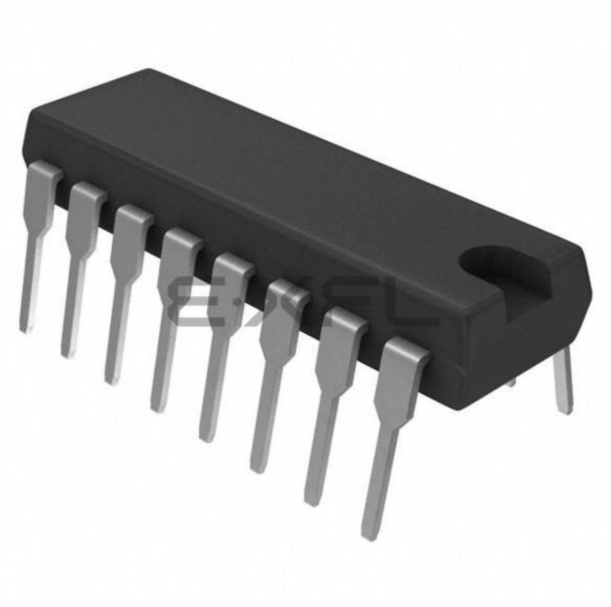


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	ST6
Core Size	8-Bit
Speed	8MHz
Connectivity	-
Peripherals	LVD, POR, WDT
Number of I/O	9
Program Memory Size	1.8KB (1.8K x 8)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	128 x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 6V
Data Converters	A/D 4x8b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Through Hole
Package / Case	16-DIP (0.300", 7.62mm)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/st62t52cb6

Table of Contents

Document Page

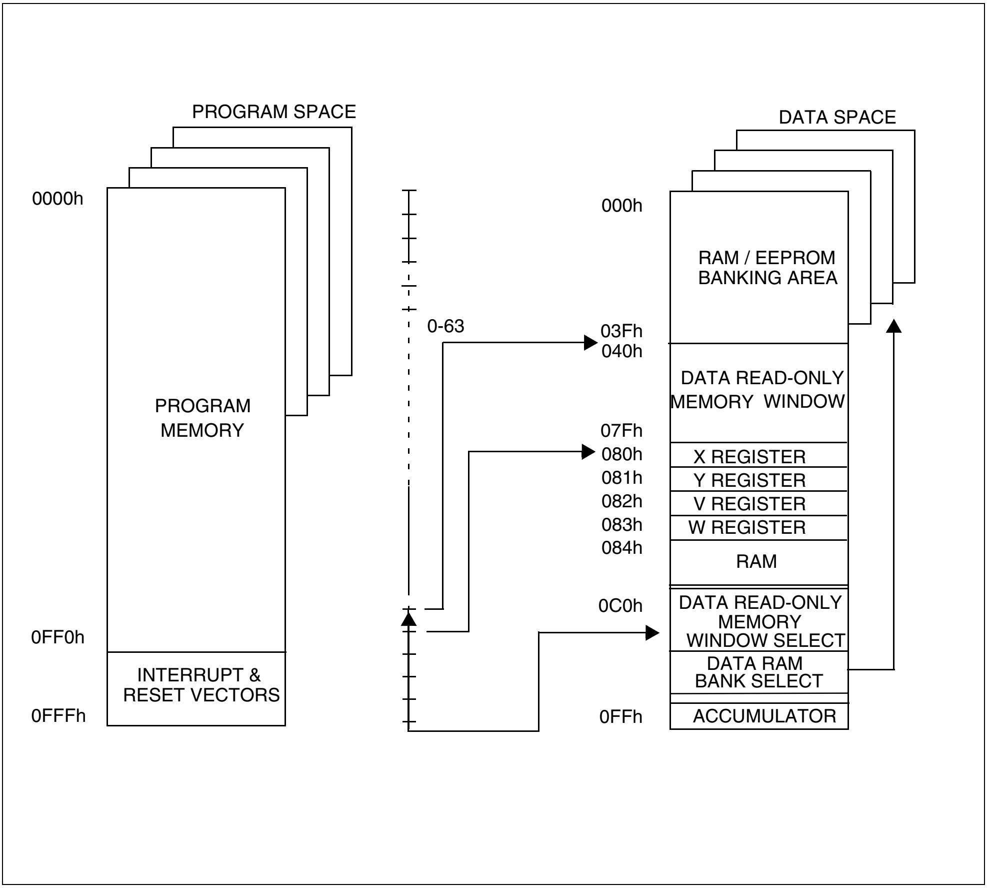
1.3 MEMORY MAP

1.3.1 Introduction

The MCU operates in three separate memory spaces: Program space, Data space, and Stack space. Operation in these three memory spaces is described in the following paragraphs.

Briefly, Program space contains user program code in OTP and user vectors; Data space contains user data in RAM and in OTP, and Stack space accommodates six levels of stack for subroutine and interrupt service routine nesting.

Figure 3. Memory Addressing Diagram



MEMORY MAP (Cont'd)

Additional Notes on Parallel Mode:

If the user wishes to perform parallel programming, the first step should be to set the E2PAR2 bit. From this time on, the EEPROM will be addressed in write mode, the ROW address and the data will be latched and it will be possible to change them only at the end of the programming cycle or by resetting E2PAR2 without programming the EEPROM. After the ROW address is latched, the MCU can only "see" the selected EEPROM row and any attempt to write or read other rows will produce errors.

The EEPROM should not be read while E2PAR2 is set.

As soon as the E2PAR2 bit is set, the 8 volatile ROW latches are cleared. From this moment on, the user can load data in all or in part of the ROW. Setting E2PAR1 will modify the EEPROM registers corresponding to the ROW latches accessed after E2PAR2. For example, if the software sets E2PAR2 and accesses the EEPROM by writing to addresses 18h, 1Ah and 1Bh, and then sets E2PAR1, these three registers will be modified simultaneously; the remaining bytes in the row will be unaffected.

Note that E2PAR2 is internally reset at the end of the programming cycle. This implies that the user must set the E2PAR2 bit between two parallel programming cycles. Note that if the user tries to set E2PAR1 while E2PAR2 is not set, there will be no programming cycle and the E2PAR1 bit will be unaffected. Consequently, the E2PAR1 bit cannot be set if E2ENA is low. The E2PAR1 bit can be set by the user, only if the E2ENA and E2PAR2 bits are also set.

