
;*****************************************************************************
FINBOUCLE JRR  0,FLAGS,BOUCLE1 ;IF ON/OFF button remains pushed,
                              ; THEN circuit is in stand by & display is
                              ; ELSE continue digits display
         LDI  DRA,00h
         LDI  DDRA,0C7h
         LDI  ORA,0C7h        ; BP1 & BP2 on output to Vss
         LDI  DRB,00h
PREPSOMM CALL CLAVIER         ; test of keyboard
         LD   A,TOUCH
         CPI  A,0Ah           ; wait falling edge of ON/OFF button
         JRZ PREPSOMM         ; before stop display mode
         JP   SOMMEIL
;*****************************************************************************
;*                              END OF MAIN PROGRAM
;*****************************************************************************
```

ANNEX 1: Software of the ST6215 Based Application (Continued)

```
;*************************************************************************
;*                            TABLE SUBROUTINE
;*
;*  task : define LCD segments driver byte to load on port B
;*         TABU defines half driver byte for L.S.DIGIT
;*         TABD defines half driver byte for M.S.DIGIT
;*************************************************************************
DATALCD  LDI  DWR,3Ch          ; move data/rom window to L.S.DIGIT TABLE 2
         LDI  A,40H
         ADD  A,TABU
         LD   X,A
         LD   A,(X)
         LD   Y,A              ; half segment driver byte is loaded
         LDI  DWR,3Dh          ; move data/rom window to M.S.DIGIT TABLE 1
         LDI  A,40h
         ADD  A,TABD
         LD   X,A
         LD   A,(X)
         AND  A,Y              ; LCD driver byte is loaded in accumulator
         RET
;*************************************************************************
;*                              END OF TABLE SUBROUTINE
;*************************************************************************

;*************************************************************************
;*                           KEYBOARD SUBROUTINE
;*
;*  task : controls display operation
;*        searchs TOUCHU (L.S.DIGIT) & TOUCHD (M.S.DIGIT) displayed data
;*
;*************************************************************************
CLAVIER  LDI  LOOP,02h
CLAVIER1 LD   A,DRA
         ANDI A,38h
         CPI  A,38h            ; test 2 times if some buttons are pushed
         JRNZ CLAVIER2         ; IF yes THEN check them ( A3 -> A5 )
         DEC  LOOP
         JRNZ CLAVIER1
         LDI  TOUCH,0FFh
         JP   CLAVIER4         ; ELSE test if keyboard is changed
CLAVIER2 LD   Y,A
         LD   A,DRA
         ANDI A,38h
         CP   A,Y
         JRZ  TSTCOL           ; IF check is OK , THEN determine column
         LDI  TOUCH,0FFh
         JP   CLAVIER4         ; ELSE test if one button was pushed
TSTCOL   JRR  3,Y,COL1         ; if A3 = 0 then column #1
         JRR  4,Y,COL2         ; if A4 = 0 then column #2
COL3     LDI  TOUCH,3          ; else column #3 is selected so TOUCH <=3
         JP   TSTLIGN
COL2     LDI  TOUCH,2          ;column #2 so TOUCH <=2
         JP   TSTLIGN
COL1     LDI  TOUCH,1          ;column #1 so TOUCH <=1
```

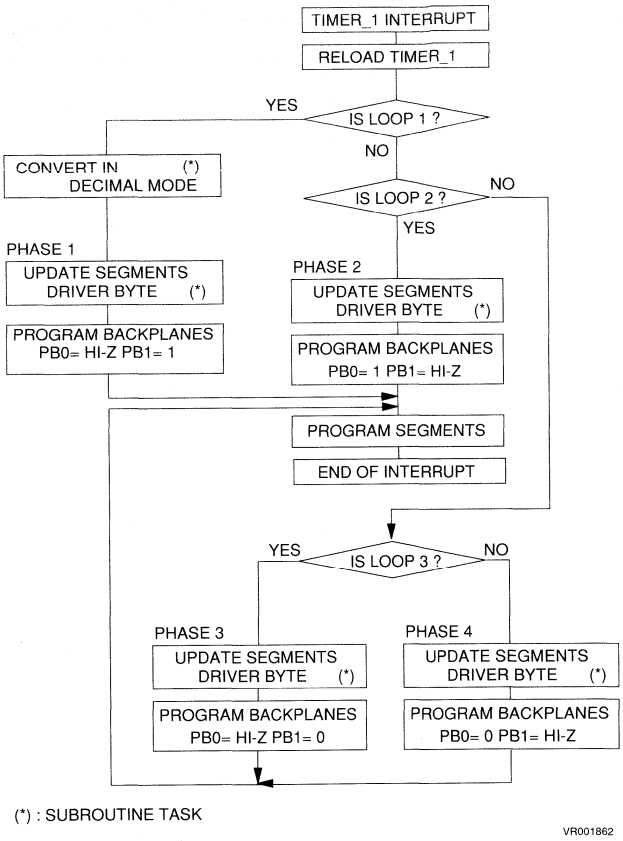
ANNEX 1: Software of the ST6215 Based Application (Continued)

