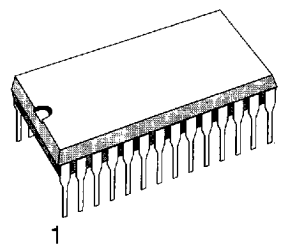
SGS-THOMSON MICROELECTRONICS

ST6210-ST6215
ST6220-ST6225

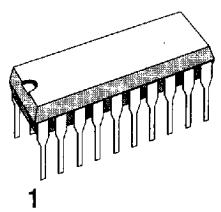
8-BIT HCMOS MCUs WITH A/D CONVERTER

- 3.0 to 6.0V Supply Operating Range
- 8 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: 1828 bytes (ST6210,15)
  3876 bytes (ST6220,25)
- Data ROM: User selectable size (in program ROM)
- Data RAM: 64 bytes
- PDIP20, PSO20 (ST6210,20) packages
- PDIP28, PSO28 (ST6215,25) packages
- 12/20 fully software programmable I/O as:
  - Input with pull-up resistor
  - Input 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
- 8 bit counter with a 7-bit programmable prescaler (Timer)
- Digital Watchdog
- 8 bit A/D Converter with up to 8 (ST6210, ST6220) and up to 16 (ST6215, ST6225) analog inputs
- On-chip clock oscillator
- Power-on Reset
- One external not maskable interrupt
- 9 powerful addressing modes
- The development tool of the ST621x, ST622x microcontrollers consists of the ST621x-EMU emulation and development system connected via a standard RS232 serial line to an MS-DOS Personal Computer

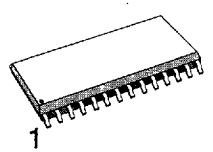
Device Summary page 3/48



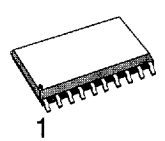
PDIP28



PDIP20



PSO28



PSO20

(Ordering Information at the end of the datasheet)

Figure 1. ST6210,ST6220 Pin Configuration

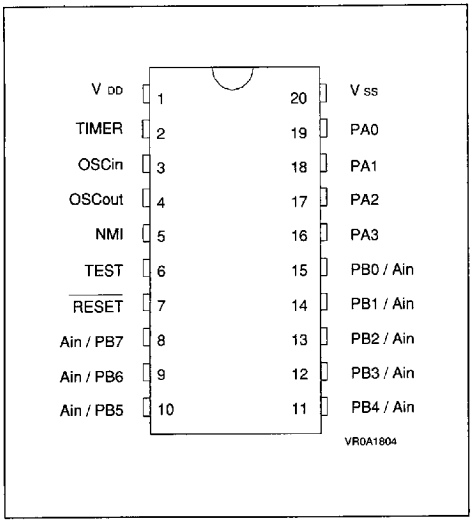


Figure 2. ST6215,ST6225 Pin Configuration

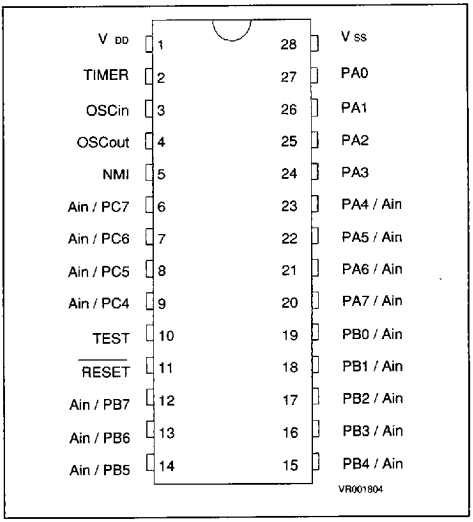
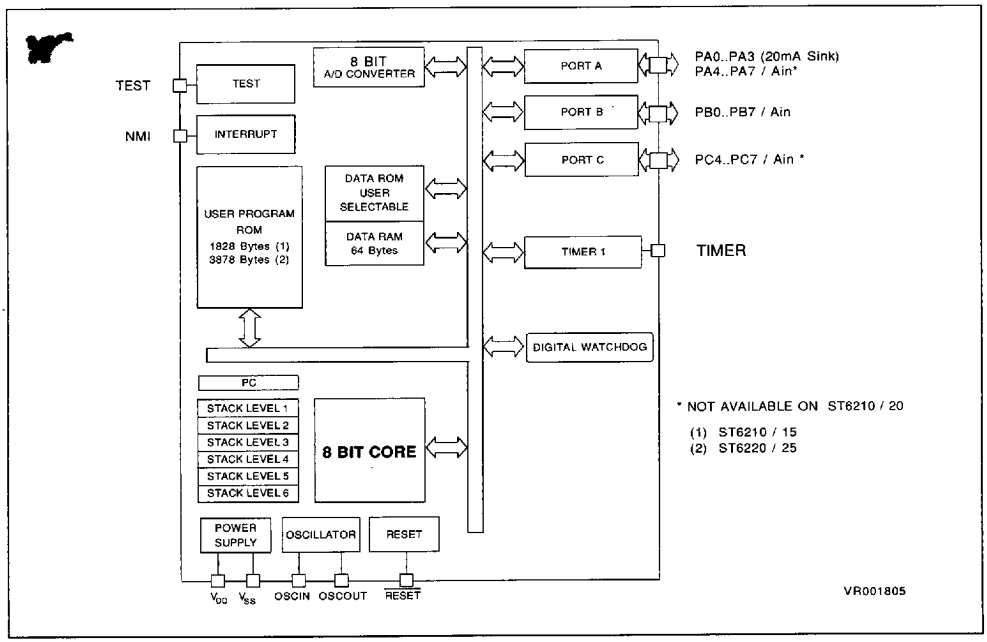


Figure 3. ST6210,15,20,25 Block Diagram



GENERAL DESCRIPTION

The ST6210, ST6215, ST6220 and ST6225 microcontrollers are members 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 surrounded by a combination of on-chip peripherals (macrocells). The macrocells of the ST6210, ST6215, ST6220 and ST6225 are: the Timer peripheral that includes an 8-bit counter with a 7-bit software programmable prescaler (Timer), the 8-bit A/D Converter with up to 8 (ST6210, ST6220) and up to 16 (ST6215, ST6225) analog inputs (A/D inputs are alternate functions of I/O pins), the Digital Watchdog (DWD). Thanks to these peripherals these devices are well suited for automotive, appliance and industrial applications. The ST62E10, ST62E15, ST62E20 and ST62E25 EPROM versions are available for prototypes and low-volume production; also OTP versions are available. The only difference between ST6210,15 and ST6220,25 is the program memory size which is 2K bytes for the ST6210,15 and 4K bytes for the ST6220,25.

DEVICE SUMMARY

Device	ROM (Bytes)	I/O Pins
ST6210	2K	12
ST6215	2K	20
ST6220	4K	12
ST6225	4K	20

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, a ceramic resonator or an external clock signal can be connected between these two pins in order to allow the correct operation of the MCU with various stability/cost trade-offs. The OSCIN pin is the input pin, the OSCOUT pin is the output pin.

$\overline{\text{RESET}}$. The active low $\overline{\text{RESET}}$ pin is used to restart the microcontroller to the beginning of its program.

TEST. The TEST pin is used to place the MCU into special operating mode. The TEST must be held at VSS 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 not maskable interrupt to the MCU. The NMI is falling edge sensitive. On ST6210,15 and ST6220,25 the user can select as ROM mask option (see option list at the end of the datasheet) the availability of an on-chip pull-up at NMI pin. On EPROM/OTP versions this pull-up is not available and should be provided externally.

TIMER. This is the timer 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. On ST6210,15 and ST6220,25 the user can select as ROM mask option (see option list at the end of the datasheet) the availability of an on-chip pull-up at TIMER pin. On EPROM/OTP versions this pull-up is not available and should be provided externally.

PA0-PA3,PA4-PA7(*). These 8 lines are organized as one I/O port (A). Each line may be configured under software control as inputs with or without internal pull-up resistors, interrupt generating inputs with pull-up resistors, open-drain or push-pull outputs. PA0-PA3 can also sink 20mA for direct led driving while PA4-PA7 can be programmed as analog inputs for the A/D converter.
(*) PA4-PA7 are not available on ST6210, ST6220.

PB0-PB7. These 8 lines are organized as one I/O port (B). Each line may be configured under software control as inputs with or without internal pull-up resistors, interrupt generating inputs with pull-up resistors, open-drain or push-pull outputs and as analog inputs 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 inputs with or without internal pull-up resistors, interrupt generating inputs with pull-up resistors, open-drain or push-pull outputs and as analog inputs for the A/D converter.
(*) PC4-PC7 are not available on ST6210, ST6220.