Notes: The EEPROM page shall not be changed through the DRBR register when the E2PAR2 bit is set.

EEPROM Control Register (EECTL)

Address: EAh — Read/Write

Reset status: 00h

7							0
D7	E2OFF	D5	D4	E2PAR1	E2PAR2	E2BUSY	E2ENA

Bit 7 = **D7**: *Unused.*

Bit 6 = **E2OFF**: *Stand-by Enable Bit.* WRITE ONLY. If this bit is set the EEPROM is disabled (any access will be meaningless) and the power consumption of the EEPROM is reduced to its lowest value.

Bit 5-4 = **D5-D4**: *Reserved.* MUST be kept reset.

Bit 3 = **E2PAR1**: *Parallel Start Bit.* WRITE ONLY. Once in Parallel Mode, as soon as the user software sets the E2PAR1 bit, parallel writing of the 8 adjacent registers will start. This bit is internally reset at the end of the programming procedure. Note that less than 8 bytes can be written if required, the undefined bytes being unaffected by the parallel programming cycle; this is explained in greater detail in the Additional Notes on Parallel Mode overleaf.

Bit 2 = **E2PAR2**: *Parallel Mode En. Bit.* WRITE ONLY. This bit must be set by the user program in order to perform parallel programming. If E2PAR2 is set and the parallel start bit (E2PAR1) is reset, up to 8 adjacent bytes can be written simultaneously. These 8 adjacent bytes are considered as a row, whose address lines A7, A6, A5, A4, A3 are fixed while A2, A1 and A0 are the changing bits, as illustrated in Table 4. E2PAR2 is automatically reset at the end of any parallel programming procedure. It can be reset by the user software before starting the programming procedure, thus leaving the EEPROM registers unchanged.

Bit 1 = **E2BUSY**: *EEPROM Busy Bit.* READ ONLY. This bit is automatically set by the EEPROM control logic when the EEPROM is in programming mode. The user program should test it before any EEPROM read or write operation; any attempt to access the EEPROM while the busy bit is set will be aborted and the writing procedure in progress will be completed.

Bit 0 = **E2ENA**: *EEPROM Enable Bit.* WRITE ONLY. This bit enables programming of the EEPROM cells. It must be set before any write to the EEPROM register. Any attempt to write to the EEPROM when E2ENA is low is meaningless and will not trigger a write cycle.

CLOCK SYSTEM (Cont'd)

Turning on the main oscillator is achieved by resetting the OSCOFF bit of the A/D Converter Control Register or by resetting the MCU. Restarting the main oscillator implies a delay comprising the oscillator start up delay period plus the duration of the software instruction at f_{LFAO} clock frequency.

3.1.2 Low Frequency Auxiliary Oscillator (LFAO)

The Low Frequency Auxiliary Oscillator has three main purposes. Firstly, it can be used to reduce power consumption in non timing critical routines. Secondly, it offers a fully integrated system clock, without any external components. Lastly, it acts as a safety oscillator in case of main oscillator failure.

This oscillator is available when the OSG ENABLED option is selected. In this case, it automatically starts one of its periods after the first missing edge from the main oscillator, whatever the reason (main oscillator defective, no clock circuitry provided, main oscillator switched off...).

User code, normal interrupts, WAIT and STOP instructions, are processed as normal, at the reduced f_{LFAO} frequency. The A/D converter accuracy is decreased, since the internal frequency is below 1MHz.

At power on, the Low Frequency Auxiliary Oscillator starts faster than the Main Oscillator. It therefore feeds the on-chip counter generating the POR delay until the Main Oscillator runs.

The Low Frequency Auxiliary Oscillator is automatically switched off as soon as the main oscillator starts.

ADCR

Address: 0D1h — Read/Write

7							0
ADCR 7	ADCR 6	ADCR 5	ADCR 4	ADCR 3	OSC OFF	ADCR 1	ADCR 0

Bit 7-3, 1-0= **ADCR7-ADCR3, ADCR1-ADCR0**: *ADC Control Register*. These bits are not used.

Bit 2 = **OSCOFF**. When low, this bit enables main oscillator to run. The main oscillator is switched off when OSCOFF is high.

3.1.3 Oscillator Safe Guard

The Oscillator Safe Guard (OSG) affords drastically increased operational integrity in ST62xx devices. The OSG circuit provides three basic functions: it filters spikes from the oscillator lines which would result in over frequency to the ST62 CPU; it gives access to the Low Frequency Auxiliary Oscillator (LFAO), used to ensure minimum processing in case of main oscillator failure, to offer reduced power consumption or to provide a fixed frequency low cost oscillator; finally, it automatically limits the internal clock frequency as a function of supply voltage, in order to ensure correct operation even if the power supply should drop.

The OSG is enabled or disabled by choosing the relevant OSG option. It may be viewed as a filter whose cross-over frequency is device dependent.

Spikes on the oscillator lines result in an effectively increased internal clock frequency. In the absence of an OSG circuit, this may lead to an over frequency for a given power supply voltage. The OSG filters out such spikes (as illustrated in Figure 9.). In all cases, when the OSG is active, the maximum internal clock frequency, f_{INT}, is limited to f_{OSG}, which is supply voltage dependent. This relationship is illustrated in Figure 12..

When the OSG is enabled, the Low Frequency Auxiliary Oscillator may be accessed. This oscillator starts operating after the first missing edge of the main oscillator (see Figure 10.).

Over-frequency, at a given power supply level, is seen by the OSG as spikes; it therefore filters out some cycles in order that the internal clock frequency of the device is kept within the range the particular device can stand (depending on V_{DD}), and below f_{OSG}: the maximum authorised frequency with OSG enabled.

Note. The OSG should be used wherever possible as it provides maximum safety. Care must be taken, however, as it can increase power consumption and reduce the maximum operating frequency to f_{OSG}.

Warning: Care has to be taken when using the OSG, as the internal frequency is defined between a minimum and a maximum value and is not accurate.

For precise timing measurements, it is not recommended to use the OSG and it should not be enabled in applications that use the SPI or the UART.

It should also be noted that power consumption in Stop mode is higher when the OSG is enabled (around 50µA at nominal conditions and room temperature).

INTERRUPTS (Cont'd)

3.5.3 Interrupt Option Register (IOR)

The Interrupt Option Register (IOR) is used to enable/disable the individual interrupt sources and to select the operating mode of the external interrupt inputs. This register is write-only and cannot be accessed by single-bit operations.