```
TSTLIGN  RES  0,ORA          ;
         RES  1,ORA          ;
         RES  2,ORA          ; A0, A1 & A2 become open drain output
         RES  0,DDRA         ;
         RES  1,DDRA         ;
         RES  2,DDRA         ; then pull up input

         SET  3,DDRA;
         SET  4,DDRA;
         SET  5,DDRA         ; A3, A4 & A5 become open drain output
         SET  3,ORA          ;
         SET  4,ORA          ;
         SET  5,ORA          ; then push pull output at Vss

         LD   A,DRA
         ANDI A,07h
         JRR  0,A,LIGN2      ; if A0 = 0 then row #2
         JRR  1,A,LIGN3      ; if A1 = 0 then row #3
         JRR  2,A,LIGN4      ; if A2 = 0 then row #4
LIGN1    JP   CLAVIER3       ; else row #1 is selected & TOUCH unchanged
LIGN2    LD   A,TOUCH
         ADDI A,3
         LD   TOUCH,A        ; row #2 so TOUCH ⇐ TOUCH + 3
         JP   CLAVIER3
LIGN3    LD   A,TOUCH
         ADDI A,6
         LD   TOUCH,A        ; row #3 so TOUCH ⇐ TOUCH + 6
         JP   CLAVIER3
LIGN4    JRS  0,TOUCH,ONOFF1
         LDI  TOUCH,00h
         JP   CLAVIER3
ONOFF1   LDI  TOUCH,0Ah      ; TOUCH ⇐ 0Ah means action on ON/OFF button

CLAVIER3 RES  3,ORA          ;
         RES  4,ORA          ;
         RES  5,ORA          ; A3, A4 & A5 become open drain ouput
         RES  3,DDRA         ;
         RES  4,DDRA         ;
         RES  5,DDRA         ; then pull up inputs

         SET  0,DDRA         ;
         SET  1,DDRA         ;
         SET  2,DDRA         ; A0, A1 & A2 become open drain output
         SET  0,ORA;
         SET  1,ORA;
         SET  2,ORA          ; then push pull output at Vss

CLAVIER4 LD   A,TOUCH
         CP   A,TOUCHP
         JRNZ CLAVIER5
         JP   FINCLAV        ; IF unchanged state keyboard THEN end
CLAVIER5 LD   TOUCHP,A       ; ELSE TOUCHP ⇐ TOUCH
         CPI  A,0FFh
         JRNZ CLAVIER7       ; IF any keyboard buttons are pushed
         JP   FINCLAV        ; THEN end of subroutine
CLAVIER7 CPI  A,0Ah          ; ELSE test ON/OFF button
         JRZ  ONOFF2         ; IF yes THEN set FLAGS
         LD   A,TOUCHU       ; ELSE shift keyboard value
         LD   TOUCHD,A;
         LD   A,TOUCH;
         LD   TOUCHU,A       ; to be displayed
         JP   FINCLAV

ONOFF2   SET  0,FLAGS

FINCLAV  RET
;*************************************************************************
;*                        END OF KEYBOARD SUBROUTINE
;*************************************************************************
```

ANNEX 1: Software of the ST6215 Based Application (Continued)

```
;************************************************************************
;*                        PORT A INTERRUPT SUBROUTINE
;************************************************************************
T_ITPA      NOP
            RETI
;************************************************************************
;*                     END OF PORT A INTERRUPT SUBROUTINE
;************************************************************************
;************************************************************************
;*                         OTHER INTERRUPTS SUBROUTINE
;************************************************************************
T_IT_TIMER  LDI TSCR,00h
            RETI
T_ITPBC     NOP
            RETI
;************************************************************************
```

ANNEX 2: Flowchart of the ST6265 Based Application



ANNEX 2: Software of the ST6265 Based Application