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 5; 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

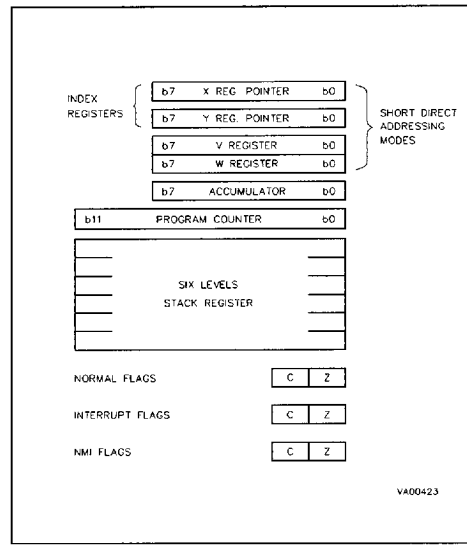
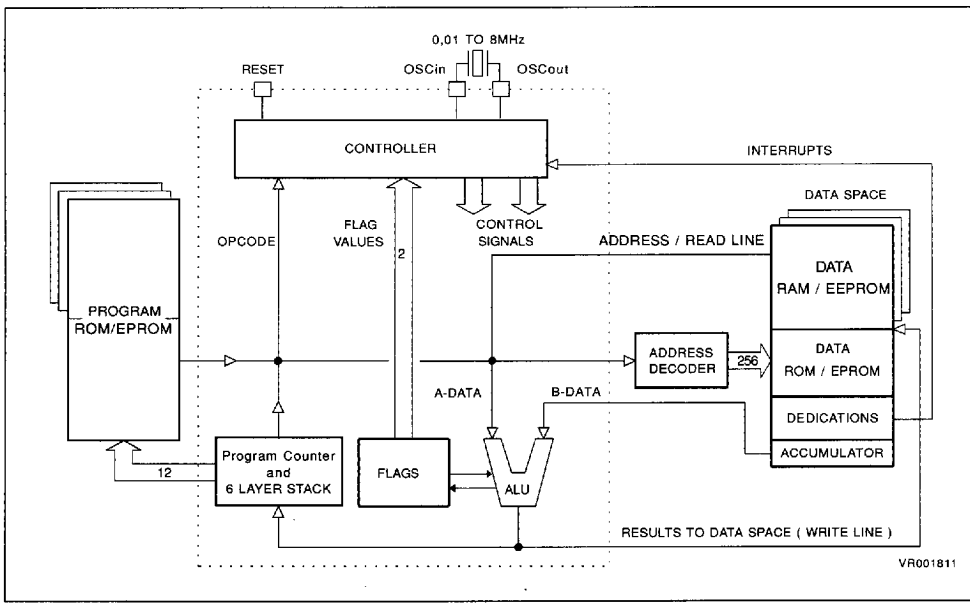


Figure 5. ST62xx Core Block Diagram



ST6xx CORE (Continued)

Indirect Registers (X, Y). These two indirect registers are used as pointers to the 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, the further program space can be addressed by using the Program Bank Switch register.
The PC value is incremented, after it is read 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 to3 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 routine (Not-maskable interrupt, normal interrupt or main routine). The flags are not cleared during the context switching and so remain in the state they were at the exit of the last routine switching.

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.

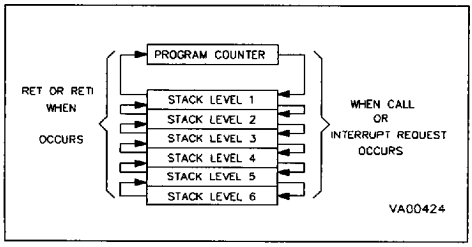
The switching between the three sets of flags is automatically performed when an NMI, an interrupt or a RETI instructions occurs. As the NMI mode is automatically selected after the reset of the MCU, the ST62xx core uses at first the NMI flags.

ST6xx CORE (Continued)

Stack

The ST62xx core includes 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 6. Since the accumulator, as all other data space registers, is not stored in this 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 6. Stack Operation



MEMORY SPACES

The MCUs operate in three different memory spaces: Program Space, Data Space, and Stack Space. A description of these spaces is shown in the following tables.

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 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 2-Kbyte ROM banks.

Table 1. ST6210,15 Program ROM Memory Map

Device Address	Description
0000h-07FFh	Not Implemented
0800H-087Fh	Reserved
0880h-0F9Fh	User Program ROM 1828 Bytes
0FA0h-0FEFh	Reserved
0FF0h-0FF7h	Interrupt Vectors
0FF8h-0FFBh	Reserved
0FFCh-0FFDh	NMI Vector
0FFEh-0FFFh	User Reset Vector

Table 2. ST6220,25 Program ROM Memory Map

Device Address	Description
0000h-007Fh	Reserved
0080h-0F9Fh	User Program ROM 3872 Bytes
0FA0h-0FEFh	Reserved
0FF0h-0FF7h	Interrupt Vectors
0FF8h-0FFBh	Reserved
0FFCh-0FFDh	NMI Vector
0FFEh-0FFFh	User Reset Vector

MEMORY SPACES (Continued)

Table 3. ST6210,15,20,25 Data Memory Space

NOT IMPLEMENTED	000h 03Fh
DATA ROM WINDOW 64 BYTES	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

* WRITE ONLY REGISTER

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 addressing. 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 (see Figure 9).

This window is located from address 40h to address 7Fh in the Data space and allows the direct reading of the bytes from address 000h to address 03Fh in the ROM memory. 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 Data ROM Window register (DRW register).

The RAM memory can be also extended by the addition of 64 bytes RAM banks addressed as being located between the addresses 00h and 7Fh.

In the ST6210, ST6215, ST6220 and ST6225 products the data space includes 60 bytes of RAM, the accumulator (A), the indirect registers (X), (Y), the short direct registers (V), (W), the I/O port registers, the peripheral data and control registers, the interrupt option register and the Data ROM Window register (DRW register).

As the data space is less than 256 bytes the ST62xx core can directly address this area and the Data Bank Switch register (DRBR) has not been implemented.

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.

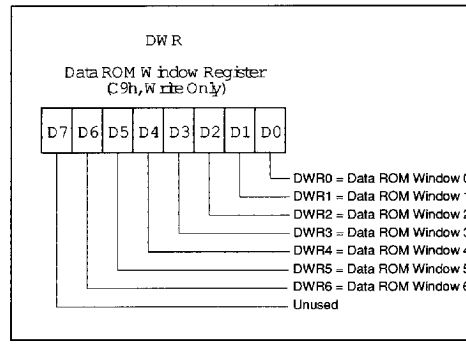
MEMORY SPACES (Continued)

Read-only Data Window register (DWR)

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 7). The DWR register is not cleared at reset, therefore it must be written to before the first access to the Data ROM window area.

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 Register

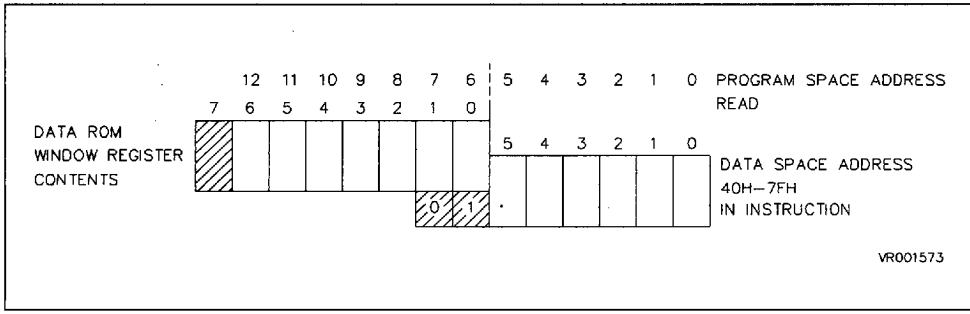


D7. This bit is not used.

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

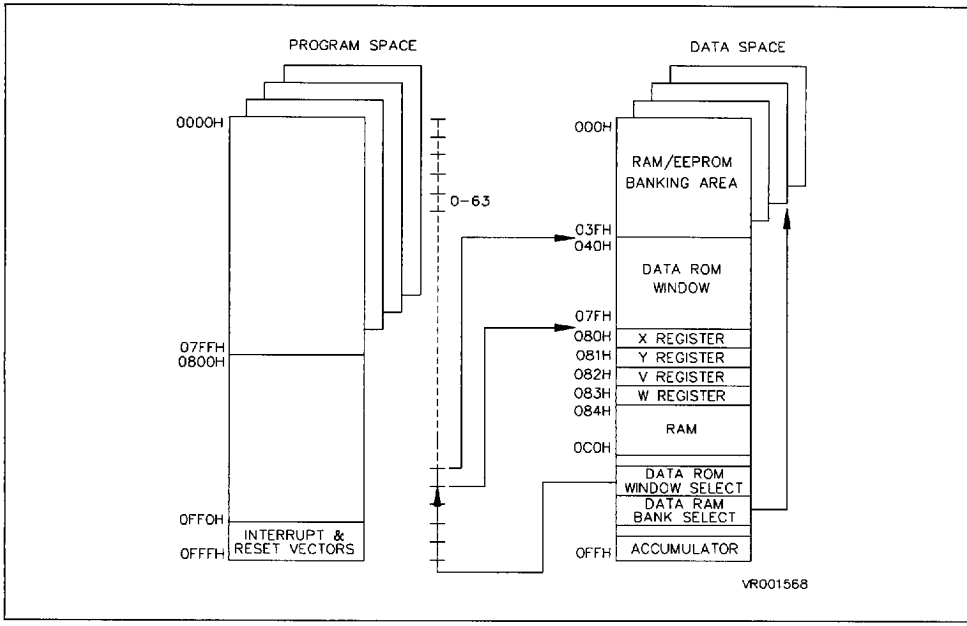
This register is undefined on reset. Neither read nor single bit instructions may be used to address this register.

Figure 7. Data ROM Window Memory Addressing



MEMORY SPACES (Continued)

Figure 9. Memory Addressing Description Diagram



TEST MODE

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

INTERRUPT

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 (see Table 1). 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 ST6210, ST6215, ST6220 and ST6225 microcontrollers have six different interrupt sources associated to different interrupt vectors as it is described in table below.

Table 4. Interrupt Vector/Source Relationship

Interrupt Source	Associated Vector	Vector Address
NMI pin	Interrupt vector #0 (NMI)	(FFCh, FFDh)
Port A pins	Interrupt vector #1	(FF6h, FF7h)
Port B pins	Interrupt vector #2	(FF4h, FF5h)
Port C pins	Interrupt vector #2	(FF4h, FF5h)
TIMER peripheral	Interrupt vector #3	(FF3h, FF2h)
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 ST6210, ST6215, ST6220 and ST6225 this vector is associated with the external falling edge sensitive interrupt pin (NMI).
- The interrupt vector located at addresses FF6h, FF7h is named interrupt vector #1. It is associated with Port A pins and can be programmed by software either in the falling edge detection mode or in the low level sensitive detection mode according to the code loaded in the Interrupt Option Register (IOR).
- The interrupt vector located at addresses FF4h, FF5h is named interrupt vector #2. It is associated with Port B and C pins and can be programmed by software either in the falling edge detection mode or in the positive edge detection mode according to the code loaded in the Interrupt Option Register (IOR).
- The two interrupt vectors located respectively at addresses FF3h, FF2h and addresses FF1h, FF0h are respectively named interrupt vector #3 and #4. Vector #3 is associated to the TIMER peripheral and vector #4 to the A/D converter peripheral.
  All the on-chip peripherals have an interrupt request flag bit (TMZ for timer, EOC for A/D), 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) that must be set to one to allow the transfer of the flag bit to the core.