Address: 0C8h — Write Only

Reset status: 00h

7							0
-	LES	ESB	GEN	-	-	-	-

Bit 7, Bits 3-0 = *Unused*.

Bit 6 = **LES**: *Level/Edge Selection bit*.

When this bit is set to one, the interrupt source #1 is level sensitive. When cleared to zero the edge sensitive mode for interrupt request is selected.

Bit 5 = **ESB**: *Edge Selection bit*.

The bit ESB selects the polarity of the interrupt source #2.

Bit 4 = **GEN**: *Global Enable Interrupt*. When this bit is set to one, all interrupts are enabled. When this bit is cleared to zero all the interrupts (excluding NMI) are disabled.

When the GEN bit is low, the NMI interrupt is active but cannot cause a wake up from STOP/WAIT modes.

This register is cleared on reset.

3.5.4 Interrupt Sources

Interrupt sources available on the ST62E62C/T62C are summarized in the Table 10 with associated mask bit to enable/disable the interrupt request.

Table 10. Interrupt Requests and Mask Bits

Peripheral	Register	Address Register	Mask bit	Masked Interrupt Source	Interrupt vector
GENERAL	IOR	C8h	GEN	All Interrupts, excluding NMI	
TIMER	TSCR1	D4h	ETI	TMZ: TIMER Overflow	Vector 4
A/D CONVERTER	ADCR	D1h	EAI	EOC: End of Conversion	Vector 4
AR TIMER	ARMC	D5h	OVIE CPIE EIE	OVF: AR TIMER Overflow CPF: Successful compare EF: Active edge on ARTIMin	Vector 3
Port PAn	ORPA-DRPA	C0h-C4h	ORPAn-DRPAn	PAn pin	Vector 1
Port PBn	ORPB-DRPB	C1h-C5h	ORPBn-DRPBn	PBn pin	Vector 1
Port PCn	ORPC-DRPC	C2h-C6h	ORPCn-DRPCn	PCn pin	Vector 2

3.6 POWER SAVING MODES

The WAIT and STOP modes have been implemented in the ST62xx family of MCUs in order to reduce the product's electrical consumption during idle periods. These two power saving modes are described in the following paragraphs.

3.6.1 WAIT Mode

The MCU goes into WAIT mode as soon as the WAIT instruction is executed. The microcontroller can be considered as being in a "software frozen" state where the core stops processing the program instructions, the RAM contents and peripheral registers are preserved as long as the power supply voltage is higher than the RAM retention voltage. In this mode the peripherals are still active.

WAIT mode can be used when the user wants to reduce the MCU power consumption during idle periods, while not losing track of time or the capability of monitoring external events. The active oscillator is not stopped in order to provide a clock signal to the peripherals. Timer counting may be enabled as well as the Timer interrupt, before entering the WAIT mode: this allows the WAIT mode to be exited when a Timer interrupt occurs. The same applies to other peripherals which use the clock signal.

If the WAIT mode is exited due to a Reset (either by activating the external pin or generated by the Watchdog), the MCU enters a normal reset procedure. If an interrupt is generated during WAIT mode, the MCU's behaviour depends on the state of the processor core prior to the WAIT instruction, but also on the kind of interrupt request which is generated. This is described in the following paragraphs. The processor core does not generate a delay following the occurrence of the interrupt, because the oscillator clock is still available and no stabilisation period is necessary.

3.6.2 STOP Mode

If the Watchdog is disabled, 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 RAM contents and peripheral registers are preserved 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 a Reset to exit the STOP state.

If the STOP state is exited due to a Reset (by activating the external pin) the MCU will enter a normal reset procedure. Behaviour in response to interrupts depends on the state of the processor core prior to issuing the STOP instruction, and also on the kind of interrupt request that is generated.

This case will be described in the following paragraphs. The processor core generates a delay after occurrence of the interrupt request, in order to wait for complete stabilisation of the oscillator, before executing the first instruction.

POWER SAVING MODE (Cont'd)

3.6.3 Exit from WAIT and STOP Modes

The following paragraphs describe how the MCU exits from WAIT and STOP modes, when an interrupt occurs (not a Reset). It should be noted that the restart sequence depends on the original state of the MCU (normal, interrupt or non-maskable interrupt mode) prior to entering WAIT or STOP mode, as well as on the interrupt type.

Interrupts do not affect the oscillator selection.

3.6.3.1 Normal Mode

If the MCU was in the main routine when the WAIT or STOP instruction was executed, exit from Stop or Wait mode will occur as soon as an interrupt occurs; the related interrupt routine is executed and, on completion, the instruction which follows the STOP or WAIT instruction is then executed, providing no other interrupts are pending.

3.6.3.2 Non Maskable Interrupt Mode

If the STOP or WAIT instruction has been executed during execution of the non-maskable interrupt routine, the MCU exits from the Stop or Wait mode as soon as an interrupt occurs: the instruction which follows the STOP or WAIT instruction is executed, and the MCU remains in non-maskable interrupt mode, even if another interrupt has been generated.

3.6.3.3 Normal Interrupt Mode

If the MCU was in interrupt mode before the STOP or WAIT instruction was executed, it exits from STOP or WAIT mode as soon as an interrupt occurs. Nevertheless, two cases must be considered:

– If the interrupt is a normal one, the interrupt routine in which the WAIT or STOP mode was entered will be completed, starting with the execution of the instruction which follows the STOP or the WAIT instruction, and the MCU is still in the interrupt mode. At the end of this routine pending interrupts will be serviced in accordance with their priority.

– In the event of a non-maskable interrupt, the non-maskable interrupt service routine is processed first, then the routine in which the WAIT or STOP mode was entered will be completed by executing the instruction following the STOP or WAIT instruction. The MCU remains in normal interrupt mode.

Notes:

To achieve the lowest power consumption during RUN or WAIT modes, the user program must take care of:

– configuring unused I/Os as inputs without pull-up (these should be externally tied to well defined logic levels);

– placing all peripherals in their power down modes before entering STOP mode;

When the hardware activated Watchdog is selected, or when the software Watchdog is enabled, the STOP instruction is disabled and a WAIT instruction will be executed in its place.

If all interrupt sources are disabled (GEN low), the MCU can only be restarted by a Reset. Although setting GEN low does not mask the NMI as an interrupt, it will stop it generating a wake-up signal.

