MARC4 – 4-bit MTP Universal Microcontroller

The T48C510 is an Multi Time Programmable (MTP) microcontroller which is pin and functionally compatible to the Atmel Wireless & Microcontrollers' M44C510E mask programmable microcontroller. It contains EEPROM, RAM, up to 34 digital I/O pins, up to 10 maskable external interrupt sources, 4 maskable internal interrupts, a watchdog timer, interval timer, 2 x 8-bit multifunction timer/counter module and a versatile software configurable on-chip system clock module.

Features / Benefits

- Programmable system clock with prescaler and five different clock sources:
  - Up to 8-MHz crystal oscillator (system clock)
  - 32-kHz crystal oscillator
  - RC-oscillator fully integrated
  - RC-oscillator with external resistor adjustment
  - External clock input
- Wide supply-voltage range (2.4 V to 6.2 V)
- Very low halt current
- 4 KByte program EEPROM, 256 x 4-bit RAM
- 8 hard- and software interrupt priority levels
- Up to 10 external and 4 internal interrupts**,** bitwise maskable with programmable priority level
- Up to 34 I/O lines
- I/O ports – bitwise configurable with combined interrupt handling (for serial I/O applications)
- 2 x 8-bit multifunction timer/counters
- Coded reset and watchdog timer
- Power-on reset and "brown out" function
- Various power-down modes
- Efficient, hardware-controlled interrupt handling
- High-level programming language in qFORTH
- Comprehensive library of useful routines
- Windows 95/NT based development and programmer tools

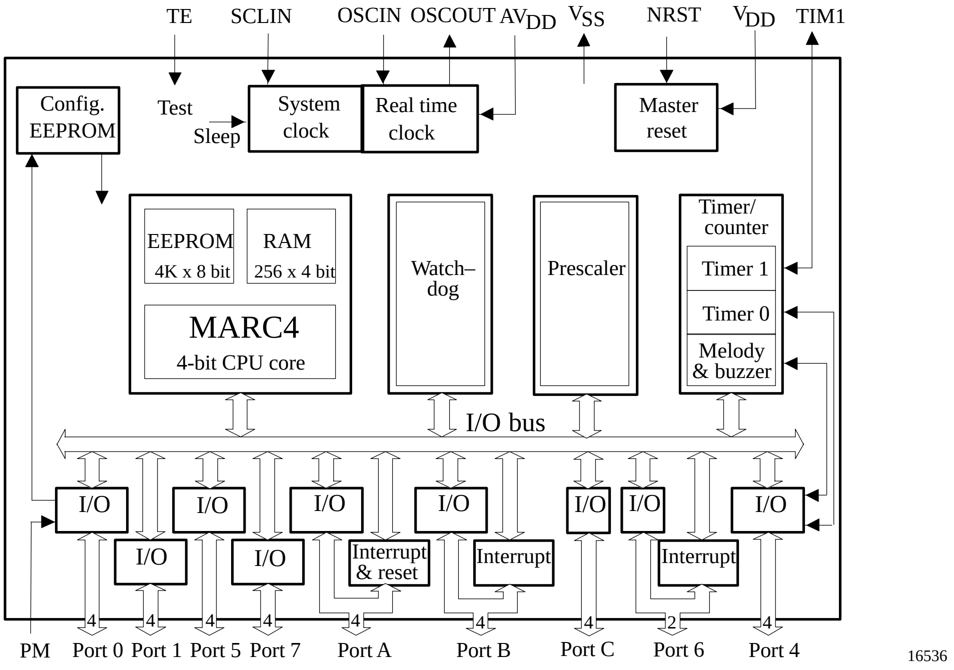


Figure 1. Block diagram

Ordering Information

Extended Type Number	Package	Remarks
T48C510 – ILS	SSO44	Stick
T48C510 – ILQ	SSO44	Taped and reeled

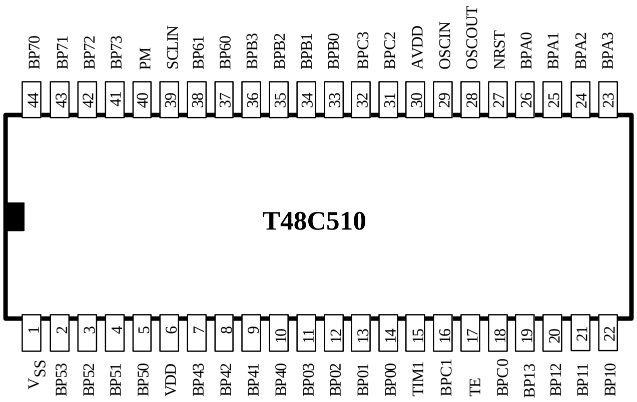


Figure 2. Pin connections SSO44-package

Table 1 Pin description

Name	Function
V_{DD}	Power supply voltage +2.4 V to +6.2 V
AV_{DD}	Analog power supply voltage +2.4 V to +6.2 V
V_{SS}	Circuit ground
BP00 – BP03	4 I/O lines of Port 0 – automatic nibblewise configurable / programmer interface
BP10 – BP13	4 I/O lines of Port 1 – automatic nibblewise configurable
BP50 – BP53	4 I/O lines of high current Port 5 – bitwise configurable
BP70 – BP73	4 I/O lines of high current Port 7 – bitwise configurable
BPA0 – BPA3	4 I/O lines of Port A – bitwise configurable, as inputs to a port monitor module and optional coded reset inputs
BPB0 – BPB3	4 I/O lines of Port B – bitwise configurable I/O and as inputs to a port monitor module
BPC0 – BPC3	4 I/O lines of Port C – bitwise configurable I/O
BP60 – BP61	2 I/O lines of Port 6 – bitwise configurable I/O or as external programmable interrupts
BP40 (T0OUT0)	I/O line BP40 of Port 4 – configurable or timer/counter I/O T0OUT0
BP41 (T0OUT1)	I/O line BP41 of Port 4 – configurable or timer/counter I/O T0OUT1
BP42 (BUZ)	High current I/O line BP42 of Port 4 – configurable or buzzer output BUZ
BP43 (NBUZ)	High current I/O line BP43 of Port 4 – configurable or buzzer output NBUZ
TIM1	Dedicated I/O for Timer 1
SCLIN	External trimming resistor or external clock input
OSCIN	32-kHz quartz crystal or 4-MHz quartz crystal input pin
OSCOUT	32-kHz quartz crystal or 4-MHz quartz crystal output pin
TE	Testmode input, used to control the production test modes (internal pull-down)
NRST	Reset input (/output), a logic low on this pin resets the device. An internal watchdog or coded reset can cause a low pulse on this pin.
PM	MTP program mode enable pin (internal pull-down)