Interrupt Priority

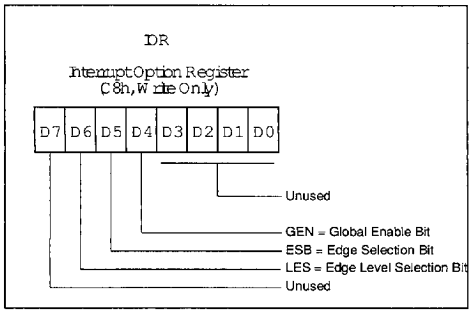
The non-maskable interrupt request has the highest priority and can interrupt any other interrupt routines at any time, nevertheless the four other interrupts can not interrupt each other. If more than one interrupt request are pending, they are processed by the ST62xx core according to their priority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

INTERRUPT (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 10. Interrupt Option Register



D7. D3-D0 These bits are 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 (excluding NMI) are disabled.

This register is cleared on reset.

Table 5. Interrupt Option Register

GEN	SET	Enable all the interrupts of the product
	CLEAR	Disable all the interrupts of the product
ESB	SET	Rising edge mode on interrupt input #2
	CLEAR	Falling edge mode on interrupt input #2
LES	SET	Level sensitive mode on interrupt input #1
	CLEAR	Falling edge mode on interrupt input #1

External Interrupts Operating Modes

The NMI interrupt is associated to the external interrupt pin of the ST6210, ST6215, ST6220 and ST6225 devices. This pin is falling edge sensitive and the interrupt pin signal is latched by a flip-flop which is automatically reset by the core at the beginning of the non-maskable interrupt service routine. A schmitt trigger is present on NMI pin.

The two interrupt sources associated with the falling/rising edge mode of the external interrupt pins (Ports A-vector #1, Ports B and C-vector #2) are connected to two internal latches. Each latch is set when a falling/rising edge occurs during the processing of the first one, will be processed as soon as the first one has been finished (if there is not a 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.

Note

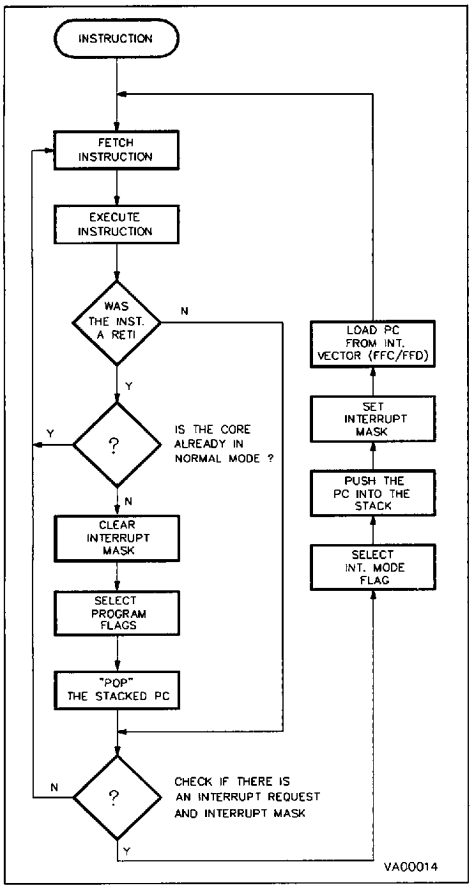
On ST6210,15 and ST6220,25 the user can select the availability of an on-chip pull-up at NMI pin as ROM mask option (see option list at the end of the datasheet).

When GEN = "0", the NMI interrupt is active but cannot cause a restart from STOP/WAIT modes

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.

INTERRUPT (Continued)

Figure 11. Interrupt Processing Flow-Chart



The following list summarizes the interrupt procedure:

- Interrupt detection
- The flags C and Z of the main routine are exchanged with the flags C and Z of the interrupt routine (resp. 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)
- The edge flip-flop is reset
- The related interrupt vector is loaded in the PC.
- 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)
- Interrupt servicing
- Return from interrupt (RETI)
- Automatically the ST62xx core switches back to the normal flags (resp 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.

INTERRUPT (Continued)

Interrupt Request and Mask Bits

Interrupt Option Register, IOR
Location 0C8h

- **GEN**. If this bit is set all the ST6210, ST6215, ST6220 and ST6225 interrupts are enabled, if reset all the interrupt are disabled (excluding the NMI).
- **ESB**. If this bit is set all the inputs 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 into this register are not used.

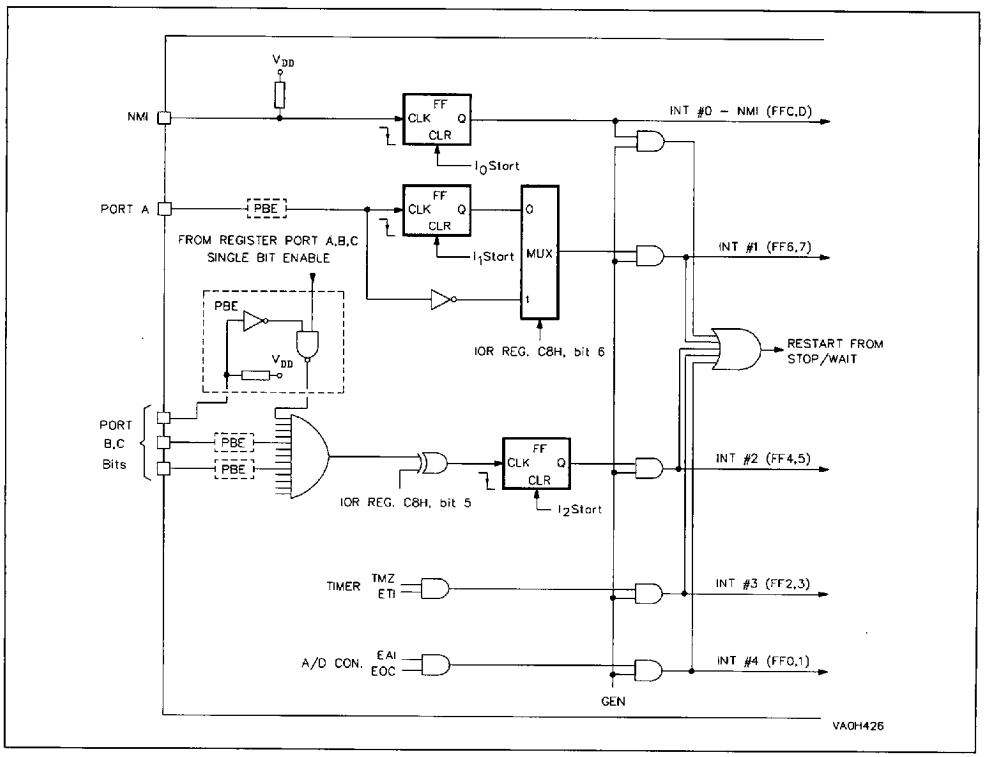
Timer Peripheral, TSCR register
Location D4h

- **TMZ bit**. 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 bit**. This bit, when set, enables the timer interrupt request.

A/D Converter Peripheral, ADCR register
Location D1H

- **C bit**. 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 bit**. This bit, when set, enables the A/D converter interrupt request.

Figure 12. Interrupt Circuit Diagram



RESET

The ST6210, ST6215, ST6220 and ST6225 MCUs 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 $\overline{\text{RESET}}$ pin can be connected to a device of the application board in order to restart the MCU during its operation. The activation of the 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 and has a schmitt trigger input. 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 running correctly (normal RUN or WAIT modes).

If the 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 with pull-up resistors switched on state. 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 with ppull-up resistors on state 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 ensure that the oscillator becomes completely stabilized.
Then, the initialization sequence is started.

Power-on Reset

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 with pull-up state) 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. Then, the initialization sequence is executed.

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

Notes

To have a correct 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

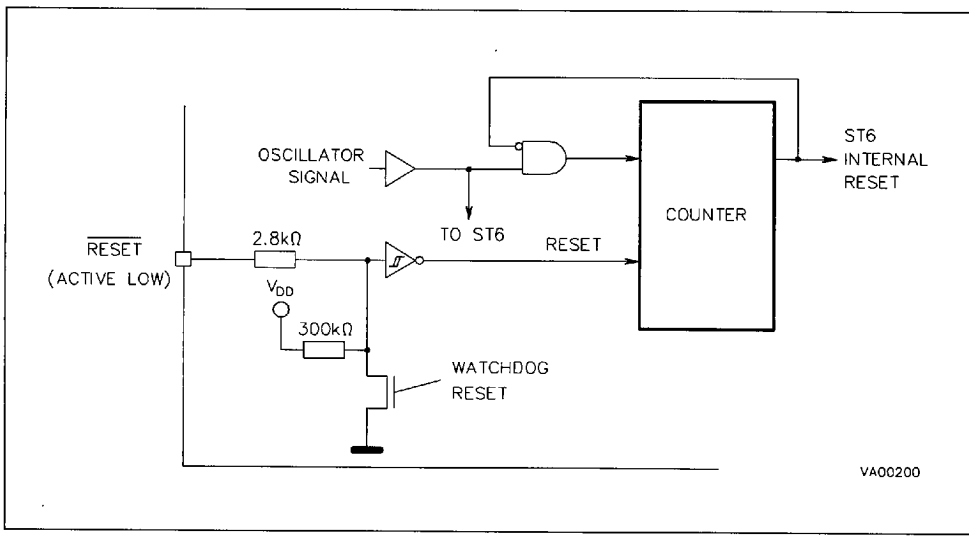
The ST6210, ST6215, ST6220 and ST6225 provide an on-chip watchdog/timer function in order to provide a graceful recovery from a software upset. If the watchdog register is not refreshed, preventing the end-of-count being reached, an internal circuit pulls down the reset pin. The MCU will enter the reset 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 reset pin. This causes the positive transition at the reset pin and terminates the reset state.