The WAIT and STOP instructions are not executed if an enabled interrupt request is pending.

4 ON-CHIP PERIPHERALS

4.1 I/O PORTS

The MCU features Input/Output lines which may be individually programmed as any of the following input or output configurations:

– Input without pull-up or interrupt

– Input with pull-up and interrupt

– Input with pull-up, but without interrupt

– Analog input

– Push-pull output

– Open drain output

The lines are organised as bytewise Ports.

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

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

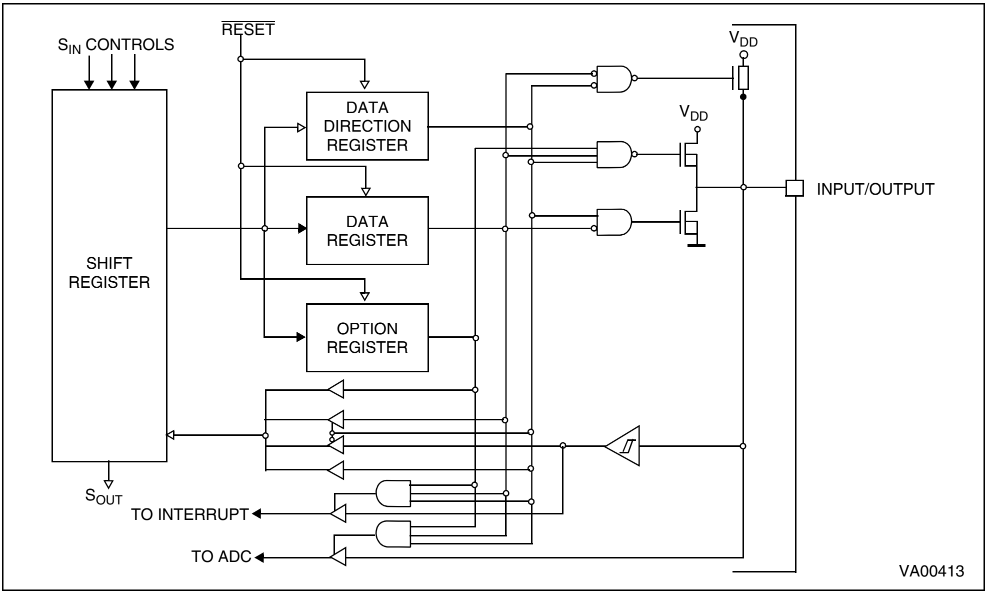
Single-bit operations on I/O registers are possible but care is necessary because reading in input mode is done from I/O pins while writing will directly affect the Port data register causing an undesired change of the input configuration.

The Data Direction registers (DDRx) allow the data direction (input or output) of each pin to be set.

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

All I/O registers can be read or written to just as any other RAM location in Data space, so no extra RAM cells are needed for port data storage and manipulation. During MCU initialization, all I/O registers are cleared and the input mode with pull-ups and no interrupt generation is selected for all the pins, thus avoiding pin conflicts.

Figure 22. I/O Port Block Diagram



I/O PORTS (Cont'd)

4.1.2 Safe I/O State Switching Sequence

Switching the I/O ports from one state to another should be done in a sequence which ensures that no unwanted side effects can occur. The recommended safe transitions are illustrated in Figure 23.. All other transitions are potentially risky and should be avoided when changing the I/O operating mode, as it is most likely that undesirable side-effects will be experienced, such as spurious interrupt generation or two pins shorted together by the analog multiplexer.

Single bit instructions (SET, RES, INC and DEC) should be used with great caution on Ports Data registers, since these instructions make an implicit read and write back of the entire register. In port input mode, however, the data register reads from the input pins directly, and not from the data register latches. Since data register information in input mode is used to set the characteristics of the input pin (interrupt, pull-up, analog input), these may be unintentionally reprogrammed depending on the state of the input pins. As a general rule, it is better to limit the use of single bit instructions on data registers to when the whole (8-bit) port is in output mode. In the case of inputs or of mixed inputs and outputs, it is advisable to keep a copy of the data register in RAM. Single bit instructions may then be used on the RAM copy, after which the whole copy register can be written to the port data register:

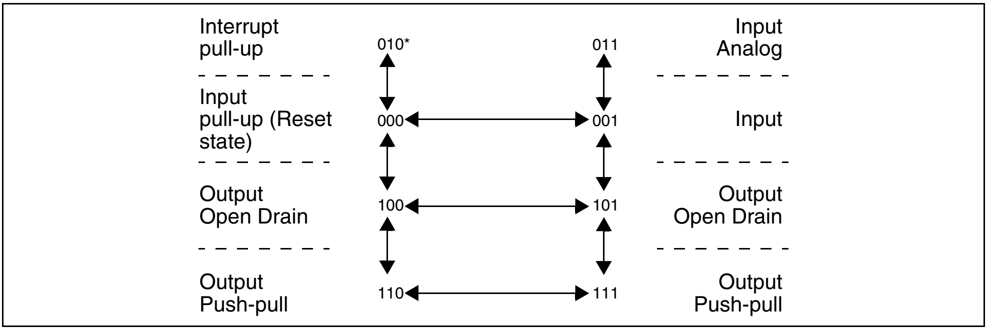
```
SET bit, datacopy
LD a, datacopy
LD DRA, a
```

Warning: Care must also be taken to not use instructions that act on a whole port register (INC, DEC, or read operations) when all 8 bits are not available on the device. Unavailable bits must be masked by software (AND instruction).