Table of Contents

Table of Contents (continued)

1 MARC4 Architecture

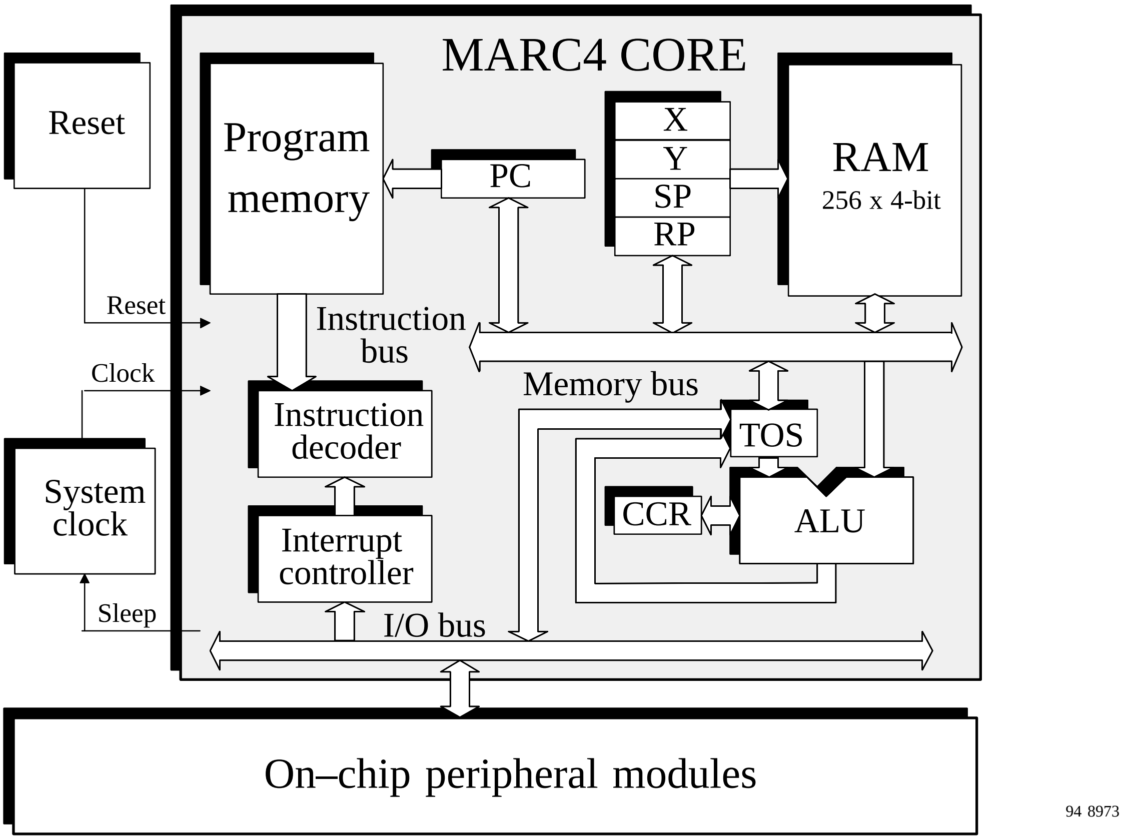


Figure 3. MARC4 core

1.1 General Description

The functionality, programming and pinning of the T48C510 is compatible with the M44C510E mask programmable microcontroller from Atmel Wireless & Microcontrollers. All on-chip modules are addressed and controlled with exactly the same programming code, so that a program targeted for the M44C510E can be read directly into the T48C510 and will operate in the same fashion.

The MARC4 microcontroller consists of an advanced stack based 4-bit CPU core and on-chip peripherals. The CPU is based on the HARVARD architecture with physically separate program memory (EEPROM) and data memory (RAM). Three independent buses, the instruction bus, the memory bus and the I/O bus are used for parallel communication between EEPROM, RAM and peripherals. This enhances program execution speed by allowing both instruction prefetching, and a simultaneous communication to the on-chip peripheral circuitry. The extremely powerful integrated interrupt controller with associated eight prioritized interrupt levels supports fast and efficient processing of hardware events. The MARC4 is designed for the high-level programming language qFORTH. The core includes an expression and a return stack. This architecture allows high-level language programming without any loss in efficiency or code density.

1.2 Components of MARC4 Core

The core contains EEPROM, RAM, ALU, a program counter, RAM address registers, an instruction decoder and an interrupt controller. The following sections describe each functional block in more detail:

1.2.1 EEPROM

The program memory (EEPROM) is programmed with the customer application program. The EEPROM is addressed by a 12-bit wide program counter, thus predefining a maximum program bank size of 4 Kbytes.

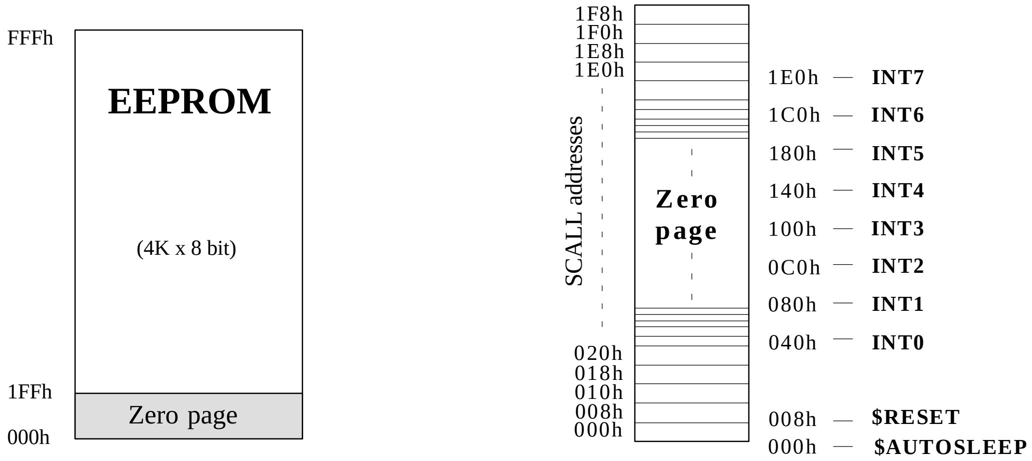


Figure 4. EEPROM map of T48C510

The lowest user EEPROM address segment is taken up by a 512-byte zero page which contains predefined start addresses for interrupt service routines and special subroutines accessible with single-byte instructions (SCALL). The corresponding memory map is shown in figure 4. Look-up tables of constants can also be held in EEPROM and are accessed via the MARC4's built-in TABLE instruction.