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 not be less than 30KΩ. If the value is lower than 30KΩ 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.

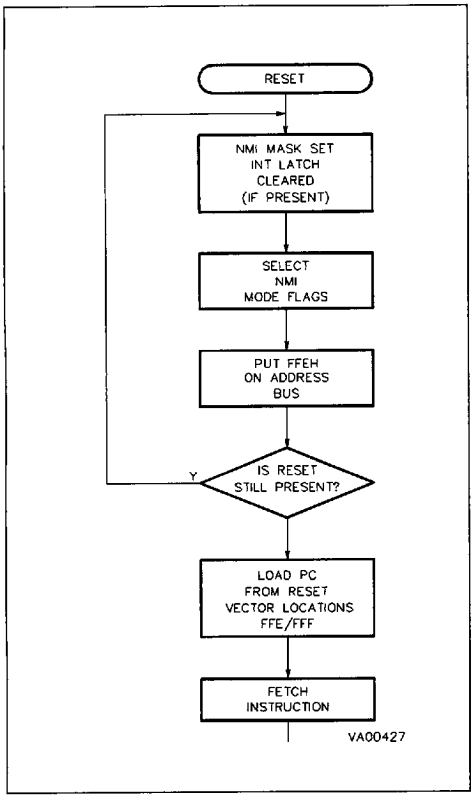
The POR device operates in a dynamic manner in the way that it brings about the initialization of the MCU when it detects a dynamic rising edge of the V_{DD} voltage. The typical detected threshold is about 2 volts, but the actual value of the detected threshold depends on the way in which the V_{DD}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

Figure 13. Reset Circuit



RESET (Continued)

Figure 14. Reset & Interrupt Processing Flow-Chart

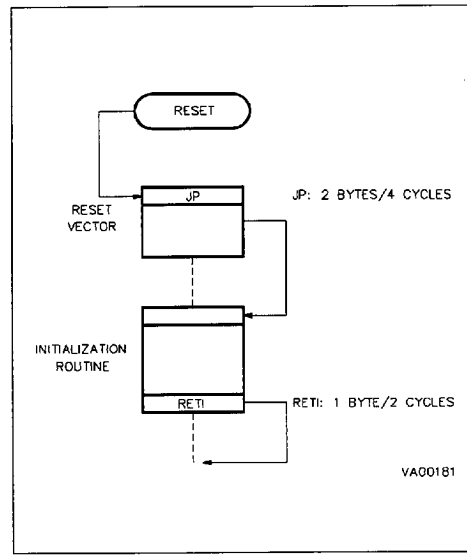


MCU Initialization Sequence

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 a NMI 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 15. Restart Initialization Program Flow-Chart



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.

WAIT & STOP MODES (Continued)

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.

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 ST6 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 ST6 core remains in the normal interrupt mode.

Note

To reach the lowest power consumption the user software must put 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.

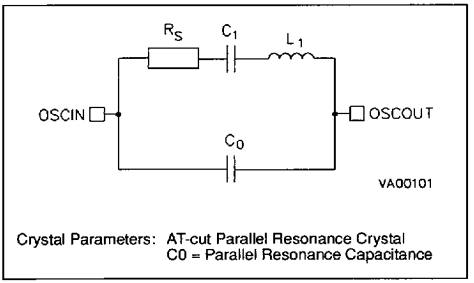
If all the interrupt sources are disabled (including NMI if GEN="0"), the restart of the MCU can only be done by a Reset activation. The Wait and Stop instructions are not executed if an enabled interrupt request is pending.

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 17.

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 RS), oscillator load capacitance (CL), IC parameters, ambient temperature, and 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 byte cycles to be executed.

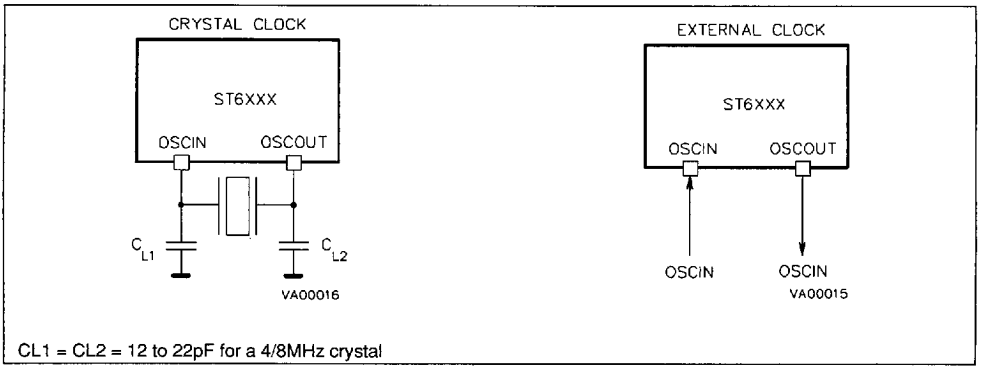
Figure 16. Crystal Parameters



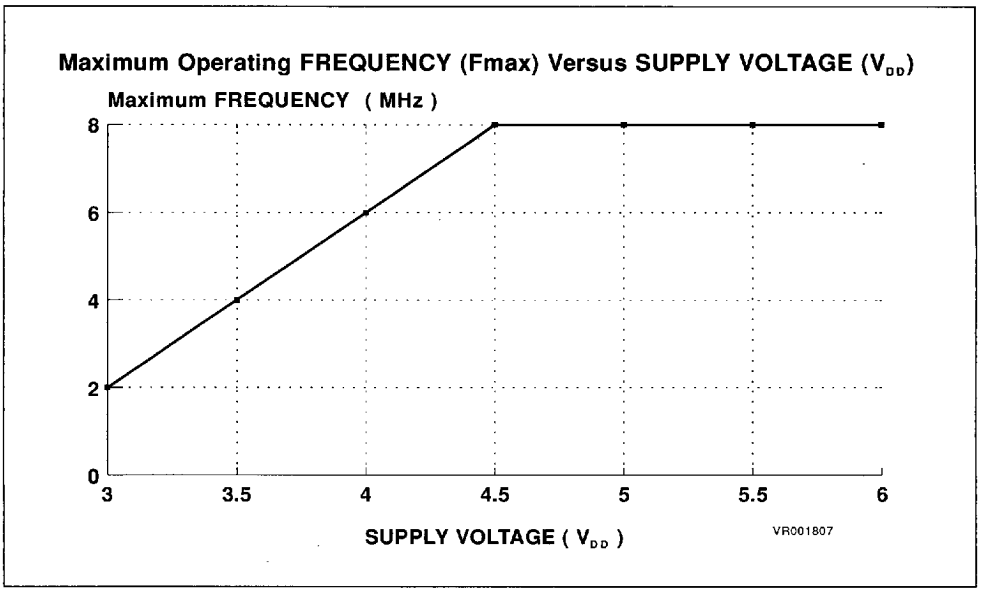
Crystal Parameters: AT-cut Parallel Resonance Crystal
C0 = Parallel Resonance Capacitance

ON-CHIP OSCILLATOR (Continued)

Figure 17. Oscillator Connection



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



INPUT/OUTPUT PORTS

The ST6210, ST6220 and ST6215, ST6225 microcontrollers have respectively 12 and 20 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 input
- Push-pull output
- Standard Open drain output
- 20mA Open drain output

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

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 three DATA registers (DRA, DRB, DRC), 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.

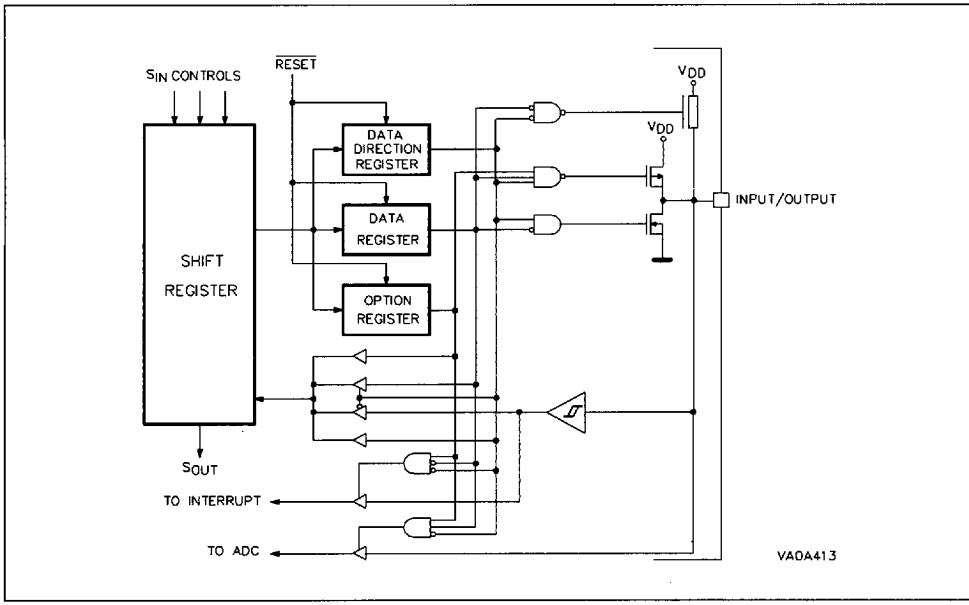
Single-bit operations on I/O registers are possible but care is necessary because reading in input mode is done from I/O pins while writing will directly affect the Port data register causing an undesired changes of the input configuration.