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

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

Figure 23. Diagram showing Safe I/O State Transitions



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

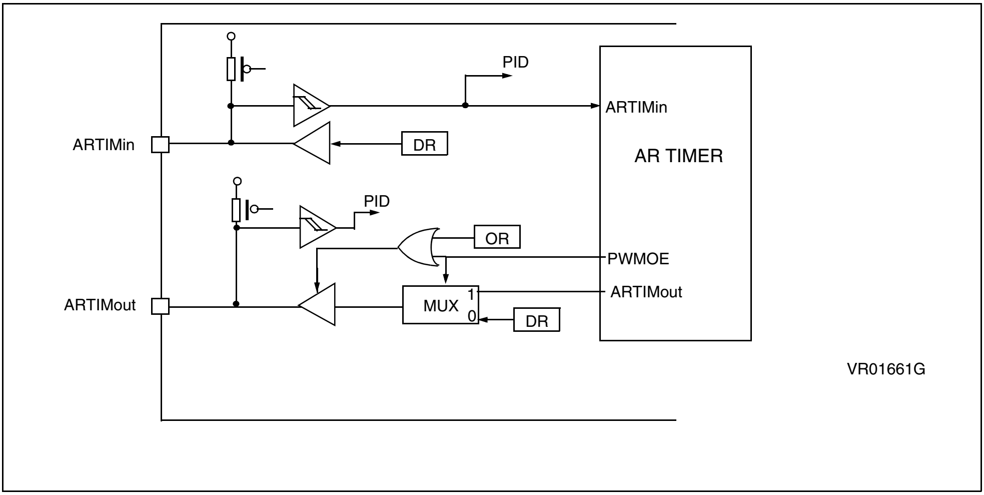
I/O PORTS (Cont'd)

4.1.3 ARTimer alternate functions

When bit PWMOE of register ARMC is low, pin ARTIMout/PB7 is configured as any standard pin of port B through the port registers. When PWMOE is high, ARTMout/PB7 is the PWM output, independently of the port registers configuration.

ARTIMin/PB6 is connected to the AR Timer input. It is configured through the port registers as any standard pin of port B. To use ARTIMin/PB6 as AR Timer input, it must be configured as input through DDRB.

Figure 24. Peripheral Interface Configuration of AR Timer



4.2 TIMER

The MCU features an on-chip Timer peripheral, consisting of an 8-bit counter with a 7-bit programmable prescaler, giving a maximum count of 2^{15}.

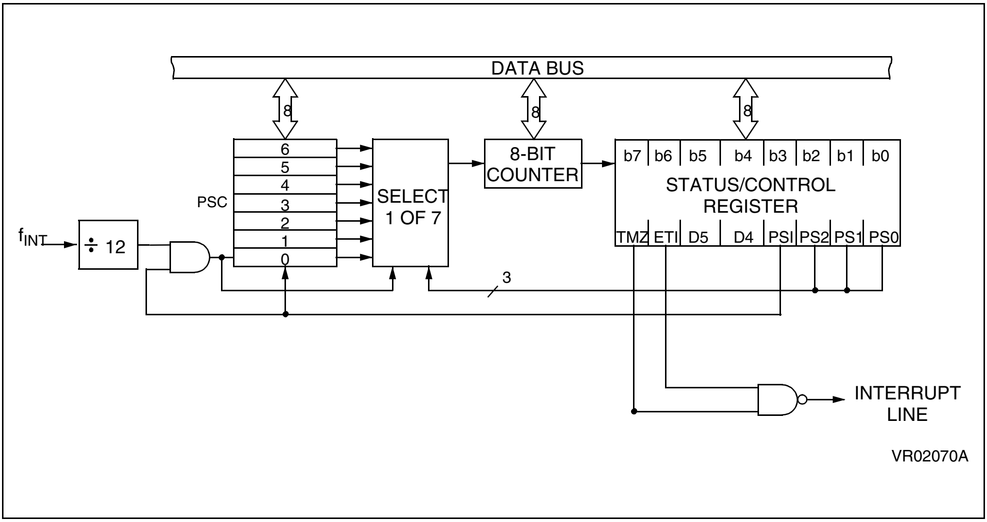
Figure 25. shows the Timer Block Diagram. The content of the 8-bit counter can be read/written in the Timer/Counter register, TCR, which can be addressed in Data space as a RAM location at address 0D3h. The state of the 7-bit prescaler can be read in the PSC register at address 0D2h. The control logic device is managed in the TSCR register 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. If the ETI (Enable Timer Interrupt) bit in the TSCR is also set, an interrupt request is generated. The Timer interrupt can be used to exit the MCU from WAIT mode.

The prescaler input is the internal frequency (f_{INT}) divided by 12. The prescaler decrements on the rising edge. Depending on the division factor programmed by PS2, PS1 and PS0 bits in the TSCR (see Table 13.), the clock input of the timer/counter register is multiplexed to different sources. For division factor 1, the clock input of the prescaler is also that of timer/counter; for factor 2, bit 0 of the prescaler register is connected to the clock input of TCR. This bit changes its state at half the frequency of the prescaler input clock. For factor 4, bit 1 of the PSC is connected to the clock input of TCR, and so forth. The prescaler initialize bit, PSI, in the TSCR register must be set to allow the prescaler (and hence the counter) to start. If it is cleared, all the prescaler bits are set and the counter is inhibited from counting. The prescaler can be loaded with any value between 0 and 7Fh, if bit PSI is set. The prescaler tap is selected by means of the PS2/PS1/PS0 bits in the control register.

Figure 26. illustrates the Timer's working principle.

Figure 25. Timer Block Diagram



4.3 AUTO-RELOAD TIMER

The Auto-Reload Timer (AR Timer) on-chip peripheral consists of an 8-bit timer/counter with compare and capture/reload capabilities and of a 7-bit prescaler with a clock multiplexer, enabling the clock input to be selected as f_{INT}, $f_{INT/3}$ or an external clock source. A Mode Control Register, ARMC, two Status Control Registers, ARSC0 and ARSC1, an output pin, ARTIMout, and an input pin, ARTIMin, allow the Auto-Reload Timer to be used in 4 modes:

– Auto-reload (PWM generation),

– Output compare and reload on external event (PLL),

– Input capture and output compare for time measurement.

– Input capture and output compare for period measurement.

The AR Timer can be used to wake the MCU from WAIT mode either with an internal or with an external clock. It also can be used to wake the MCU from STOP mode, if used with an external clock signal connected to the ARTIMin pin. A Load register allows the program to read and write the counter on the fly.

4.3.1 AR Timer Description

The AR COUNTER is an 8-bit up-counter incremented on the input clock's rising edge. The counter is loaded from the ReLoad/Capture Register, ARRC, for auto-reload or capture operations, as well as for initialization. Direct access to the AR counter is not possible; however, by reading or writing the ARLR load register, it is possible to read or write the counter's contents on the fly.