```
;*************************************************************************
;*                    SGS THOMSON MICROELECTRONICS
;*
;*                  CENTRAL APPLICATIONS LABORATORY
;*
;*                           ROUSSET FRANCE
;*
;*************************************************************************
;*************************************************************************
;*
;*                              19 FEB 1993
;*
;*
;*
;*                                LCD005
;*
;*************************************************************************
;*************************************************************************
;*                         LCD DRIVER SOFTWARE PROGRAM
;*                                 ST62E65
;*               DUPLEXED CONTROL OF  2  DIGITS WITH PORT  A;
;*             PA0 to PA3 FOR DIGIT1;PA4 to PA7 FOR DIGIT2;
;*                           BACPLANES ON PB0 & PB1
;*
;*                      with 4 phase sequences generator
;*                      with Hexadecimal/Decimal DATA conversion
;*
;*************************************************************************
;*************************************************************************
;*                        ST6265/6 Registers Declaration
;*************************************************************************
x       .def 80h                ; Index register.
y       .def 81h                ; Index register.
v       .def 82h                ; Short direct register.
w       .def 83h                ; Short direct register.
a       .def 0ffh               ; Accumulator.
pa      .def 0c0h               ; Port a data register.
pb      .def 0c1h               ; Port b data register.
pc      .def 0c2h               ; Port c data register.
padir   .def 0c4h               ; Port a direction register.
pbdir   .def 0c5h               ; Port b direction register.
pcdir   .def 0c6h               ; Port c direction register.
paopt   .def 0cch               ; Port a option register.
pbopt   .def 0cdh               ; Port b option register.
pcopt   .def 0ceh               ; Port c option register.
ior     .def 0c8h               ; Interrupt Option Register.
drwr    .def 0c9h               ; Data rom window register.
adc     .def 0d0h               ; A/D result register.
adcc    .def 0d1h               ; A/D control register.
psc1    .def 0d2h               ; Timer 1 prescaler register.
tcr1    .def 0d3h               ; Timer 1 counter register.
tscr1   .def 0d4h               ; Timer 1 status control register.
```

ANNEX 2: Software of the ST6265 Based Application (Continued)

```
mc        .def 0d5h             ; Timer 2 mode control register.
sc0       .def 0d6h             ; Timer 2 status/control register.
sc1       .def 0d7h             ; Timer 2 status/control register.
rc        .def 0d9h             ; Timer 2 reload/capture register.
cp        .def 0dah             ; Timer 2 compare register.
lr        .def 0dbh             ; Timer 2 load register.

wdt       .def 0d8h             ; Watchdog register.

oscr      .def 0dch             ; Oscillator control register.

lvi       .def 0ddh             ; Multiplex register / lvi flag register.

spirad    .def 0e0h             ; SPI register RAD.
spidiv    .def 0e1h             ; SPI register DIV.

spimod    .def 0e2h             ; SPI register MOD.

mbr       .def 0e8h             ; memory bank register.

eecr      .def 0eah             ; eeprom control register.
;***************************************************************************
;*                        RAM DATA DEFINITION
;***************************************************************************

duty0     .def 087h,0ffh,0ffh ; user request reference
counter1 .def 089h,0ffh,0ffh ; duty0 variation rate control byte

DIG1      .def 0a6h,0ffh,0ffh ; data/rom @ of DIGIT1 segment driver
DIG2      .def 0a7h,0ffh,0ffh ; data/rom @ of DIGIT2 segment driver
LCDCTL    .def 0a8h,0ffh,0ffh ; LCD phase sequence control byte
pbbuf     .def 0aah,0ffh,0ffh ; buffer byte of data port b
BCD       .def 0a9h,0ffh,0ffh ; BinCodDec converted DATA

AUX1      .def 0a3h,0ffh,0ffh ; accumulator save byte
AUX2      .def 0a4h,0ffh,0ffh ; data/rom window register save byte
AUX3      .def 0a5h,0ffh,0ffh ; x register save byte
;***************************************************************************
;*                        EQUATES DEFINITION
;***************************************************************************
FASTIM    .equ 036h      ; duration of each LCD drive phase (14 ms at 8 MHz)
;***************************************************************************
;***************************************************************************
;*                     MACROFUNCTIONS DEFINITION
;***************************************************************************
.macro jumpnc jpadress,?lbl
       jrc lbl
       jp jpadress
lbl
.endm
;***************************************************************************
.macro jumpz jpadress,?lbl
       jrnz lbl
       jp jpadress
lbl
.endm
;***************************************************************************
```

ANNEX 2: Software of the ST6265 Based Application (Continued)