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

The three Option registers (ORPA, ORPB, ORPC) are used to select the different port options that are 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 18. I/O Port Block Diagram



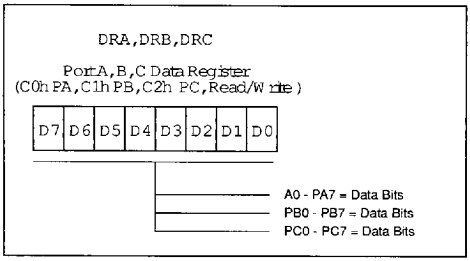
INPUT/OUTPUT PORTS (Continued)

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 6 shows all the port configurations that can be selected by user software.

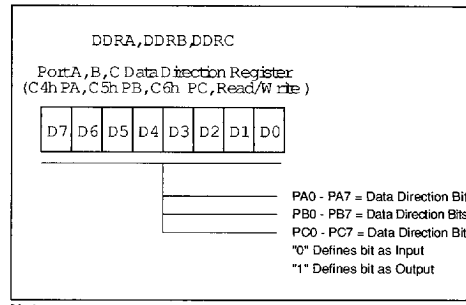
Figure 19. I/O Port Data Registers



Notes:

1. For complete coding explanation refer to Table 6.
2. PA4-PA7 and PC4-PC7 are not available on ST6210, ST6220/E20. PC0-PC3 are not available as pins.

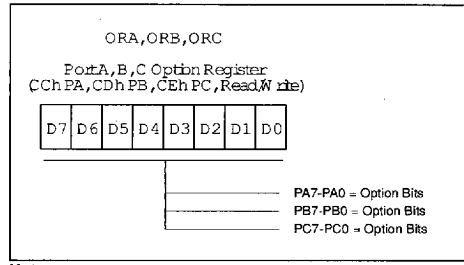
Figure 20. I/O Port Data Direction Registers



Notes:

1. For complete coding explanation refer to Table 6.
2. PA4-PA7 and PC4-PC7 are not available on ST6210, ST6220. PC0-PC3 are not available as pins. They should be programmed in output mode.

Figure 21. I/O Port Option Registers



Notes:

1. For complete coding explanation refer to Table 6.
2. PA4-PA7 and PC4-PC7 are not available on ST6210, ST6220. PC0-PC3 are not available as pins.

Table 6. 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 the PA0-PA3 pins).
			Input	Analog input (for the PA4-PA7, PB0-PB7, PC4-PC7 pin)
1	0	X	Output	20mA sink Open-drain output (for the PA0-PA3 pins)
			Output	Standard Open-drain output (for the PA4-PA7, PB0-PB7, PC4-PC7 pins)
1	1	X	Output	Push-pull output

Notes:

X. Means don't care.

1. PA4-PA7 and PC4-PC7 are not available on ST6210, ST6220.

INPUT/OUTPUT PORTS (Continued)

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 (see table 4). 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 (see table 4). The pins of Port A are AND-connected to the interrupt associated to the vector #1. The pins of Port B & C are AND-connected to the interrupt associated to the vector #2. The interrupt modes (falling edge sensitive, rising edge sensitive, low level sensitive) can be selected by software for each port by programming the IOR register.

Analog Input Option. The sixteen PA4-PA7, PB0-PB7, PC4-PC7 pins can be configured to be analog inputs according to the codes programmed in the OR and DR registers (see table 6). 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.

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 transistions 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 inut lines.

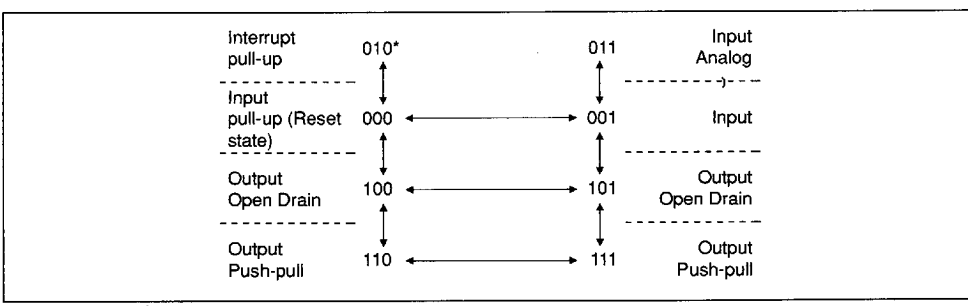
Single bit SET and RES instructions should be used very carefully with Port A, B and C data registers becaues these instructions make an implicit read and write back of the whole addressed register byte. In port input mode however data register address reads from input pins, not from data register latches 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 ST6210, ST6215, ST6220 and ST6225 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 22. I/O Port StateTransition Diagram for Safe Transitions



Note *. xxx = DDR, OR, DR Bits

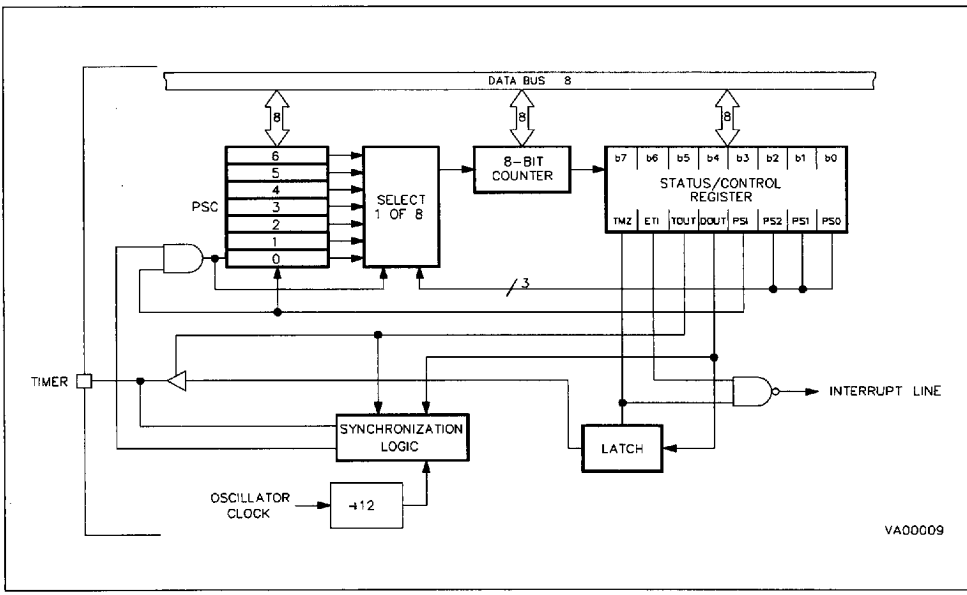
TIMER

The ST6210, ST6215, ST6220 and ST6225 offer one on-chip Timer peripheral consisting 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 23 shows the Timer block diagram. This timer 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 that can be addressed in the data space as a RAM location at address 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 (D4h address) 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 can be the oscillator frequency divided by 12 or an external clock at TIMER pin. The prescaler decrements on the rising edge. Depending on the division factor programmed by PS2, PS1 and PS0 bits in the TSCR (see table 6), 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 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 24 shows the Timer working principle.

Figure 23. Timer Peripheral Block Diagram



TIMER (Continued)

Timer Operating Modes

There are three 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 (TOSC/12 or TIMER pin signal) and to the output mode.

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".

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 when the processor is running but can be higher when the WAIT mode is entered (This is due to the need for synchronization with the core, this not being necessary during WAITing).

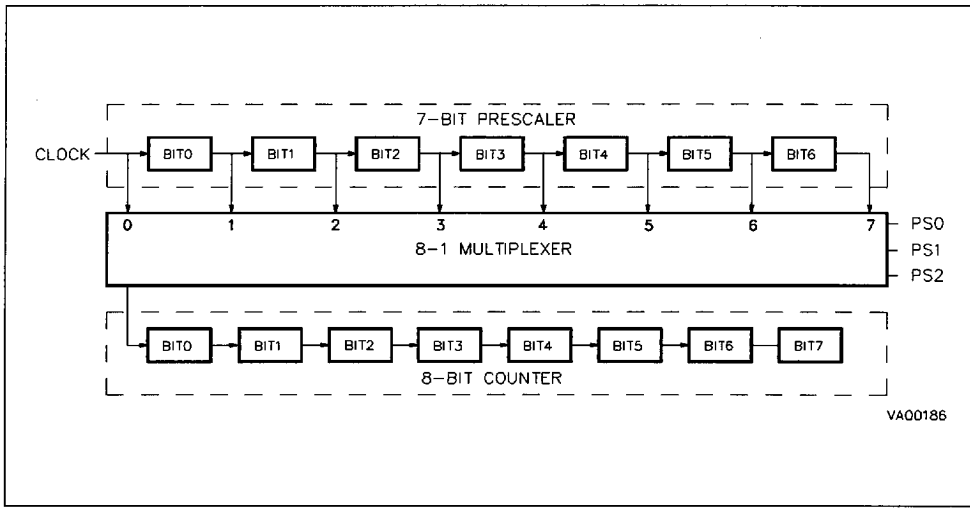
Output Mode (TOUT = "1", DOUT = data out). The TIMER pin is connected to the DOUT latch. Therefore the timer prescaler is clocked by the prescaler clock input (OSC/12).

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.

Table 7. Timer Operating Modes

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

Figure 24. Timer Working Principle



TIMER (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.

Notes

On ST6210,15, ST6220,25 the user can select the availability of an on-chip pull-up at TIMER pin as ROM mask option (see option list at the end of the datasheet).

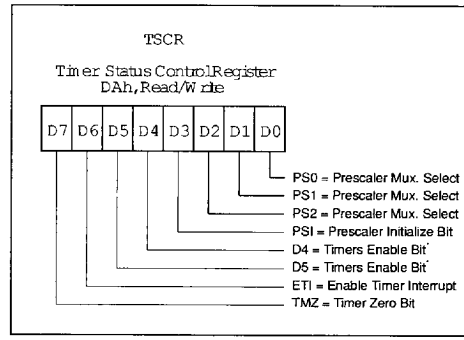
TMZ is set when the counter reaches 00h ; however, it may be set by writing 00h in the TCR register or setting bit 7 of the TSCR register. TMZ bit must be cleared by user software when servicing the timer interrupt to avoid undesired interrupts when leaving the interrupt service routine. After reset, 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.