The AR Timer's input clock can be either the internal clock (from the Oscillator Divider), the internal clock divided by 3, or the clock signal connected to the ARTIMin pin. Selection between these clock sources is effected by suitably programming bits CC0-CC1 of the ARSC1 register. The output of the AR Multiplexer feeds the 7-bit programmable AR Prescaler, ARPSC, which selects one of the 8 available taps of the prescaler, as defined by PSC0-PSC2 in the AR Mode Control Register. Thus the division factor of the prescaler can be set to 2n (where n = 0, 1,..7).

The clock input to the AR counter is enabled by the TEN (Timer Enable) bit in the ARMC register. When TEN is reset, the AR counter is stopped and the prescaler and counter contents are frozen. When TEN is set, the AR counter runs at the rate of the selected clock source. The counter is cleared on system reset.

The AR counter may also be initialized by writing to the ARLR load register, which also causes an immediate copy of the value to be placed in the AR counter, regardless of whether the counter is running or not. Initialization of the counter, by either method, will also clear the ARPSC register, whereupon counting will start from a known value.

4.3.2 Timer Operating Modes

Four different operating modes are available for the AR Timer:

Auto-reload Mode with PWM Generation. This mode allows a Pulse Width Modulated signal to be generated on the ARTIMout pin with minimum Core processing overhead.

The free running 8-bit counter is fed by the prescaler's output, and is incremented on every rising edge of the clock signal.

When a counter overflow occurs, the counter is automatically reloaded with the contents of the Reload/Capture Register, ARCC, and ARTIMout is set. When the counter reaches the value contained in the compare register (ARCP), ARTIMout is reset.

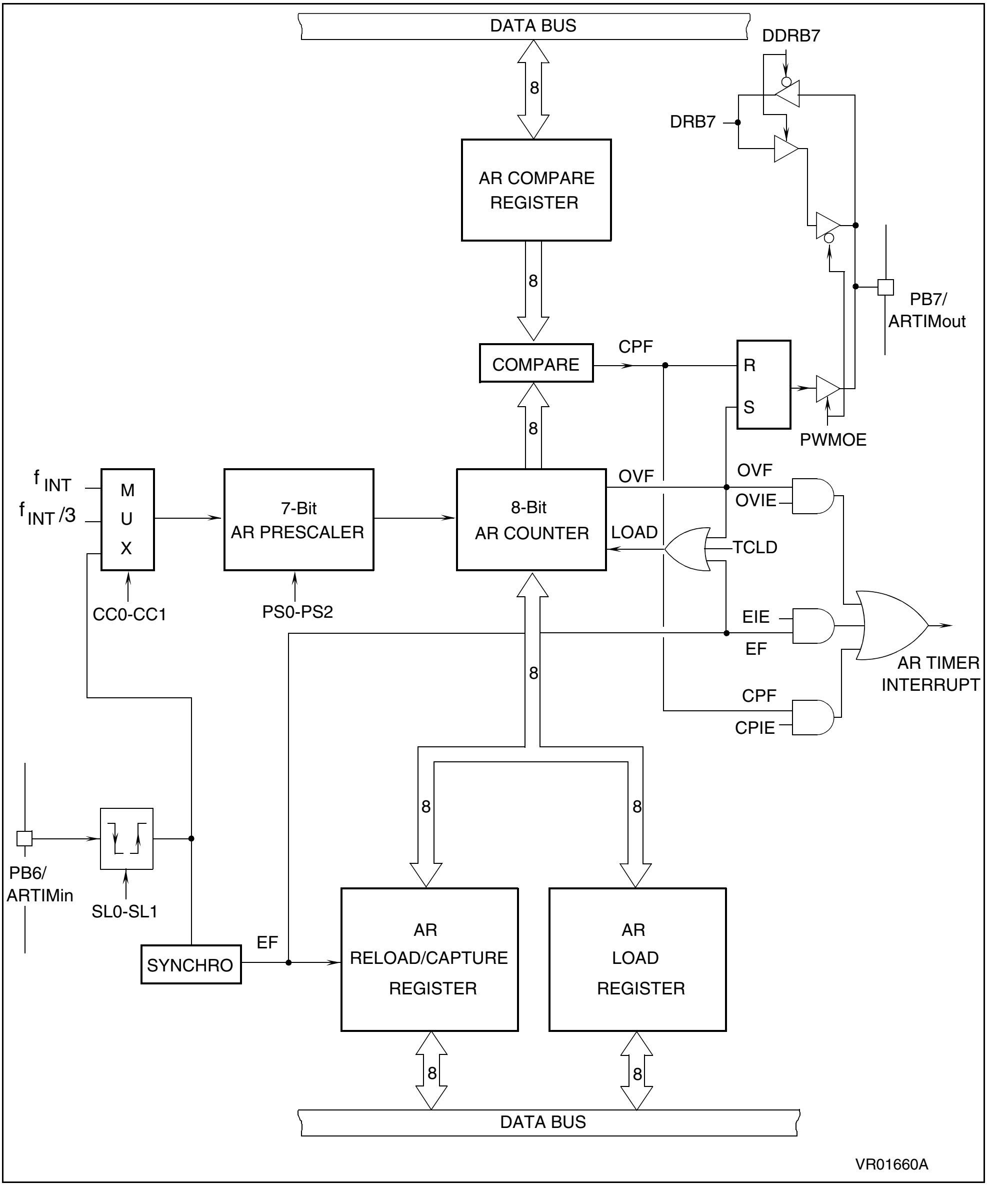
On overflow, the OVF flag of the ARSC0 register is set and an overflow interrupt request is generated if the overflow interrupt enable bit, OVIE, in the Mode Control Register (ARMC), is set. The OVF flag must be reset by the user software.

When the counter reaches the compare value, the CPF flag of the ARSC0 register is set and a compare interrupt request is generated, if the Compare Interrupt enable bit, CPIE, in the Mode Control Register (ARMC), is set. The interrupt service routine may then adjust the PWM period by loading a new value into ARCP. The CPF flag must be reset by user software.

The PWM signal is generated on the ARTIMout pin (refer to the Block Diagram). The frequency of this signal is controlled by the prescaler setting and by the auto-reload value present in the Reload/Capture register, ARRC. The duty cycle of the PWM signal is controlled by the Compare Register, ARCP.