1.2.2 RAM

The MARC4 contains 256 x 4-bit wide static random access memory (RAM). It is used for the expression stack, the return stack and data memory for variables and arrays. The RAM is addressed by any of the four 8-bit wide RAM address registers SP, RP, X and Y.

Expression Stack

The 4-bit wide expression stack is addressed with the expression stack pointer (SP). All arithmetic, I/O and memory reference operations take their operands from, and return their result to the expression stack. The MARC4 performs the operations with the top of stack items (TOS and TOS-1). The TOS register contains the top element of the expression stack and works in the same way as an accumulator. This stack is also used for passing parameters between subroutines and as a scratch pad area for temporary storage of data.

Return Stack

The 12-bit wide return stack is addressed by the return stack pointer (RP). It is used for storing return addresses of subroutines, interrupt routines and for keeping loop index counts. The return stack can also be used as a temporary storage area.

The MARC4 instruction set supports the exchange of data between the top elements of the expression stack and the return stack. The two stacks within the RAM have a user-definable location and maximum depth.

1.2.3 Registers

The MARC4 controller has seven programmable registers and one condition code register. They are shown in figure 6.

Program Counter (PC)

The program counter (PC) is a 12-bit register that contains the address of the next instruction to be fetched from the EEPROM. Instructions currently being executed are decoded in the instruction decoder to determine the internal micro operations. For linear code (no calls or branches) the program counter is incremented with every instruction cycle. If a branch, call, return instruction or an interrupt is executed, the program counter is loaded with a new address. The program counter is also used with the TABLE instruction to fetch 8-bit wide constants.

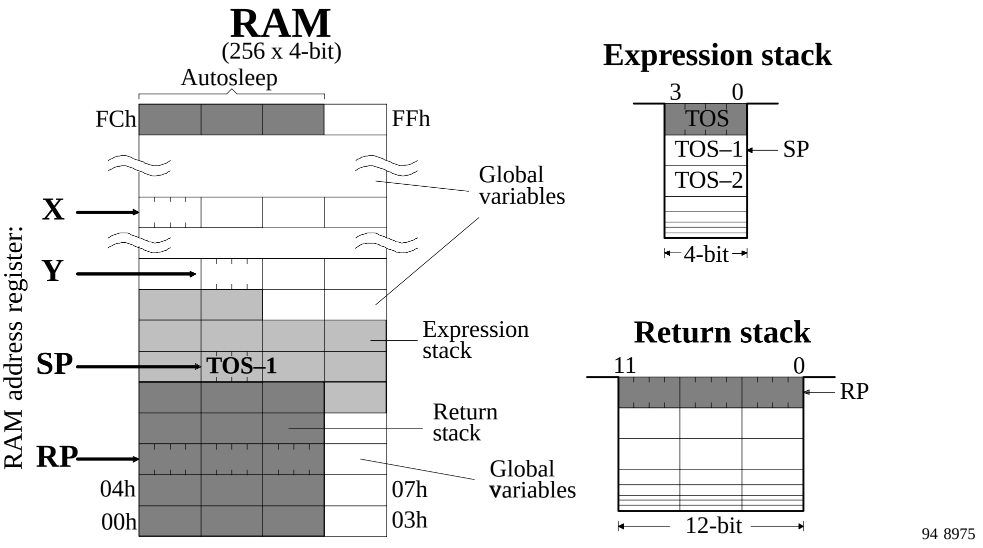


Figure 5. RAM map

RAM Address Registers

The RAM is addressed with the four 8-bit wide RAM address registers: SP, RP, X and Y. These registers allow access to any of the 256 RAM nibbles.

Expression Stack Pointer (SP)

The stack pointer (SP) contains the address of the next-to-top 4-bit item (TOS-1) of the expression stack. The pointer is automatically preincremented if a nibble is moved onto the stack, or postdecremented if a nibble is removed from the stack. Every postdecrement operation moves the item (TOS-1) to the TOS register before the SP is decremented. After a reset the stack pointer has to be initialized with “ >SP S0 ” to allocate the start address of the expression stack area.

Return Stack Pointer (RP)

The return stack pointer points to the top element of the 12-bit wide return stack. The pointer automatically pre-increments if an element is moved onto the stack or it postdecrements if an element is removed from the stack. The return stack pointer increments and decrements in steps of 4. This means that every time a 12-bit element is stacked, a 4-bit RAM location is left unwritten. These locations are used by the qFORTH compiler to allocate 4-bit variables. After a reset, the return stack pointer has to be initialized with “>RP FCh ”.

RAM Address Register (X and Y)

The X and Y registers are used to address any 4-bit item in the RAM. A fetch operation moves the addressed nibble onto the TOS. A store operation moves the TOS to the addressed RAM location. By using either the preincrement or postdecrement, addressing mode arrays in the RAM can be compared, filled or moved.

Top Of Stack (TOS)

The top of stack register is the accumulator of the MARC4. All arithmetic/logic, memory reference and I/O operations use this register. The TOS register receives data from the ALU, EEPROM, RAM or I/O bus.

Condition Code Register (CCR)

The 4-bit wide condition code register contains the branch, the carry and the interrupt-enable flag. These bits indicate the current state of the CPU. The CCR flags are set or reset by ALU operations. The instructions SET_BCF, TOG_BF, CCR! and DI allow direct manipulation of the condition code register.

Carry/Borrow (C)

The carry/borrow flag indicates that borrow or carry out of arithmetic logic unit (ALU) occurred during the last arithmetic operation. During shift and rotate operations, this bit is used as a fifth bit. Boolean operations have no affect on the C flag.

Branch (B)

The branch flag controls the conditional program branching. Should the branch flag have been set by a previous instruction, a conditional branch will cause a jump. This flag is affected by arithmetical, logical, shift, and rotate operations.

Interrupt Enable (I)

The interrupt-enable flag globally enables or disables the triggering of all interrupt routines with the exception of the non-maskable reset. After a reset, or on executing the

DI instruction, the interrupt-enable flag is reset, thus disabling all interrupts. The core will not accept any further interrupt requests until the interrupt-enable flag has been set again either by executing an EI, RTI or SLEEP instruction.