Timer Registers

Figure 25. Timer Status Control Register



TMZ. Low-to-high transition indicates that the timer count register has decrement 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).

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 8. Prescaler Division Factors

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

TIMER (Continued)

Figure 26. Timer Counter Register

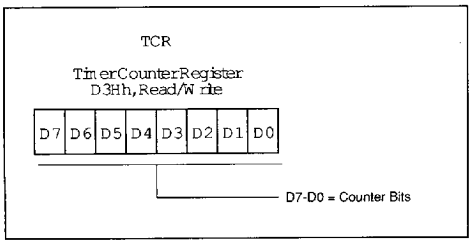
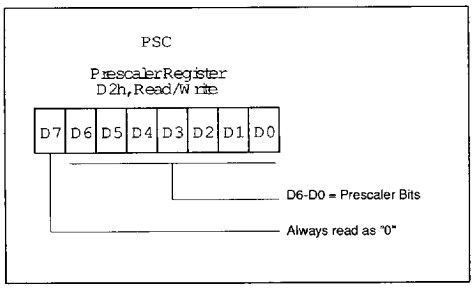


Figure 27. Prescaler Register



DIGITAL WATCHDOG

The digital Watchdog of the ST6210, ST6215, ST6220 and ST6225 devices consists of a down counter that can be used to provide a controlled recovery from a software upset.

On ST6210, ST6215, ST6220 and ST6225 the Watchdog activation (hardware or software) is user selectable; on masked devices the Watchdog activation can be selected as ROM option while for EPROM/OTP versions different part numbers are available (see ordering information at the end of the datasheet). If the hardware option is selected the Watchdog 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. In case of software option the Watchdog activation can be controlled by the user software so that the Low power mode (STOP, WAIT) may be used.

The Watchdog uses one data space register (DWDR location D8h). The Watchdog register is set to FEh on reset and immediately starts to count down, requiring no software start if the hardware option has been selected. The Watchdog time can be programmed using the 6 MSbits 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 check time can be set differently for different routines within the general program. 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 software option is selected the Low power enable option (Watchdog deactivated) there are 7 available counter bits for timer functions. This is because when the cell is used as Watchdog function, bit 1 of the register is used for managing the watchdog.

Note: Care must be taken when using the software Watchdog as a timer as the Watchdog bits are in reverse order.

If the Watchdog is active (either by hardware or software activation) the STOP instruction is deactivated 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.

DIGITAL WATCHDOG (Continued)

If the software option is selected, after a reset, the Watchdog/timer 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 (this is automatically done in the hardware activated option). 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 cannot be cleared by software without generating a Reset.

Note

In many applications the user may need to synchronize the RESET with the external circuitry and so when the watchdog initiates the ST6210, ST6215,E15 reset the external RESET pin will be held low until the on-chip reset circuit ensures the good start-up condition (see RESET description for additional information). This time is at least 50ns.

Figure 29. Watchdog Working Principle

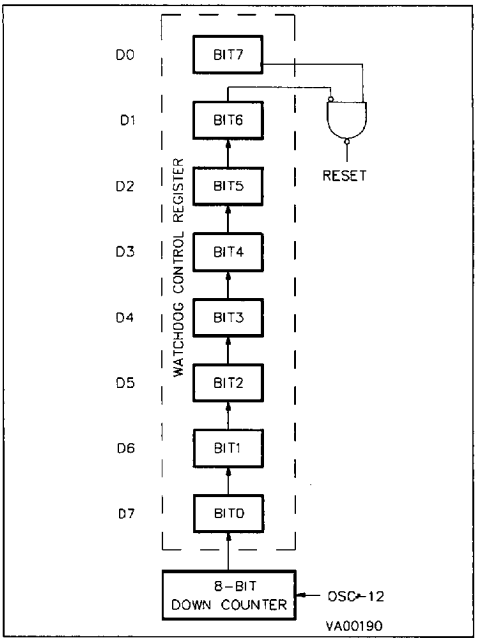
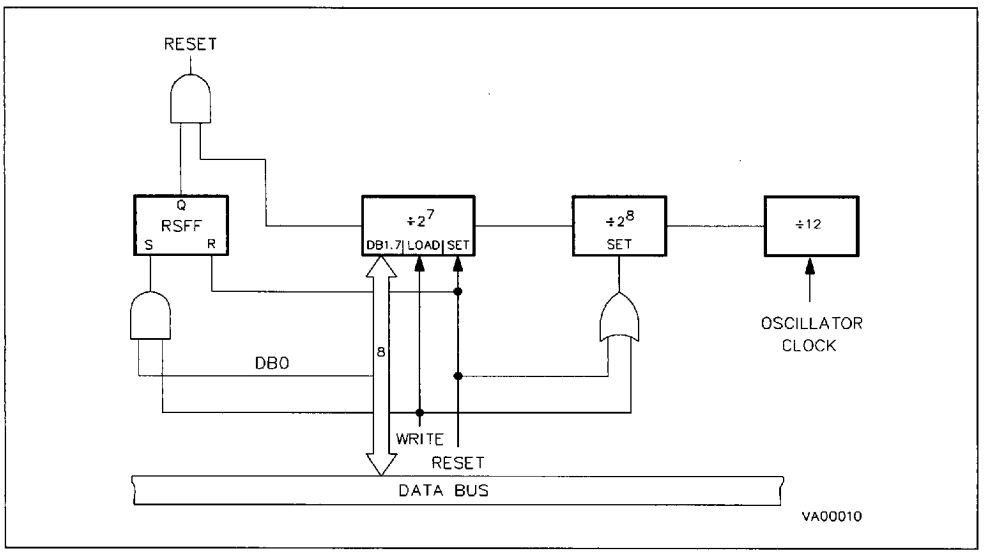
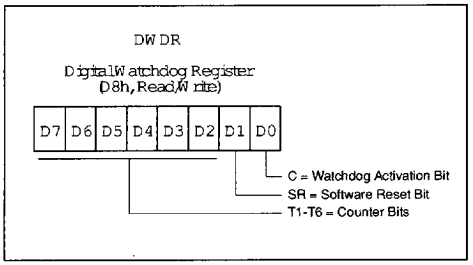


Figure 28. Digital Watchdog Block Diagram



DIGITAL WATCHDOG (Continued)

Figure 30. Watchdog Register



C. This is the Watchdog activation bit. This bit is hardware set to one if hardware option is selected and the user cannot change the value of this bit (the Watchdog is always active). When the software option is selected, if this bit is set to one the Watchdog function will be activated. When cleared to zero it allows the use of the counter as a 7-bit timer.

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 only) 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 Notes

The hardware activation option is very useful when the external circuitry may inject noises 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 ST6210, ST6215, ST6220 and ST6225 is an 8-bit analog to digital converter with 8 (PB0-PB7 on ST6210, ST6220) or 16 (PA4-PA7, PB0-PB7, PC4-PC7 on ST6215, ST6225) analog inputs (as alternate functions of I/O lines) offering 8-bit resolution with total accuracy ±2 LSB and a 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.

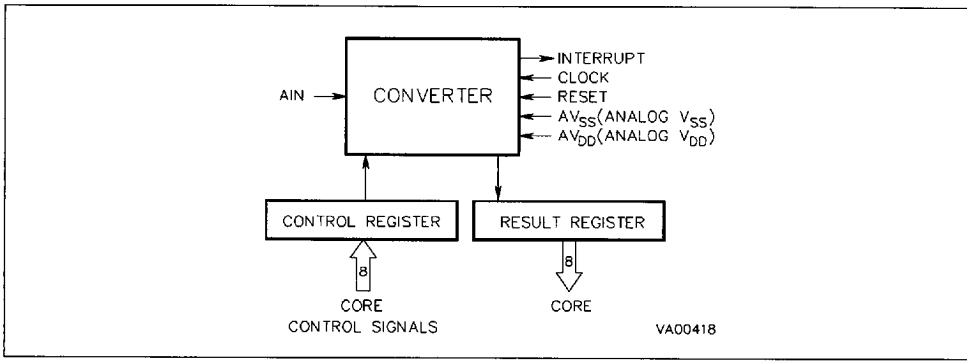
The selection of the pin signal that has to be converted is done by configuring the related I/O line as analog input through the I/O ports option and data registers (refer to I/O ports description for additional information). Only One I/O line must be configured as analog input at a time. The ADC uses 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.

Figure 31. A/D Converter Block Diagram



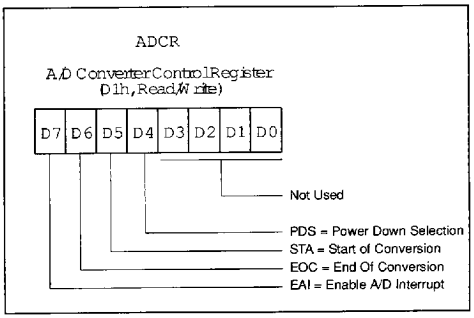
A/D CONVERTER (Continued)

To reduce the power consumption of the devices by turning off the ADC peripheral. The PDS bit in the ADC control register must be 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*

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 Registers

Figure 32. 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. 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.

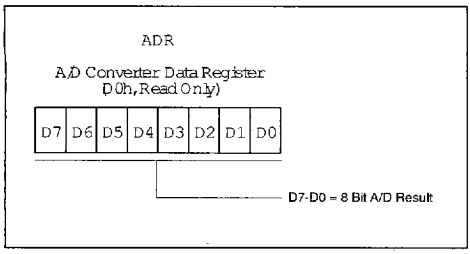
STA. Writing a '1' in this bit will start a conversion on the selected channel and automatically reset to zero the EOC bit. If the bit is set again when a conversion is in progress, the present conversion is stopped and a new one will take place. This bit is write only, any attempt to read it will show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

Figure 33. A/D Converter Data Register



D7-D0. 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 ST62 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).