AUTO-RELOAD TIMER (Cont'd)

Figure 27. AR Timer Block Diagram



AUTO-RELOAD TIMER (Cont'd)

AR Status Control Register 1(ARSC1)

Address: D7h — Read/Write

7							0
PS2	PS1	PS0	D4	SL1	SL0	CC1	CC0

Bist 7-5 = **PS2-PS0**: *Prescaler Division Selection Bits 2-0.* These bits determine the Prescaler division ratio. The prescaler itself is not affected by these bits. The prescaler division ratio is listed in the following table:

Table 14. Prescaler Division Ratio Selection

PS2	PS1	PS0	ARPSC Division Ratio
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

Bit 4 = **D4**: *Reserved*. Must be kept reset.

Bit 3-2 = **SL1-SL0**: *Timer Input Edge Control Bits 1-0.* These bits control the edge function of the Timer input pin for external synchronization. If bit SL0 is reset, edge detection is disabled; if set edge detection is enabled. If bit SL1 is reset, the AR Timer input pin is rising edge sensitive; if set, it is falling edge sensitive.

SL1	SL0	Edge Detection
X	0	Disabled
0	1	Rising Edge
1	1	Falling Edge

Bit 1-0 = **CC1-CC0**: *Clock Source Select Bit 1-0.* These bits select the clock source for the AR Timer through the AR Multiplexer. The programming of the clock sources is explained in the following Table 15 :

Table 15. Clock Source Selection.

CC1	CC0	Clock Source
0	0	F_{int}
0	1	F_{int} Divided by 3
1	0	ARTIMin Input Clock
1	1	Reserved

AR Load Register ARLR. The ARLR load register is used to read or write the ARTC counter register "on the fly" (while it is counting). The ARLR register is not affected by system reset.

AR Load Register (ARLR)

Address: DBh — Read/Write

7							0
D7	D6	D5	D4	D3	D2	D1	D0

Bit 7-0 = **D7-D0**: *Load Register Data Bits.* These are the load register data bits.

AR Reload/Capture Register. The ARRC reload/capture register is used to hold the auto-reload value which is automatically loaded into the counter when overflow occurs.

AR Reload/Capture (ARRC)

Address: D9h — Read/Write

7							0
D7	D6	D5	D4	D3	D2	D1	D0

Bit 7-0 = **D7-D0**: *Reload/Capture Data Bits*. These are the Reload/Capture register data bits.

AR Compare Register. The CP compare register is used to hold the compare value for the compare function.

AR Compare Register (ARCP)

Address: DAh — Read/Write

7							0
D7	D6	D5	D4	D3	D2	D1	D0

Bit 7-0 = **D7-D0**: *Compare Data Bits*. These are the Compare register data bits.

4.4 A/D CONVERTER (ADC)

The A/D converter peripheral is an 8-bit analog to digital converter with analog inputs as alternate I/O functions (the number of which is device dependent), offering 8-bit resolution with a typical conversion time of 70us (at an oscillator clock frequency of 8MHz).

The ADC 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, conversion accuracy is decreased.

Selection of the input pin is done by configuring the related I/O line as an analog input via the Option and Data registers (refer to I/O ports description for additional information). Only one I/O line must be configured as an analog input at any time. The user must avoid any situation in which more than one I/O pin is selected as an analog input simultaneously, to avoid device malfunction.

The ADC uses two registers in the data space: the ADC data conversion register, ADR, which stores the conversion result, and the ADC control register, ADCR, 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 is complete, the 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.

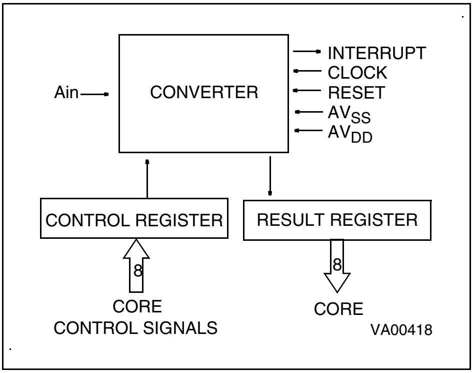
The STA bit is continuously scanned so that, if the user sets it to “1” while a previous conversion is in progress, a new conversion is started before completing the previous one. The start bit (STA) is a write only bit, any attempt to read it will show a logical “0”.

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

The power consumption of the device can be reduced by turning off the ADC peripheral. This is done by setting the PDS bit in the ADC control register to “0”. If PDS=“1”, the A/D is powered and enabled for conversion. This bit must be set at least one instruction before the beginning of the conversion to allow stabilisation of the A/D converter. This action is also needed before entering WAIT mode, since the A/D comparator is not automatically disabled in WAIT mode.

During Reset, any conversion in progress is stopped, the control register is reset to 40h and the ADC interrupt is masked (EAI=0).

Figure 29. ADC Block Diagram



4.4.1 Application Notes

The A/D converter does not feature a sample and hold circuit. The analog voltage to be measured should therefore be stable during the entire conversion cycle. Voltage variation should not exceed ±1/2 LSB for the optimum conversion accuracy. A low pass filter may be used at the analog input pins to reduce input voltage variation during conversion.

When selected as an analog channel, the input pin is internally connected to a capacitor C_{ad} of typically 12pF. For maximum accuracy, this capacitor must be fully charged at the beginning of conversion. In the worst case, conversion starts one instruction (6.5 µs) after the channel has been selected. In worst case conditions, the impedance, ASI, of the analog voltage source is calculated using the following formula:

$$6.5\mu s = 9 \times C_{ad} \times ASI$$

(capacitor charged to over 99.9%), i.e. 30 kΩ including a 50% guardband. ASI can be higher if C_{ad} has been charged for a longer period by adding instructions before the start of conversion (adding more than 26 CPU cycles is pointless).

5.3 INSTRUCTION SET

The ST6 core offers a set of 40 basic instructions which, when combined with nine addressing modes, yield 244 usable opcodes. They can be divided into six different types: load/store, arithmetic/logic, conditional branch, control instructions, jump/call, and bit manipulation. The following paragraphs describe the different types.