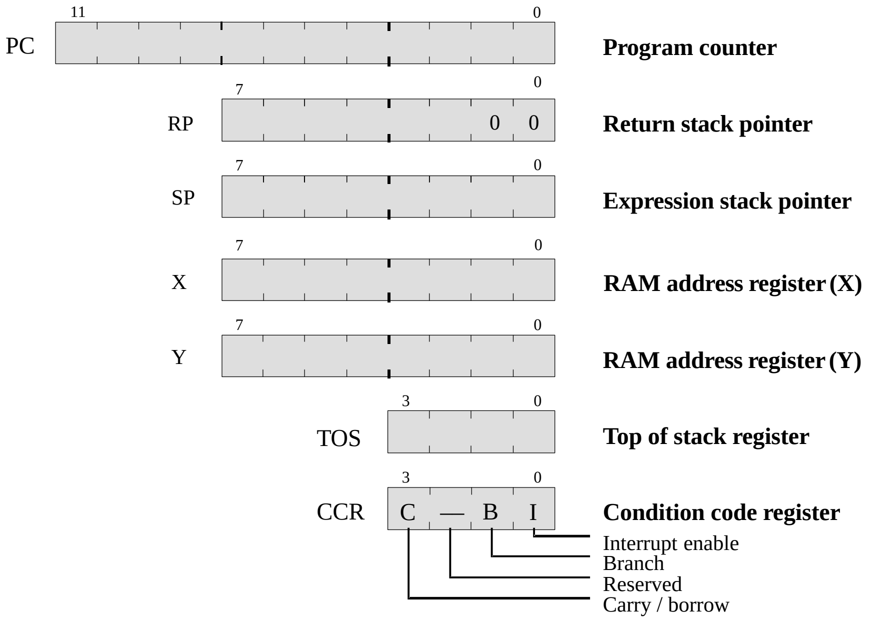


Figure 6. Programming model

1.2.4 ALU

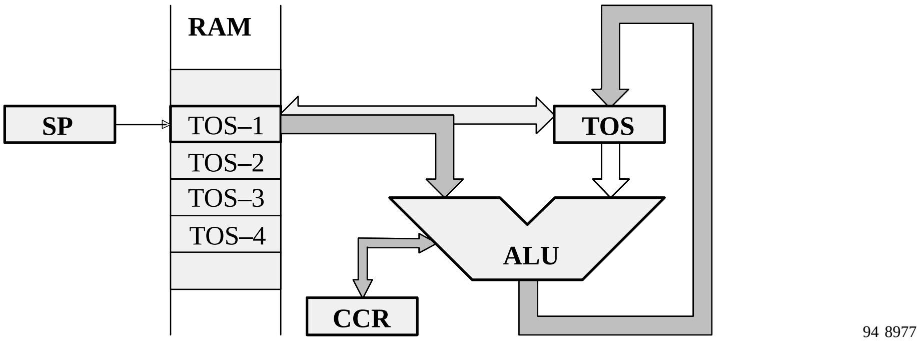


Figure 7. ALU zero-address operations

The 4-bit ALU performs all the arithmetical, logical, shift and rotate operations with the top two elements of the expression stack (TOS and TOS-1) and returns the result to the TOS. The ALU operations affect the carry/borrow and branch flag in the condition code register (CCR).

1.2.5 Instruction Set

The MARC4 instruction set is optimized for the high-level programming language qFORTH. Many MARC4 instructions are qFORTH words. This enables the compiler to generate a fast and compact program code. The CPU has an instruction pipeline which allows the controller to prefetch an instruction from EEPROM at the same time as the present instruction is being executed. The MARC4 is a zero-address machine. The instructions contain only the operation to be performed and no source or destination address fields. The operations are implicitly performed on the data placed on the stack. There are one and two byte instructions which are executed within 1 to 4 machine cycles. A MARC4 machine cycle is made up of two system clock (SYSCL) cycles. Most of the instructions are only one byte long and are executed in a single machine cycle.

1.2.6 I/O Bus

The I/O ports and the registers of the peripheral modules (Timer 0, Timer 1, Interval timer, Watchdog etc.) are I/O mapped. All communication between the core and the on-chip peripherals takes place via the I/O bus and the associated I/O control. With the MARC4 IN and OUT instructions, the I/O bus enables a direct read or write access to one of the 16 primary I/O addresses. More about the I/O access to the on-chip peripherals is described in the “Peripheral Modules”. The I/O bus is internal and is not accessible by the customer on the final microcontroller device, but is used as the interface for the MARC4 emulation.

1.3 Interrupt Structure

The MARC4 can handle interrupts with eight different priority levels. They can be generated from the internal and external interrupt sources or by a software interrupt from the CPU itself. Each interrupt level has a hard-wired priority and an associated vector for the service routine in the EEPROM (see table 2, page 11). The programmer can postpone the processing of interrupts by resetting the interrupt enable flag (I) in the CCR. An interrupt occurrence will still be registered but the interrupt routine is only started after the I flag is set. All interrupts can be masked, and the priority individually software configured by programming the appropriate control register of the interrupting module (see section “Peripheral Modules”).