```
;************************************************************************
;*                        INTERRUPT VECTORS
;************************************************************************
.org      0ff0h
it_tim1   jp  LCD_LP          ; timer1 interrupt synchronizes LCD drive
it_tim2   nop                 ; it.timer2
          reti
it_pc_spi nop                 ; it.port c & SPI
          reti
it_pa_pb  nop                 ; it. port a & port b
          reti
          nop
          nop
          nop
          nop
nmi       nop
          reti
res       jp  START

;************************************************************************
;*                    MAIN PROGRAM EXAMPLE
;************************************************************************
.org      0880h
START     reti  ; end of reset interrupt
          call INIT
MAIN      call DUTY0
          jp   MAIN
;************************************************************************
;*                      END OF MAIN PROGRAM
;************************************************************************


;************************************************************************
;*                  MAIN INITIALIZATION SUBROUTINE
;************************************************************************
INIT      ldi    wdt,00000111b ; watchdog initialization
          ldi    eecr,040h     ; EEPROM in stand by for power saving
          ldi    oscr,008h     ; CKOUT  output  disabled for power saving
          ldi    pbdir,00110000b   ; b0 & b1 is common :Hi/Impedance Input
          ldi    pbopt,00110000b   ; b2 &b3 is input for +/- push button
          ldi    pbbuf,00000011b   ; pb buffer byte load
          ldi    pb,00000011b       ; b6,b7 in input with pull up
          ldi    padir,0ffh    ; a0 to a7 in push pull output
          ldi    paopt,0ffh    ; pa = driver of LCD segments
          ldi    pa,00h        ; output is zero
          ldi    pcdir,00h     ;
          ldi    pcopt,00h     ; c0 -> c7 inputs with pull up
          ldi    pc,00h        ;
          ldi    drwr,3ch      ; data/rom window origin
          clr    LCDCTL        ; clear LCD phase sequence control
          ldi    ior , 010h    ; interrupt validation
          ldi    tcr1 ,009h    ; load timer1 for LCD phase generation
          ldi    tscr1,07fh    ; timer initialization
          clr    duty0
          ldi wdt, 00000111b   ; hello watchdog
          ret
;************************************************************************
;*                    END OF MAIN INITIALIZATION
;************************************************************************
```

ANNEX 2: Software of the ST6265 Based Application (Continued)

```
;************************************************************************
;*    MAIN TASK EXAMPLE : ACQUISITION OF USER SPEED REFERENCE DUTY0      *
;*                                                                       *
;*       duty0 is a user reference that can vary from 0 to 255d          *
;************************************************************************
DUTY0    jrr    2,pb,slower   ; if PB2=0,then slower ; priority on slower
         jrr    3,pb,faster   ; if PB3=0,then faster ; else continue
         ret                  ; if PB2 & PB3 are high; then continue
slower   ld     a,duty0
         cpi    a,000h
         jumpz  retour
         ld     a,counter1
         addi   a,01h
         ld     counter1,a
         jumpnc retour
         dec    duty0         ; increment duty cycle
         ret
faster   ld     a,duty0
         cpi    a,0ffh
         jrz    retour
         ld     a,counter1
         addi   a,01h
         ld     counter1, a
         jrnc   retour
         inc    duty0         ; decrement duty cycle
retour   ret
;************************************************************************
;*                    end subroutine get_dut0
;************************************************************************


;************************************************************************
;*            DUPLEXED LCD DRIVER INTERRUPT SUBROUTINE
;*
;*  task :  generate alternative signals of LCD backplanes control
;*          drive the segments of LCD through port a
;*          calculate duration of each LCD phase
;*
;************************************************************************
LCD_LP   ldi    wdt,0ffh      ; hello watchdog
;        ld     AUX1, a       ; |
;        ld     a, x          ; | save context of main task (if needed)
;        ld     AUX3, a       ; |
;        ldi    tscr1,00h     ; timer stop (if needed)
         ldi    tcr1 , FASTIM ; LCD phase duration calculation
         ldi    tscr1, 07fh   ; timer initialization
;************************************************************************
;* LCD phase generation can be here synchronized by other clock system.
;*          for instance : mains voltage synchronization or external clock
;************************************************************************
         jrr    0,LCDCTL,LOOP1; determine phase1 operation
         jrr    1,LCDCTL,LOOP2; determine phase2 operation
         jrr    2,LCDCTL,LOOP3; determine phase3 operation
```

ANNEX 2: Software of the ST6265 Based Application (Continued)