All the instructions belonging to 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 16. 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

Opcode Map Summary (Continued)

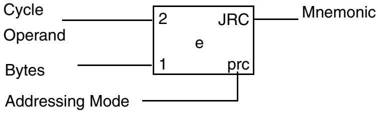
LOW HI	**8** **1000**	**9** **1001**	**A** **1010**	**B** **1011**	**C** **1100**	**D** **1101**	**E** **1110**	**F** **1111**	LOW HI
0 **0000**	2 JRNZ e 1 pcr	4 JP abc 2 ext	2 JRNC e 1 pcr	4 RES b0,rr 2 b.d	2 JRZ e 1 pcr	4 LDI rr,nn 3 imm	2 JRC e 1 prc	4 LD a,(y) 1 ind	**0** **0000**
1 **0001**	2 JRNZ 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 prc	4 LD a,rr 2 dir	**1** **0001**
2 **0010**	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	2 JRC e 1 prc	4 CP a,(y) 1 ind	**2** **0010**
3 **0011**	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 prc	4 CP a,rr 2 dir	**3** **0011**
4 **0100**	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 prc	4 ADD a,(y) 1 ind	**4** **0100**
5 **0101**	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 prc	4 ADD a,rr 2 dir	**5** **0101**
6 **0110**	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 prc	4 INC (y) 1 ind	**6** **0110**
7 **0111**	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 prc	4 INC rr 2 dir	**7** **0111**
8 **1000**	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 prc	4 LD (y),a 1 ind	**8** **1000**
9 **1001**	2 RNZ 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 prc	4 LD rr,a 2 dir	**9** **1001**
A **1010**	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 RCL a 1 inh	2 JRC e 1 prc	4 AND a,(y) 1 ind	**A** **1010**
B **1011**	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 prc	4 AND a,rr 2 dir	**B** **1011**
C **1100**	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 prc	4 SUB a,(y) 1 ind	**C** **1100**
D **1101**	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 prc	4 SUB a,rr 2 dir	**D** **1101**
E **1110**	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 prc	4 DEC (y) 1 ind	**E** **1110**
F **1111**	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 prc	4 DEC rr 2 dir	**F** **1111**

Abbreviations for Addressing Modes:

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

Legend:

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



6 ELECTRICAL CHARACTERISTICS

6.1 ABSOLUTE MAXIMUM RATINGS

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

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

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

Tj=TA + PD x RthJA

Where:TA = Ambient Temperature.

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

PD = Pint + Pport.

Pint =IDD x VDD (chip internal power).

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

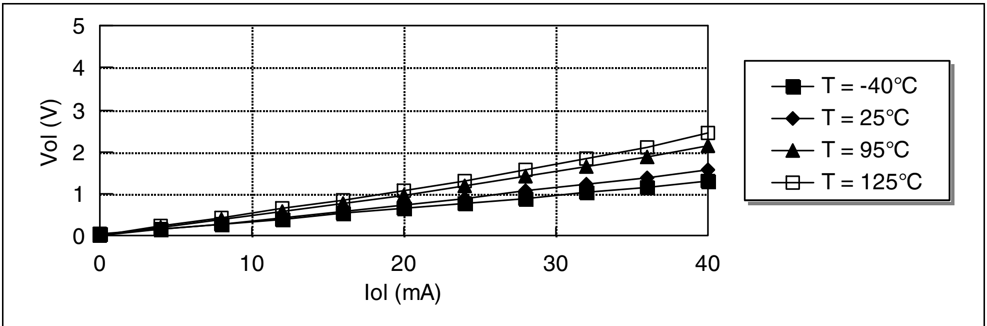
Symbol	Parameter	Value	Unit
V_{DD}	Supply Voltage	-0.3 to 7.0	V
V_I	Input Voltage	V_{SS} - 0.3 to V_{DD} + 0.3[(1)]	V
V_O	Output Voltage	V_{SS} - 0.3 to V_{DD} + 0.3[(1)]	V
IV_{DD}	Total Current into V_{DD} (source)	80	mA
IV_{SS}	Total Current out of V_{SS} (sink)	100	mA
Tj	Junction Temperature	150	°C
T_{STG}	Storage Temperature	-60 to 150	°C

Notes:

- 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.

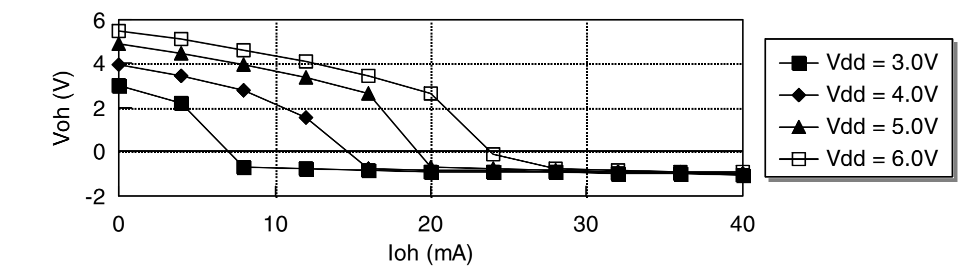
- (1) Within these limits, clamping diodes are guarantee to be not conductive. Voltages outside these limits are authorised as long as injection current is kept within the specification.

Figure 34. Vol versus Iol for High sink (30mA) I/O ports at Vdd=5V



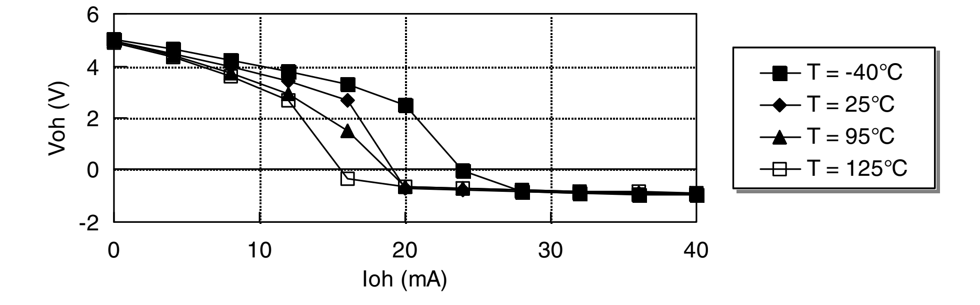
This curves represents typical variations and is given for guidance only

Figure 35. Voh versus Ioh on all I/O port at 25°C



This curves represents typical variations and is given for guidance only

Figure 36. Voh versus Ioh on all I/O port at Vdd=5V



This curves represents typical variations and is given for guidance only

Please Read Carefully:

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST’s terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN ST’S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZED ST REPRESENTATIVE, ST PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. ST PRODUCTS WHICH ARE NOT SPECIFIED AS "AUTOMOTIVE GRADE" MAY ONLY BE USED IN AUTOMOTIVE APPLICATIONS AT USER’S OWN RISK.

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan -
Malaysia - Malta - Morocco - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

www.st.com