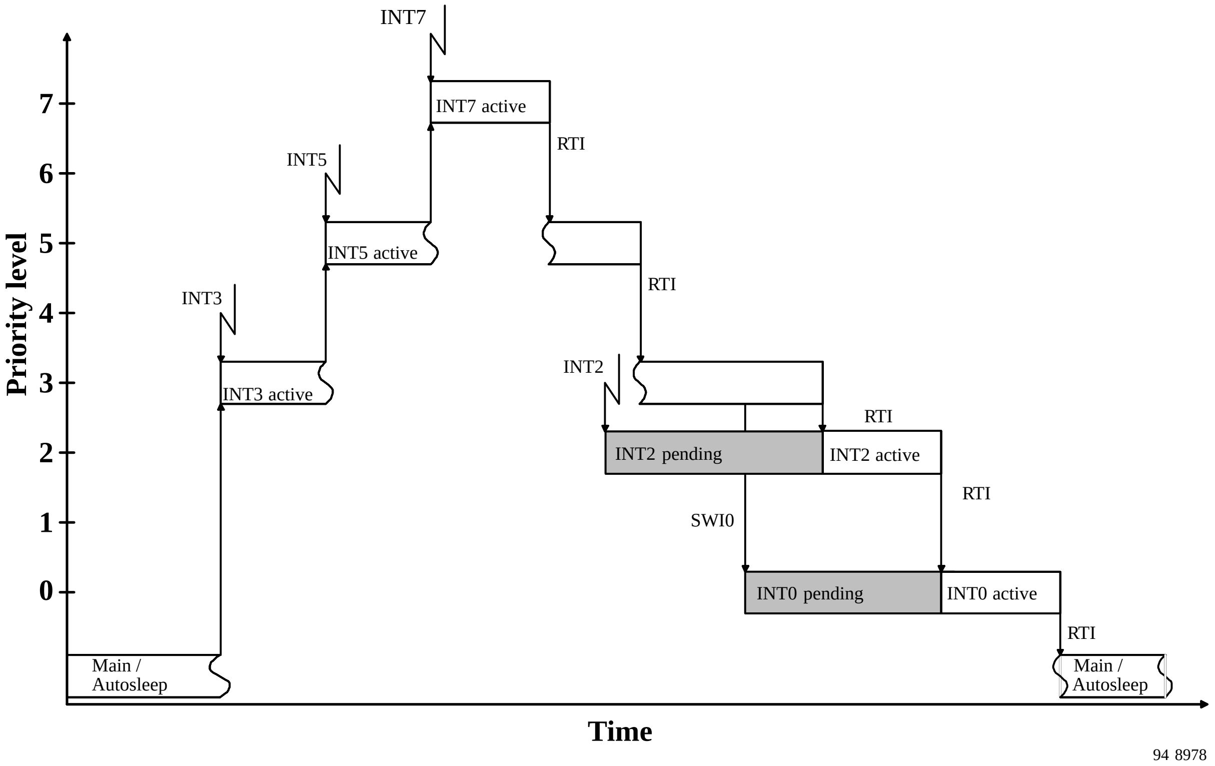


Figure 8. Interrupt handling

Interrupt Processing

For processing the eight interrupt levels, the MARC4 includes an interrupt controller with two 8-bit wide “interrupt pending” and “interrupt active” registers. The interrupt controller samples all interrupt requests during every non-I/O instruction cycle and latches these in the interrupt pending register. Whenever an interrupt request is detected, the CPU interrupts the program currently being executed, on condition that no higher priority interrupt is present in the interrupt active register. If the interrupt-enable bit is set, the processor enters an interrupt acknowledge cycle. During this cycle a short call (SCALL) instruction is executed to the service routine and the current PC is saved on the return stack. An interrupt service routine is finished with the RTI instruction. This instruction sets the interrupt-enable flag, resets the corresponding bits in the interrupt pending/active register and fetches the return address from the return stack to the program counter. When the interrupt-enable flag is reset (triggering of interrupt routines is disabled), the execution of new interrupt service routines is inhibited, but not the logging of the interrupt requests in the interrupt pending register. The execution of the interrupt is then delayed until the interrupt-enable flag is set again. Note that interrupts are only lost if an interrupt request occurs while the corresponding bit in the pending register is still set (i.e., the interrupt service routine is not yet finished).

After a master reset (power-on, external or watchdog reset), the interrupt-enable flag and the interrupt pending and interrupt active registers are all reset.

Interrupt Latency

The interrupt latency is the time from the occurrence of the interrupt to the interrupt service routine being activated. In the MARC4, this is extremely short and takes between 3 to 5 machine cycles depending on the state of the core.

Table 2 Interrupt priority table

Interrupt	Priority	EEPROM Address	Maskable	Interrupt Opcode
INT0	lowest	040h	Yes	C8h (SCALL 040h)
INT1	|	080h	Yes	D0h (SCALL 080h)
INT2	|	0C0h	Yes	D8h (SCALL 0C0h)
INT3	|	100h	Yes	E8h (SCALL 100h)
INT4	|	140h	Yes	E8h (SCALL 140h)
INT5	|	180h	Yes	F0h (SCALL 180h)
INT6	↓	1C0h	Yes	F8h (SCALL 1C0h)
INT7	highest	1E0h	Yes	FCh (SCALL 1E0h)

1.3.1 Hardware Interrupts

Table 3 Hardware interrupts

Interrupt	Possible Interrupt Priorities								RST	Interrupt Mask		Function
Source	0	1	2	3	4	5	6	7		Register	Bit	
NRST external									X	–	–	low level active
Watchdog									#	–	–	1/2 – 2 sec. time out
Port A coded reset									#	–	–	level any inputs
Port A monitor		*		*		*		*		PAIPR	3	any edge, any input
Port B monitor		*		*		*		*		PBIPR	3	any edge, any input
Port 60 external		*		*		*		*		P6CR	1,0	any edge
Port 61 external	*		*		*		*			P6CR	3,2	any edge
Interval timer INTA		*				*				ITIPR	0	1 of 8 frequencies (1 – 128 Hz)
Interval timer INTB			*				*			ITIPR	1	1 of 8 frequencies (8 – 8192 Hz)
Timer 0		*		*		*		*		T0CR	0	overflow/compare/ end measurement
Timer 1	*		*		*		*			T1CR	0	compare

X = hardwired (neither optional or software configurable)
# = configurable option (see “Hardware Options”)
* = software configurable (see “Peripheral Modules” section for further details)

In the T48C510, there are eleven hardware interrupt sources which can be programmed to occupy a variety of priority levels. With the exception of the reset sources (RST), each source can be individually masked by mask bits in the corresponding control registers. An overview of the possible hardware configurations is shown in table 3.

1.3.2 Software Interrupts

The program can generate interrupts using the software interrupt instruction (SWI) which is supported in qFORTH by predefined macros named SWI0...SWI7. The software triggered interrupt operates in exactly the same way as any hardware triggered interrupt.

The SWI instruction takes the top two elements from the expression stack and writes the corresponding bits via the I/O bus to the interrupt pending register. Thus, by using the SWI instruction, interrupts can be re-prioritized or lower priority processes scheduled for later execution.

1.4 Hardware Reset

The master reset forces the CPU into a well-defined condition, is unmaskable and is activated independent of the current program state. It can be triggered by either initial supply power-up, a short collapse of the power supply, a watchdog time out, activation of the NRST input or the occurrence of a coded reset on Port A (see figure 9). A master reset activation will reset the interrupt enable flag, the interrupt pending registers, the interrupt active registers and initialize all on-chip peripherals. In this state all ports take on a high resistance input status with deactivated pullup and pulldown transistors (see figure 10).

When the reset condition disappears, the hardware configuration previously programmed in the configuration EEPROM (see MTP Programming section) is loaded into the peripherals so that all port characteristics and pullup/ downs reflect the programmed configuration. This configuration period is immediately followed by a further reset delay time (approx. 80 ms), after which a short call instruction (opcode C1h) to the EEPROM address 008h is performed. This activates the initialization routine $RESET which in turn initializes all necessary RAM variables, stack pointers and internal peripheral configuration registers.

Power-on Reset

The fully integrated power-on reset circuit ensures that the core is held in a reset state until the minimum operating supply voltage has been reached. A reset condition is also generated should the supply voltage drop momentarily below the minimum operating supply.

External Reset (NRST)

An external reset can be triggered with the NRST pin. To activate an external reset, the pin should be low for a minimum of 4 µs.

Coded Reset (Port A)

The coded reset circuit is connected directly to the Port A terminals. By using the appropriate configuration, the user can define a hardwired code combination (e.g., all pins low) which, if occurring on the Port A, will generate a reset in the same way as the NRST pin.