```
;*************************   PHASE 4    *********************************
LOOP4    inc    DIG1          ; increment DIG1 & DIG2 for phase 4
         inc    DIG2          ;
         call   DATALCD       ; calculate phase4 segments driver byte
         ldi    pbbuf,00000011b   ; pb buffer load
         ldi    pb   ,00000011b   ; pb1 in High Impedance (HI) & pb0 to 0
         ldi    pbdir,00110001b   ;
         ldi    pbbuf,00000010b   ; pb buffer load
         ldi    pb   ,00000010b   ;
         ld     pa,a          ; load segment driver byte on port a
         ld     a,AUX3        ;|
         ld     x,a           ;|
         ld     a,AUX2        ;|return to main tasks with context ;
         ld     drwr,a        ;| ( AUX2 <== drwr in main program );
         ld     a,AUX1        ;|
         clr    LCDCTL        ; end of loop4 & full LCD sequence
         reti
;*************************   PHASE 3    *********************************
LOOP3    inc    DIG1          ; increment DIG1 & DIG2 for phase 3
         inc    DIG2
         call   DATALCD       ; calculate phase3 segments driver byte
         ldi    pbopt,00110000b   ; pb0 in HI & pb1 to 0
         ldi    pbdir,00110010b   ;
         ldi    pbbuf,00000001b   ; pb buffer load
         ldi    pb,   00000001b   ;
         ld     pa ,a         ; load segment driver byte on port a
;        ld     a,AUX3        ;|
;        ld     x,a           ;|
;        ld     a,AUX2        ;|return to main tasks with context
;        ld     drwr,a        ;| ( AUX2 <== drwr in main program )
;        ld     a,AUX1        ;|
         set    2,LCDCTL      ; end of loop3
         reti
;*************************   PHASE 2    *********************************
LOOP2    inc    DIG1          ; increment DIG1 & DIG2 for phase 2
         inc    DIG2
         call   DATALCD       ; calculate phase2 segments driver byte
         ldi    pbopt,00110000b   ; pb1 in HI & pb0 to 1
         ldi    pbdir,00110001b   ;
         ldi    pbopt,00110001b   ;
         ld     pa,a          ; load segment driver byte on port a
;        ld     a,AUX3        ;|
;        ld     x,a           ;|
;        ld     a,AUX2        ;|return to main tasks with context
;        ld     drwr,a        ;| ( AUX2 <== drwr in main program )
;        ld     a,AUX1        ;|
         set    1,LCDCTL      ; end of loop2
         reti
```

ANNEX 2: Software of the ST6265 Based Application (Continued)

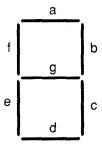
```
;*************************   PHASE 1    *********************************
LOOP1    call   DECCONV        ; do hexa-decimal data conversion
         call   DATALCD        ; calculate phase1 segment driver byte
                               ; driver data is stored in accumulator

         ldi    pbbuf,00000011b   ; pb buffer load
         ldi    pb,00000011b      ; b1 to 1  &   b0 in HI
         ldi    pbdir,00110010b   ; b2 - > b7 unchanged
         ldi    pbopt,00110010b   ;
         ld     pa,a              ;load segment driver byte on port a
;        ld     a,AUX3            ;
;        ld     x,a               ;
;        ld     a,AUX2            ; return to main tasks with context
;        ld     drwr,a            ;(AUX2 <== drwr in main program)
;        ld     a,AUX1            ;
         set    0,LCDCTL          ; end of loop1
         reti
;*****************************************************************************
;*                    END OF LCD DRIVER SUBROUTINE
;*****************************************************************************

;*****************************************************************************
;*             DEC DATA/LCD SEGMEMTS CONVERSION SUBROUTINE
;*
;* task : calculate the segments driver byte of the LCD
;*         depends on the displayed data AND on the # of phase
;*        based on DIG1 & DIG2 calculation
;*        LCD segments driver full byte is in accumulator
;*****************************************************************************
DATALCD  ldi    drwr,3ch       ; move data/rom window on DIGIT2 table
         ld     a,DIG2
         ld     x,a
         ld     a,(x)
         ld     y,a            ; load DIGIT2 driver half byte (MSB)
         ldi    drwr,3dh       ; move data/rom window on DIGIT1 table
         ld     a,DIG1
         ld     x,a
         ld     a,(x)
         and    a,y            ; load DIGIT1 driver half byte (LSB)
         ret
;*****************************************************************************
;*             END OF DATA/LCD CONVERSION SUBROUTINE
;*****************************************************************************
```

ANNEX 2: Software of the ST6265 Based Application (Continued)