It must be observed that the more accurate measurements are obtained on the pins PC4-PC7, but in all cases, no pin must be switched during the conversion to avoid any noise disturbance.

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. For instance, if V_{DD} = 3V, a 15mV resolution can be guaranteed.

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 output 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 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.

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 9. 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 10. 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 r	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. See Table 11.

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. See Table 12.

Control Instructions. The control instructions control the MCU operations during program execution. See Table 13.

Jump and Call. These two instructions are used to perform long (12-bit) jumps or subroutines call inside the whole program space. Refer to Table 14.

Table 11. 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 12. 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 13. 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 hardware activated watchdog function is selected.
Δ. Affected
*. Not Affected

Table 14. 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)

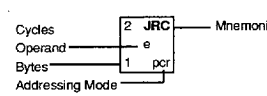
Opcode Map Summary. The following table contains an opcode map for the instructions used on the MCU.

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
rr1byte dataspace address
nn 1 byte immediate data
abc 12 bit address
ee 8 bit Displacement



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 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_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	PDIP28			60	°C/W
		PDIP20			55	
		PSO28			80	
		PSO20			75	

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		3.0		6.0	V
F_{OSC}	Oscillator Frequency	$4.5 < V_{DD} < 6.0V$ $V_{DD} = 3.5V$ $V_{DD} = 3.0V$	0.01 0.01 0.01		8.0 4.0 2.0	MHz
AV_{DD} AV_{SS}	Analog Supply Voltage(1)	$V_{SS} \leq AV_{SS} < AV_{DD} \leq V_{DD}$	V_{SS}		V_{DD}	V
I_{INJ+}	Pin Injection Current (positive) Digital Input Analog Inputs (2)	$V_{DD} = 4.5$ to 5.5V			+5	mA
I_{INJ-}	Pin Injection Current (negative) Digital Input Analog Inputs (3)	$V_{DD} = 4.5$ to 5.5V			-5	mA

Notes :

1. An oscillator frequency above 1MHz is recommended 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.
3. If a total current of +1 mA is flowing into the single analog channel or if the total current flowing into all the analog inputs is of 1mA, all the resulting conversions are shifted by +1 LSB. If a total positive current is flowing into the single analog channel or if the total current flowing into all the analog inputs is of 5mA, all the resulting conversions are shifted by +2 LSB.

DC ELECTRICAL CHARACTERISTICS

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

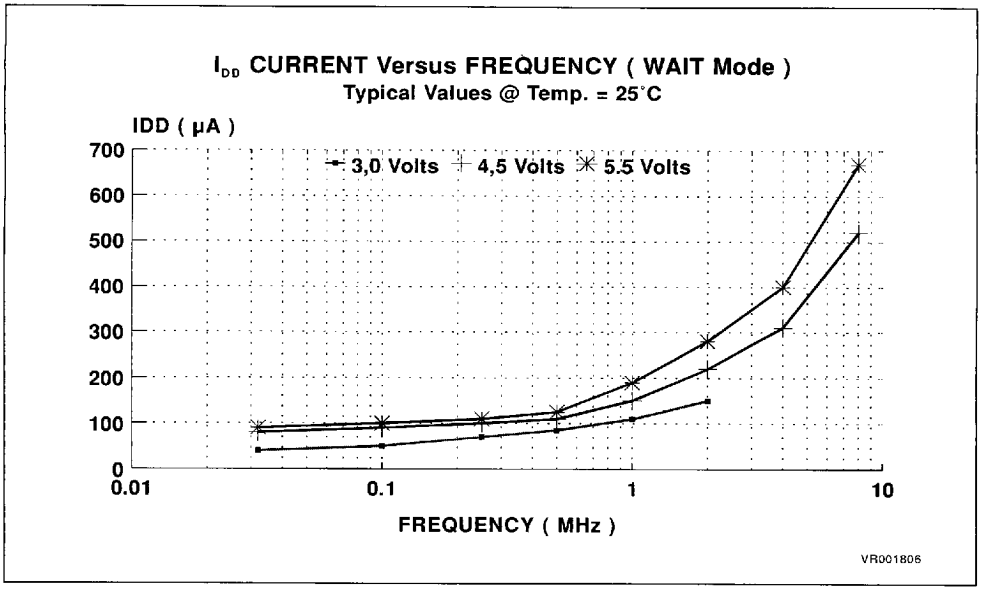
Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
V_{IL}	Input Low Level Voltage	RESET Pin V_{DD}=5V V_{DD}=3V			1.6 1	V
V_{IH}	Input High Level Voltage	RESET Pin V_{DD}=5V V_{DD}=3V	3.4 2			V
I_{IL} I_{IH}	Input Leakage Current	RESET Pin V_{IN}=V_{DD} (1) V_{IN}=V_{DD} (2) V_{IN}=V_{SS}	 -8	 -16	10 1 -30	µA mA µA
V_{IL}	Input Low Level Voltage	NMI,TIMER V_{DD}=5V V_{DD}=3V			0.3xV_{DD}	V
V_{IH}	Input High Level Voltage	NMI,TIMER V_{DD}=5V V_{DD}=3V	0.7xV_{DD}			V
V_{OL}	Low Level Output Voltage	TIMER, IOL=5.0mA V_{DD}=5V			0.2xV_{DD}	V
V_{OH}	High Level Output Voltage	TIMER, I_{OL}=-5.0mA V_{DD}=5V	0.65V_{DD}			V
I_{IL} I_{IH}	Input Leakage Current	TIMER V_{IN}=V_{DD} or V_{SS} V_{IN}=5.0V V_{IN}=3.9V		 0.1 0.1	 1.0 1.0	µA
I_{DD}	Supply Current in RESET Mode	V_{RESET}=V_{SS} f_{OSC}=8MHz			3.5	mA
	Supply Current in RUN Mode	f_{OSC}=8MHz I_{LOAD}=0mA V_{DD}=5.0V			3.5	mA
	Supply Current in WAIT Mode	f_{OSC}=8MHz (4) I_{LOAD}=0mA V_{DD}=5.0V			1.5	mA
	Supply Current in STOP Mode(3)	I_{LOAD}=0mA V_{DD}=5.0V			10	µA

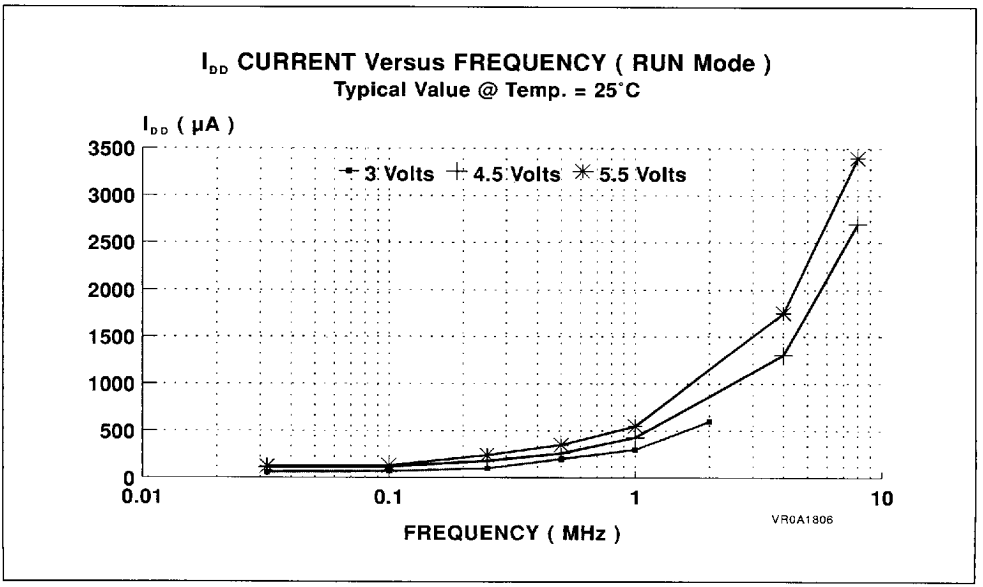
Notes :
1. No Watchdog Reset Actived.
2. Reset generated by Watchdog.
3. When the Watchdog function is activated the STOP instruction is deactivated. WAIT instruction is automatically executed.
4. Timer and A/D in OFF state.