Table 4 Multiple key reset options

NO_RST	Not used (default)
RST2	BPA0 & BPA1 = low
RST3	BPA0 & BPA1 & BPA2 = low
RST4	BPA0 & BPA1 & BPA2 & BPA3 = low
RST5	BPA0 & BPA1 = high
RST6	BPA0 & BPA1 & BPA2 = high
RST7	BPA0 & BPA1 & BPA2 & BPA3 = high

Note, that if this option is used, the reset is not maskable and will also trigger if the predefined code is written on to the Port A by the CPU itself. Care should also be taken not to generate an unwanted reset by inadvertently passing through the reset code on input transitions. This applies especially if the pins have a high capacitive load.

Watchdog Reset

The watchdog's function can be enabled via a configuration option and triggers a reset with every watchdog counter overflow. To suppress the watchdog reset, the counter must be regularly reset by reading the watchdog timer register address (CWD).

The CPU reacts in exactly the same manner as a reset stimulus from any of the above sources.

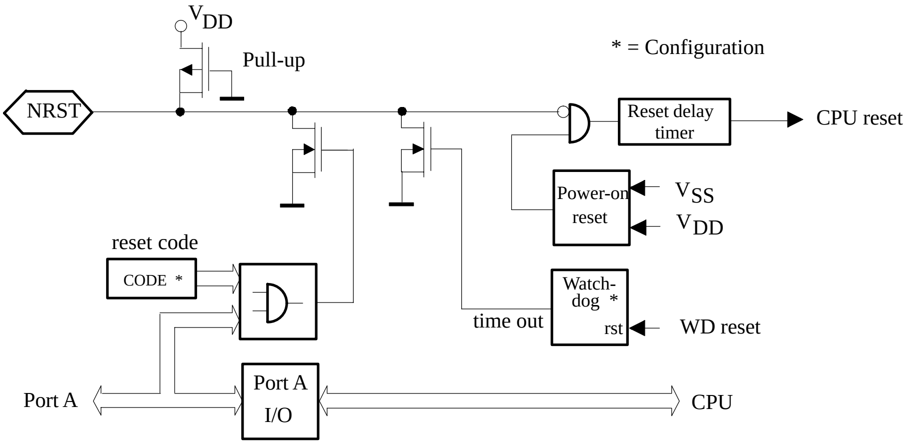


Figure 9. Reset configuration/ start-up sequence

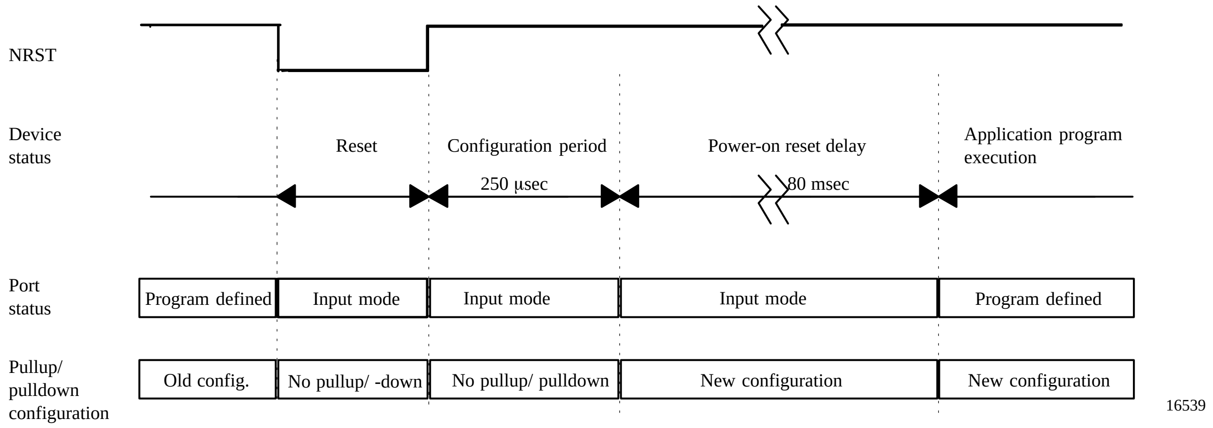


Figure 10. Normal mode start-up

1.5 Clock Generation

1.5.1 Clock Module

The clock module generates two clocks. The system clock (SYSCL) supplies the CPU and the peripherals while the lower frequency periphery sub-clock (SUBCL) supplies only the peripherals. The modes for clock sources are programmable with the OS1-bit and OS0-bit in the SC-register and the CCS-bit in the CM-register.

The clock module includes 4 different internal oscillator types: two RC-oscillators, one 4-MHz crystal oscillator and one 32-kHz crystal oscillator. The pins OSC1 and OSC2 provide the interface to connect a crystal either for the 4-MHz, or the 32-kHz crystal oscillator. SCLIN can be used as an input for an external clock or for connecting an external trimming resistor for the RC-oscillator 2. All necessary components with the exception of the crystal and the trimming resistor are integrated on-chip. Any one of these clock sources can be selected to generate the system clock (SYSCL).

In applications that do not require exact timing, it is possible to use the fully integrated RC-oscillator 1 without any external components. The RC-oscillator 2 is more stable but the oscillator frequency must be trimmed with an external resistor attached between SCLIN and V_{DD}. In this configuration, for system clock frequencies below 2 MHz, the RC-oscillator 2 frequency can be maintained to within a tolerance of ± 10% over the full operating temperature and voltage range.

The clock module is programmable via software using the clock management register (CM) and the system configuration register (SC). The required oscillator configuration is selected with the OS[1:0]-bits in the SC-register. A programmable 4-bit divider stage allows the adjustment of the system clock speed. A synchronization stage avoids any clock glitches which could be caused by clock source switching.

The CPU always requires SYSCL clocks to execute instructions, process interrupts and enter or leave the SLEEP state. Internal oscillators are, depending on the condition of the NSTOP-bit automatically stopped and started where necessary. Special care must however be taken when using an external clock source which is gated by a one of the microcontroller port signals. This configuration can hang up if the external oscillator is switched off while the external clock source is still selected. It is therefore advisable in such a case to switch first to the internal RC-oscillator 1 source using CSS-bit. The external source can then be reselected later when the external oscillator has again been restarted.