```
;*************************************************************************
;*         HEXADECIMAL – > DECIMAL DATA CONVERSION SUBROUTINE
;*
;* task : convert an hexadecimal data in a Binary Coded Decimal data
;*        determine data/rom window address of segments driver byte
;*
;*
;*         hexadecimal data is a stable data varying from 0 to 256
;*        result is a percent decimal data varying from 0 to 99
;*        BCD  Binary Coded Decimal data
;*
;*       for instance, data is duty0
;*
;*        DIG1 = data/rom WDW @ of DIGIT1 segments driver = 01wxyzAB
;*        DIGIT1 = wxyz ( from 0 to 9 ) and
;*        AB are defined by phase operation : AB = 00 for phase1 ...
;*                                            AB = 11 for phase4
;*        DIG2 = data/rom WDXW @ of DIGIT2 segments driver
;*         similar writing than DIG1
;*
;*************************************************************************
DECCONV  jrr    7,duty0,ET1   ; DATA <== duty0
         ldi    drwr,039h     ; b7 of DATA is 1 : 50 < DATA < 99
         jp     ET2
ET1      ldi    drwr,038h     ; b7 of DATA is 0 : 00 < DATA < 49
ET2      ld     a,duty0       ;
         rlc    a             ;
         rlc    a             ;
         rlc    a             ; calculation of BinCodDec DATA
         rlc    a             ; address
         rlc    a             ;
         rlc    a             ;
         rlc    a             ;
         rlc    a             ;
         set    6,a           ;
         res    7,a           ;
         ld     x,a           ;
         ld     a,(x)         ;
         ld     BCD,a         ; BCD <= Binary Coded Decimal DATA
         andi   a,00fh        ; determine DIGIT1 value and the
         sla    a             ; data/rom address of DIGIT1 segments
         sla    a             ; driver
         addi   a,040h        ;
         ld     DIG1,a        ; DIG1 = data/rom @ of segment driver
         ld     a,BCD         ; determine DIGIT2 value and the
         andi   a,0f0h        ; data/rom address of DIGIT2 segments
         rlc    a             ;  driver
         rlc    a             ;
         rlc    a             ;
         rlc    a             ;
         rlc    a             ;
         sla    a             ;
         sla    a             ;
         addi   a,040h        ;
         ld     DIG2,a        ; DIG2 = data/rom @ of LCD driver
         ret
;*************************************************************************
;*           END OF HEXA – > DECIMAL DATA CONVERSION SUBROUTINE
;*************************************************************************
```

ANNEX 2: Software of the ST6265 Based Application (Continued)

```
;************************************************************************
;*    TABLE OF HEXA - > DECIMAL DATA CONVERSION ( 00 TO 49 BinCodDec )
;************************************************************************

.org      0e00h

.byte     00h,01h,02h,02h,03h,04h,05h,05h
.byte     06h,07h,08h,09h,09h,10h,11h,12h
.byte     12h,13h,14h,15h,16h,16h,17h,18h
.byte     19h,20h,20h,21h,22h,23h,23h,24h
.byte     25h,26h,27h,27h,28h,29h,30h,30h
.byte     31h,32h,33h,34h,34h,35h,36h,37h
.byte     37h,38h,39h,40h,41h,41h,42h,43h
.byte     44h,45h,45h,46h,47h,48h,48h,49h
;************************************************************************
;*       END OF HEXA -> DECIMAL DATA CONVERSION TABLE ( 00 to 49 )
;************************************************************************

;************************************************************************
;*    TABLE OF HEXA -> DECIMAL DATA CONVERSION ( 50 to 99 BinCodDec )
;************************************************************************

.org      0e40h

.byte     50h,51h,52h,52h,53h,54h,55h,55h
.byte     56h,57h,58h,59h,59h,60h,61h,62h
.byte     62h,63h,64h,65h,66h,66h,67h,68h
.byte     69h,70h,70h,71h,72h,73h,73h,74h
.byte     75h,76h,77h,77h,78h,79h,80h,80h
.byte     81h,82h,83h,84h,84h,85h,86h,87h
.byte     88h,88h,89h,90h,91h,91h,92h,93h
.byte     94h,95h,95h,96h,97h,98h,98h,99h
;************************************************************************
;*       END OF HEXA - > DECIMAL DATA CONVERSION TABLE ( 50 to 99 )
;************************************************************************