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	V_{DD} = 3.0V V_{DD} = 4.5V V_{DD} = 5.5V	0.01 0.01 0.01		1 8 8	MHz
t_{ILH}	Interrupt Pin Maximum Pulse Widht	V_{DD} = 3.0V V_{DD} = 4.5V V_{DD} = 5.5V			no limit	µs
t_{SU}	Oscillator Start-up Time			5	10	ms
t_{SR}	Supply Rise Time		0.01		100	ms
t_{REC}	Supply Recovery Time		100			ms
T_{WNMI}	Minimum Pulse Width	NMI Pin V_{DD}=5V	100			ns
T_{WRES}	Minimum Pulse Width	RESET Pin	100			ns
C_{IN}	Input Capacitance	All Inputs Pins			10	pF
C_{OUT}	Output Capacitance	All outputs Pins			10	pF





I/O PORT CHARACTERISTICS

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
V_{IL}	Input Low Level Voltage	I/O Pins			0.3x V_{DD}	V
V_{IH}	Input High Level Voltage	I/O Pins	0.7x V_{DD}			V
V_{OL}	Low Level Output Voltage	V_{DD}= 5.0V I_{OL}= 10µA , All I/O Pins I_{OL}= 5mA , Standard I/O I_{OL}= 10mA , PA0-PA3 I_{OL}= 20mA , PA0-PA3			0.1 0.8 0.8 1.3	V
V_{OH}	High Level Output Voltage	I_{OH}= – 10µA I_{OH}= – 5mA, V_{DD}= 5.0V I_{OH}= – 1.5mA, V_{DD}= 3.0V	V_{DD}-0.1 3.5 2.0			V
I_{IL} I_{IH}	Input Leakage Current	Vin= V_{DD} or V_{SS} V_{DD}= 3.0V V_{DD}= 5.5V		0.1 0.1	1.0 1.0	µA
R_{PU}	Pull-up Resistor	Vin= 0V	50	100	200	KΩ

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	V_{DD} = 3.0V V_{DD} = 4.5V			$\frac{f_{OSC}}{8}$	MHz
t_W	Pulse Width at TIMER Pin	V_{DD} = 3.0V V_{DD} = 4.5V V_{DD} = 5.5V	1 125 125			µs ns ns

A/D CONVERTER CHARACTERISTICS

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

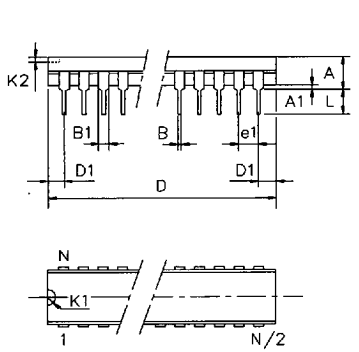
Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
Res	Resolution			8		Bit
A_{TOT}	Total Accuracy	f_{OSC} > 1.2MHz f_{OSC} > 32kHz			2 4	LSB
t_C (1)	Conversion Time	f_{OSC} = 8MHz		70		µs
V_{AN}	Conversion Range		AV_{SS}		AV_{DD}	V
ZIR	Zero Input Reading	Conversion result when Vin = AV_{SS}	00			Hex
FSR	Full Scale Reading	Conversion result when Vin = AV_{DD}			FF	Hex
AV_{SS} (2) AV_{DD}	Analog Reference		V_{SS}		V_{DD}	V
AD_I	Analog Input Current During Conversion	V_{DD}= 4.5V			1.0	µA
AC_{IN} (3)	Analog Input Capacitance				2	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. It is recommended not to use the A/D with fOSC < 1Mhz.
2. In ST6210, ST6215, ST6220 and ST6225 AV_{SS} and AV_{DD} are internally connected to digital V_{SS} and V_{DD}.
3. Excluding Pad Capacitance.

PACKAGE MECHANICAL DATA

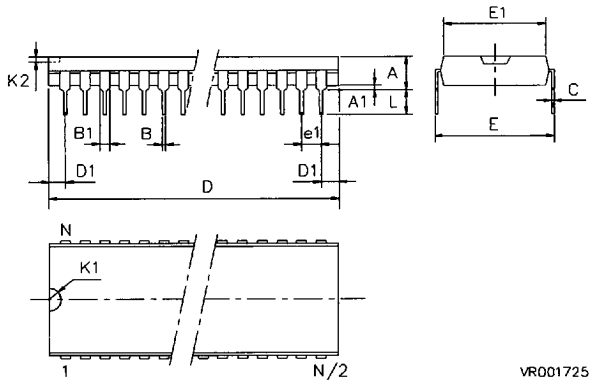
Figure 34. 20-Pin Dual in Line Plastic (B), 300-Mil Width



VR0A1725

Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A			3.93			0.155
A1	0.254			0.010		
B			0.45			0.017
B1	1.39		1.65	0.055		0.065
C			0.25			0.009
D			25.4			1.000
D1			1.34			0.053
E		8.5			0.334	
E1			7.1			0.279
K1	–	–	–	–	–	–
K2	–	–	–	–	–	–
L		3.30			0.129	
e1		2.54			0.10	
Number of Pins						
N	20					

Figure 35. 28-Pin Dual in Line Plastic (B), 600-Mil Width

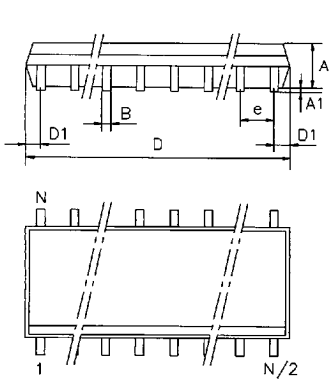


VR001725

Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A	2.2		4.8	0.086		0.189
A1	0.51		1.77	0.010		0.069
B	0.38		0.58	0.015		0.023
B1	0.97		1.52	0.055		0.065
C	0.2		0.3	0.008		0.009
D	35.06		36.22	1.400		1.425
D1	–	–	–	–	–	–
E			16.3			0.641
E1		12.9				0.508
K1	–	–	–	–	–	–
K2	–	–	–	–	–	–
L	3.18		4.44	1.25		0.174
e1		2.54			0.10	
Number of Pins						
N	28					

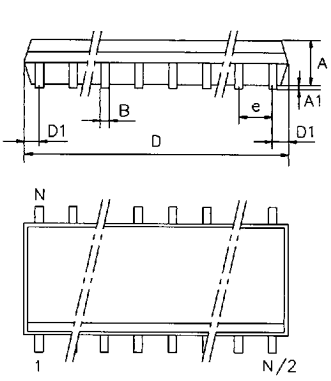
PACKAGES MECHANICAL DATA (Continued)

Figure 36. 20-Lead Small Outline Plastic (M), 300-Mil Width



Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A	2.05		2.55	0.081		0.10
A1	0.1		0.3	0.004		.0118
B	0.35		0.49	.0138		.0192
C	0.23		0.32	.0091		.0125
D	12.6		13.0	.4961		.5118
D1	–	–	–	–	–	–
E	10.0		10.65	0.394		0.419
E1	7.40		7.60	.2914		.2992
E2	–	–	–	–	–	–
L	0.40		1.27	0.016		0.050
e		1.27			0.05	
α	–	–	–	–	–	–
Number of Pins						
N	20					

Figure 37. 28-Lead Small Outline Plastic (M), 300-Mil Width



Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A	2.05		2.55	0.081		0.10
A1	0.1		0.3	0.004		.0118
B	0.35		0.49	.0138		.0192
C	0.23		0.32	.0091		.0125
D	17.7		18.1	.6969		.7125
D1	–	–	–	–	–	–
E	10.0		10.65	0.394		0.419
E1	7.40		7.60	.2914		.2992
E2	–	–	–	–	–	–
L	0.40		1.27	0.016		0.050
e		1.27			0.05	
α	–	–	–	–	–	–
Number of Pins						
N	28					

ORDERING INFORMATION

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

Communication of the ROM content. To communicate the contents of Program/Data ROM memories to SGS-THOMSON, the customer has to send one diskette with the hexadecimal file generated by the development tool. All unused byte must be set to FFh.

Listing Generation & Verification. When SGS-THOMSON receives the diskette, a computer listing is generated from it. 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 listing 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.

Table 15. ROM Memory Map

ST6210,ST6215 (2K ROM Devices)

Device Address	Description
0000h-087Fh	Reserved (1)
0880h-0F9Fh	User ROM
0FA0h-0FEFh	Reserved (1)
0FF0h-0FF7h	Interrupt Vectors
0FF8h-0FFBh	Reserved (1)
0FFCh-0FFDh	NMI Interrupt Vector
0FFEh-0FFFh	Reset Vector

ST6220,ST6225 (4K ROM Devices)

Device Address	Description
0000h-007Fh	Reserved (1)
0080h-0F9Fh	User ROM
0FA0h-0FEFh	Reserved (1)
0FF0h-0FF7h	Interrupt Vectors
0FF8h-0FFBh	Reserved (1)
0FFCh-0FFDh	NMI Interrupt Vector
0FFEh-0FFFh	Reset Vector

Notes :
1. Reserved Areas should be filled with FFh

ORDERING INFORMATION TABLE

Sales Type	ROM x8	I/O	Additional Features	Temperature Range	Package
ST6210B1 ST6210B6	2K Bytes	12	A/D CONVERTER	0 to +70°C -40 to +85°C	PDIP20
ST6210M1 ST6210M6				0 to +70°C -40 to +85°C	PSO20
ST6215B1 ST6215B6		20	A/D CONVERTER	0 to +70°C -40 to +85°C	PDIP28
ST6215M1 ST6215M6				0 to +70°C -40 to +85°C	PSO28
ST6220B1 ST6220B6	4K Bytes	12	A/D CONVERTER	0 to +70°C -40 to +85°C	PDIP20
ST6220M1 ST6220M6				0 to +70°C -40 to +85°C	PSO20
ST6225B1 ST6225B6		20	A/D CONVERTER	0 to +70°C -40 to +85°C	PDIP28
ST6225M1 ST6225M6				0 to +70°C -40 to +85°C	PSO28

Note: Each ROM content is identifical by 2 alphabetic characters to be added to the sales type.

ST6210, ST6215, ST6220, ST6225 MICROCONTROLLER OPTION LIST

Customer . . . . . .
Address . . . . . .
. . . . . .
Contact . . . . . .
Phone No . . . . . .
Reference . . . . . .

SGS-THOMSON Microelectronics references

Device:
[ ] ST6210, [ ] ST6215, [ ] ST6220, [ ] ST6225

Package:
[ ] Dual in Line Plastic [ ] Small Outline Plastic

Temperature Range:
[ ] 0°C to + 70°C [ ] – 40°C to + 85°C

Special Marking:
[ ] No
[ ] Yes "_ _ _ _ _ _ _ _ _ _ _ _ "
Authorized characters are letters, digits, '.', '–', '/' and spaces only.
Maximum character count are 10 char. for DIP packages and 8 char. for SO packages.

Input pull-up selection on NMI pin : [ ] Yes [ ] No
Input pull-up selection on TIMER pin : [ ] Yes [ ] No

Watchdog Selection:
[ ] Hardware Activation (no STOP mode) [ ] Software Activation (STOP mode available)

Notes . . . . . .
. . . . . .

Signature . . . . . .

Date . . . . . .