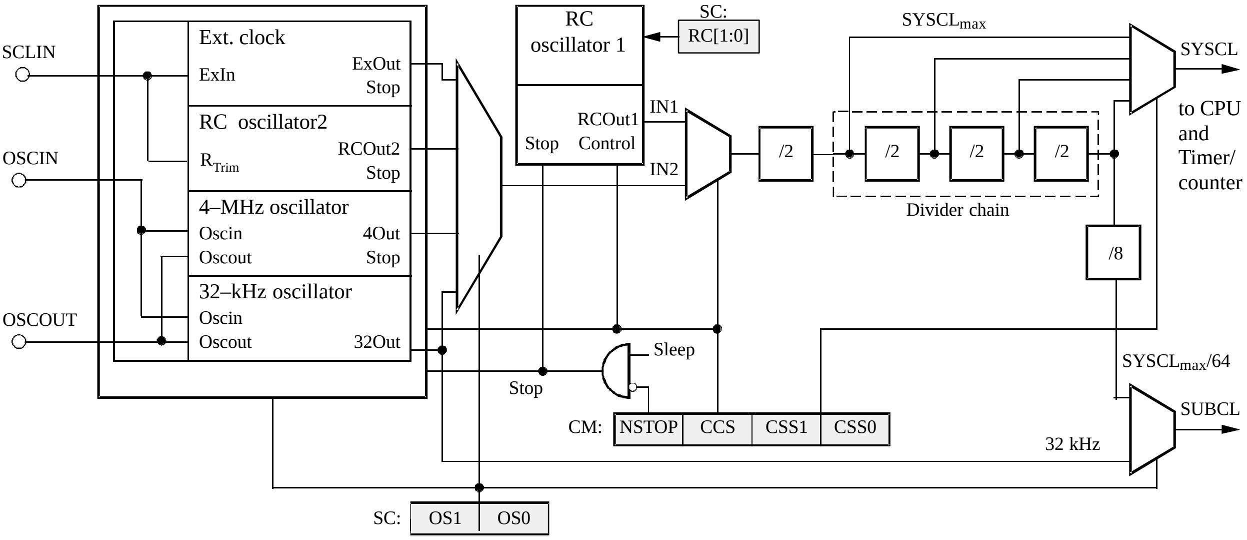


Figure 11. Clock module

Table 5 Clock modes

Mode			Clock Source for SYSCL		Clock Source for SUBCL	
	OS1	OS0	CCS = 1	CCS = 0	CCS = 1	CCS = 0
1	1	1	RC-oscillator 1 (intern)	External input clock	SYSCL $_{max}$/64	SCLIN / 128
2	0	1	RC-oscillator 1 (intern)	RC-oscillator 2 with external trimming resistor	SYSCL $_{max}$/64	SYSCL $_{max}$/64
3	1	0	RC-oscillator 1 (intern)	4-MHz oscillator	SYSCL $_{max}$/64	f_{xtal} / 128
4	0	0	RC-oscillator 1 (intern)	32-kHz oscillator	32 kHz	

1.5.2 Oscillator Circuits and External Clock Input Stage

RC-Oscillator 1 Fully Integrated

For timing insensitive applications, it is possible to use the fully integrated RC oscillator 1. This operates without any external components and thus saves on component costs. The RC-oscillator 1 frequency tolerance is better than ±50% over the full temperature and voltage range. A reduction in the application operating supply voltage and temperature ranges will result in an improved frequency tolerance. For more detailed information see figures 55 – 57. The basic center frequency of the RC-oscillator 1 is programmable with the RC1 and the RC0-bits in the SC register.

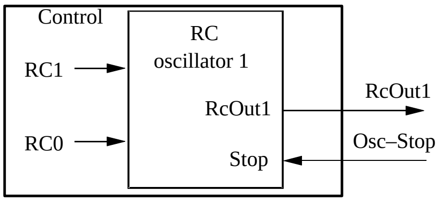


Figure 12. RC-oscillator 1

External Input Clock

The SCLIN pin can be driven by an external clock source provided it meets the specified input levels, duty cycle, rise and fall times. The maximum system clock frequency $f_{SYSCLmax}$ that the core can operate is $f_{SCLIN}/2$ (see figure 13).

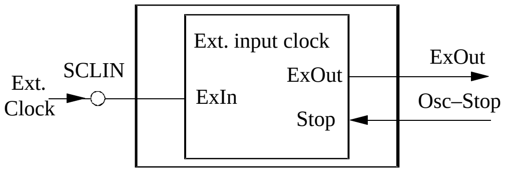


Figure 13. External input clock

RC-Oscillator 2 with External Trimming Resistor

The RC-oscillator 2 is a high stability oscillator whereby the oscillator frequency can be trimmed with an external resistor connected between SCLIN and V_{DD}. In this configuration, as long as the system clock frequency does not exceed 2 MHz, the RC-oscillator 2 frequency can be maintained stable to within a tolerance of ± 10% over the full operating temperature and voltage range.

For example: A $SYSCL_{max}$ frequency of 2 MHz, can be obtained by connecting a resistor R_{ext} = 150 kΩ (see figures 14, 52, 53 and 54).

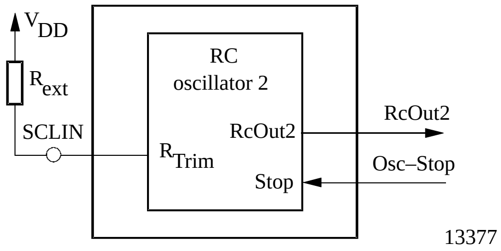


Figure 14. RC-oscillator 2

4-MHz Oscillator

The integrated system clock oscillator requires an external crystal or ceramic resonator connected between the OSCIN and OSCOUT pins to establish oscillation. All the necessary oscillator circuitry, with the exception of the actual crystal, resonator and the optional C3 and C4 are integrated on-chip.

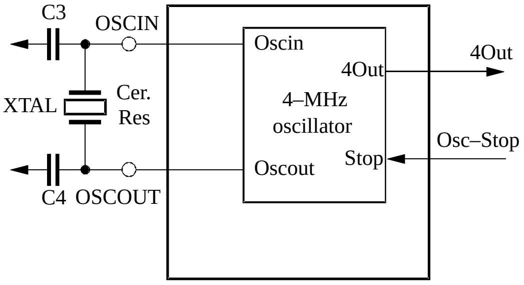


Figure 15. System clock oscillator

32-kHz Oscillator

Some applications require accurate long-term time keeping without putting excessive demands on the CPU or alternatively low resolution computing power. In this case, the on-chip ultra low power 32-kHz crystal oscillator can be used to generate both the SUBCL and/or the SYSCL. In this mode, power consumption can be significantly reduced. The 32-kHz crystal oscillator will remain operating (not stopped) during any CPU power-down/SLEEP mode.

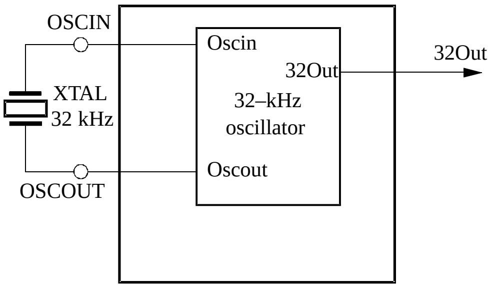


Figure 16. 32-kHz crystal oscillator

Quartz Oscillator Configuration

If the customer’s application necessitates the use of a quartz crystal clock source and this requires capacitive trimming, the trimming capacitors are not integrated into the MTP unlike the M44C510E and should therefore be connected externally as descrete components between the respective Quartz Crystal terminals (OSCIN, OSCOUT) and VSS.