;************************************************************************
;*            SEGMENT CONTROL WITHOUT ANY POINT DISPLAY
;*PA0,4 --> SEG CB
;*PA1,5 --> SEG DH
;*PA2,6 --> SEG DH
;*PA3,7 --> SEG GA
;*
;*PA0,PA1,PA2,PA3--> DIGIT1
;*PA4,PA5,PA6,PA7--> DIGIT2
;*
;*Backplane #1 (pb1) biases C,D,E,G
;*Backplane #2 (pb0) biases A,B,F,H
;*
;*These tables are dedicated to one LCD type with two digits and LCD
;*control is described above ; when LCD is changing these tables have
;* to be  modified
```



ANNEX 2: Software of the ST6265 Based Application (Continued)

```
;*********************************************************************
;*                      DIGIT2  TABLE
;*********************************************************************
        .org    0f00h
        .byte   08fh,02fh,07fh,0dfh,0efh,0efh,01fh,01fh
        .byte   01fh,06fh,0efh,09fh,04fh,06fh,0bfh,09fh
        .byte   06fh,0afh,09fh,05fh,04fh,03fh,0bfh,0cfh
        .byte   00fh,03fh,0ffh,0cfh,0efh,06fh,01fh,09fh
        .byte   00fh,02fh,0ffh,0dfh,04fh,02fh,0bfh,0dfh
        .byte   0ffh,0ffh,00fh,00fh,000h,000h,000h,000h
        .byte   000h,000h,000h,000h,000h,000h,000h,000h
        .byte   000h,000h,000h,000h,000h,000h,000h,000h
;*********************************************************************
;*                      DIGIT1  TABLE
;*********************************************************************
        .org    0f40h
        .byte   0f8h,0f2h,0f7h,0fdh,0feh,0feh,0f1h,0f1h
        .byte   0f1h,0f6h,0feh,0f9h,0f4h,0f6h,0fbh,0f9h
        .byte   0f6h,0fah,0f9h,0f5h,0f4h,0f3h,0fbh,0fch
        .byte   0f0h,0f3h,0ffh,0fch,0feh,0f6h,0f1h,0f9h
        .byte   0f0h,0f2h,0ffh,0fdh,0f4h,0f2h,0fbh,0fdh
        .byte   0ffh,0ffh,0f0h,0f0h,000h,000h,000h,000h
        .byte   000h,000h,000h,000h,000h,000h,000h,000h
        .byte   000h,000h,000h,000h,000h,000h,000h,000h
;*********************************************************************
```

EUROPE

DENMARK

2730 HERLEV
Herlev Torv, 4
Tel. (45-44) 94.85.33
Telex: 35411
Telefax: (45-44) 948694

FINLAND

LOHJA SF-08150
Ratakatu, 26
Tel. (358-12) 155.11
Telefax. (358-12) 155.66

FRANCE

94253 GENTILLY Cedex
7 - avenue Gallieni - BP. 93
Tel.: (33-1) 47.40.75.75
Telex: 632570 STMHQ
Telefax: (33-1) 47.40.79.10

67000 STRASBOURG
20, Place des Halles
Tel. (33-88) 75.50.66
Telefax: (33-88) 22.29.32

GERMANY

8011 GRASBRUNN
Bretonischer Ring 4
Postfach 1122
Tel.: (49-89) 460060
Telefax: (49-89) 4605454
Teletex: 897107=STDISTR

6000 FRANKFURT
Gutleutstrasse 322
Tel. (49-69) 237492-3
Telefax: (49-69) 231957
Teletex: 6997689=STVBF

3000 HANNOVER 51
Rotenburger Strasse 28A
Tel. (49-511) 615960-3
Teletex: 5118418 CSFBEH
Telefax: (49-511) 6151243

8500 NÜRNBERG 20
Erlenstegenstrasse, 72
Tel.: (49-911) 59893-0
Telefax: (49-911) 5980701

7000 STUTTGART 31
Mittlerer Pfad 2-4
Tel. (49-711) 13968-0
Telefax: (49-711) 8661427

ITALY

20090 ASSAGO (MI)
V.le Milanofiori - Strada 4 - Palazzo A/4/A
Tel. (39-2) 89213.1 (10 linee)
Telex: 330131 - 330141 SGSAGR
Telefax: (39-2) 8250449

40033 CASALECCHIO DI RENO (BO)
Via R. Fucini, 12
Tel. (39-51) 593029
Telex: 512442
Telefax: (39-51) 591305

00161 ROMA
Via A. Torlonia, 15
Tel. (39-6) 8443341
Telex: 620653 SGSATE I
Telefax: (39-6) 8444474

NETHERLANDS

5652 AR EINDHOVEN
Meerenakkerweg 1
Tel.: (31-40) 550015
Telex: 51186
Telefax: (31-40) 528835

SPAIN

08021 BARCELONA
Calle Platon, 6 4th Floor, 5th Door
Tel. (34-3) 4143300-4143361
Telefax: (34-3) 2021461

28027 MADRID
Calle Albacete, 5
Tel. (34-1) 4051615
Telex: 46033 TCCEE
Telefax: (34-1) 4031134

SWEDEN

S-16421 KISTA
Borgarfjordsgatan, 13 - Box 1094
Tel.: (46-8) 7939220
Telex: 12078 THSWS
Telefax: (46-8) 7504950

SWITZERLAND

1218 GRAND-SACONNEX (GENEVA)
Chemin Francois-Lehmann, 18/A
Tel. (41-22) 7986462
Telex: 415493 STM CH
Telefax: (41-22) 7984869

UNITED KINGDOM and EIRE

MARLOW, BUCKS
Planar House, Parkway
Globe Park
Tel.: (44-628) 890800
Telex: 847458
Telefax: (44-628) 890391

AMERICAS

BRAZIL

05413 SÃO PAULO
Av. Henrique Schaumann 286-CJ33
Tel. (55-11) 883-5455
Telex: (391)11-37988 "UMBR BR"
Telefax : (55-11) 282-2367

CANADA

KANEPEAN ONTARIO K2H 9C4
301 Moodie Drive
Suite 307
Tel. (613) 829-9944

U.S.A.

NORTH & SOUTH AMERICAN
MARKETING HEADQUARTERS
2000 East Bell Road
Phoenix, AZ 85022
1-602) 867-6100

SALES COVERAGE BY STATE

ALABAMA
Huntsville - (205) 533-5995

ARIZONA
Phoenix - (602) 867-6217

CALIFORNIA
Santa Ana - (714) 957-6018
San Jose - (408) 452-8585

COLORADO
Boulder (303) 449-9000

ILLINOIS
Schaumburg - (708) 517-1890

INDIANA
Kokomo - (317) 455-3500

MASSACHUSETTS
Lincoln - (617) 259-0300

MICHIGAN
Livonia - (313) 953-1700

NEW JERSEY
Voorhees - (609) 772-6222

NEW YORK
Poughkeepsie - (914) 454-8813

NORTH CAROLINA
Raleigh - (919) 787-6555

TEXAS
Carrollton - (214) 466-8844

FOR RF AND MICROWAVE
POWER TRANSISTORS CON-
TACT
THE FOLLOWING REGIONAL
OFFICE IN THE U.S.A.

PENNSYLVANIA
Montgomeryville - (215) 361-6400

ASIA / PACIFIC

AUSTRALIA

NSW 2220 HURTSVILLE
Suite 3, Level 7, Otis House
43 Bridge Street
Tel. (61-2) 5803811
Telefax: (61-2) 5806440

HONG KONG

WANCHAI
22nd Floor - Hopewell centre
183 Queen's Road East
Tel. (852) 8615788
Telex: 60955 ESGIES HX
Telefax: (852) 8656589

INDIA

NEW DELHI 110001
Liaison Office
62, Upper Ground Floor
World Trade Centre
Barakhamba Lane
Tel. (91-11) 6424603
Telex: 031-66816 STMI IN
Telefax: (91-11) 6424615

MALAYSIA

PETALING JAYA, 47400
11C, Jalan SS21/60
Damansara Utama
Tel.: (03) 717 3976
Telefax: (03) 719 9512

PULAU PINANG 10400
4th Floor - Suite 4-03
Bangunan FOP-123D Jalan Anson
Tel. (04) 379735
Telefax (04) 379816

KOREA

SEOUL 121
8th floor Shinwon Building
823-14, Yuksam-Dong
Kang-Nam-Gu
Tel. (82-2) 553-0399
Telex: SGSKOR K29998
Telefax: (82-2) 552-1051

SINGAPORE

SINGAPORE 2056
28 Ang Mo Kio - Industrial Park 2
Tel. (65) 4821411
Telex: RS 55201 ESGIES
Telefax: (65) 4820240

TAIWAN

TAIPEI
12th Floor
325, Section 1 Tun Hua South Road
Tel. (886-2) 755-4111
Telex: 10310 ESGIE TW
Telefax: (886-2) 755-4008

JAPAN

TOKYO 108
Nisseki - Takanawa Bld. 4F
2-18-10 Takanawa
Minato-Ku
Tel. (81-3) 3280-4121
Telefax: (81-3) 3280-4131