1.5.3 Clock Management Register (CM)

The clock management register (CM) controls the system clock divider chain, as well as the peripheral clock in the power-down modes.

Auxiliary register address: ’E’hex

	Bit 3	Bit 2	Bit 1	Bit 0	
CM:	**NSTOP**	**CCS**	**CSS1**	**CSS0**	**Reset value: 1111b**

NSTOP **N**ot **STOP** peripheral clock
NSTOP = 0, stops the peripheral clock (SUBCL) when the core is in SLEEP mode.
The 32-kHz crystal oscillator SUBCL clock cannot be stopped.
NSTOP = 1, enables the peripheral clock (SUBCL) when the core in SLEEP mode

CCS **C**ore **C**lock **S**elect
CCS = 1, the internal RC-oscillator 1 generates SYSCL
CCS = 0, the 4-MHz crystal oscillator, the 32-kHz crystal oscillator, an external clock source or the RC-oscillator 2 (with the external resistor) will generate SYSCL dependent on the setting of OS0 and OS1 in the system configuration register

CSS[1:0] **C**ore **S**peed **S**elect These two bits control the system clock divider chain

Auxiliary register address: ’E’hex

CSS1	CSS0	Divider	Note
0	0	16	$SYSCL_{max}$/8
0	1	8	$SYSCL_{max}$/4
1	0	4	$SYSCL_{max}$/2
1	1	2	Reset value = $SYSCL_{max}$

System Configuration Register (SC)

Primary register address: ’E’hex

	Bit 3	Bit 2	Bit 1	Bit 0	
SC: write	**RC1**	**RC0**	**OS1**	**OS0**	**Reset value: 1111b**

RC1, RC0 Internal RC oscillator 1 frequency select ($SYSCL_{max}$)

RC1	RC0	SYSCLmax @ 25°C, V_{DD} = 5 V	Note
0	0	7.0 MHz (f_{iRC0})	
0	1	3.0 MHz (f_{iRC1})	
1	0	2.0 MHz (f_{iRC2})	
1	1	0.8 MHz (f_{iRC3})	Reset value

OS1, OS0 Oscillator selection bits (in conjunction with the CCS-bit)

CCS	OS1	OS0	SUBCL	System Oscillator Selection
0	1	1	$SYSCL_{max}$/64	External input clock at SCLIN
0	0	1		RC-oscillator 2 with R_{ext}
0	1	0		4-MHz crystal oscillator
0	0	0	32 kHz	32-kHz crystal oscillator
1	x	x	$SYSCL_{max}$/64 or 32 kHz	RC-oscillator 1

If CCS = 0 in the CM-register, the RC-oscillator 1 is stopped.

1.5.4 Power-down Modes

The T48C510 encorporates several modes which enable the power consumption to be tailored to a minimum without sacrificing computational power. When the controller exits the lowest priority interrupt task, it reverts to a SLEEP state. This is a CPU shutdown condition which is used to reduce average system power consumption where the CPU itself is only partially utilized. In SLEEP, the CPU clocking system is deactivated whereby the peripherals and associated clock sources may remain active (Standby Mode) or they can also be halted (Halt Mode). In Standby Mode, the peripherals are able to continue operation and if required also generate interrupts which can, along with a reset reactivate the CPU to bring it out of the sleep state.

SLEEP can only be maintained when none of the interrupt pending or active register bits are set. The application of the $AUTOSLEEP routine ensures the correct function of the sleep mode.

In both Standby and Active modes the current consumption is largely dependent on the frequency of the CPU system clock (SYSCL) and the supply voltage (V_{DD}). (see figures 50 and 51) while the Halt Mode current is merely controller static leakage current.

Selection of Standby or Halt mode is performed by the NSTOP bit in the clock managent register (CM). It should be noted that the low power 32-kHz crystal oscillator, if enabled will always remain active in both Standby and Halt modes.

Table 6 Power-down modes

Mode	CPU Core State	NSTOP	RC-Oscillator 1 RC-Oscillator 2 4-MHz Oscillator	32-kHz Oscillator	External Input Clock at SCLIN
Active	RUN	1	RUN	RUN	Enabled
Standby	SLEEP	1	RUN	RUN	Enabled
Halt	SLEEP	0	STOP	RUN	Disabled

1.5.5 Clock Monitor Mode

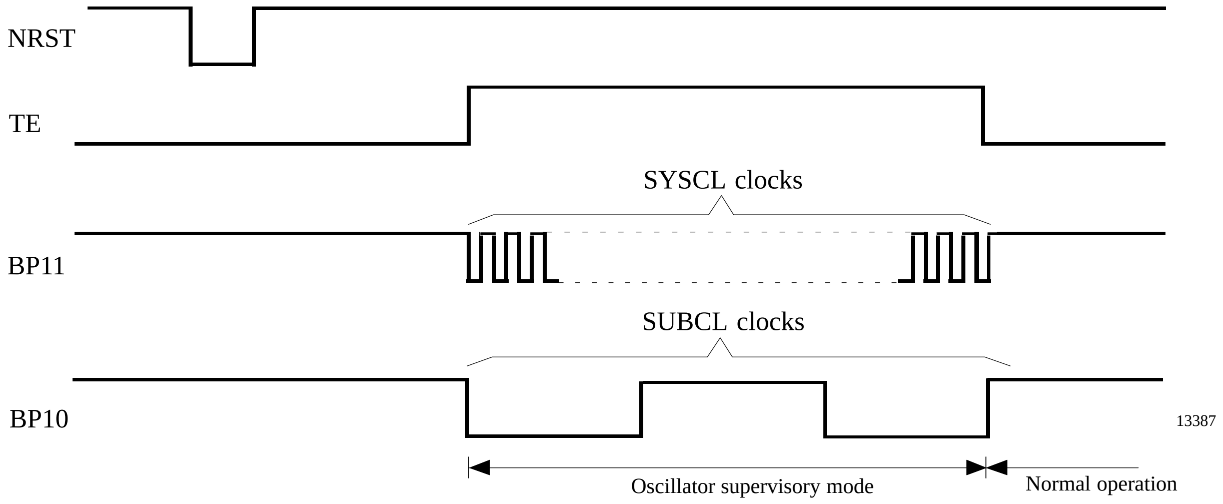


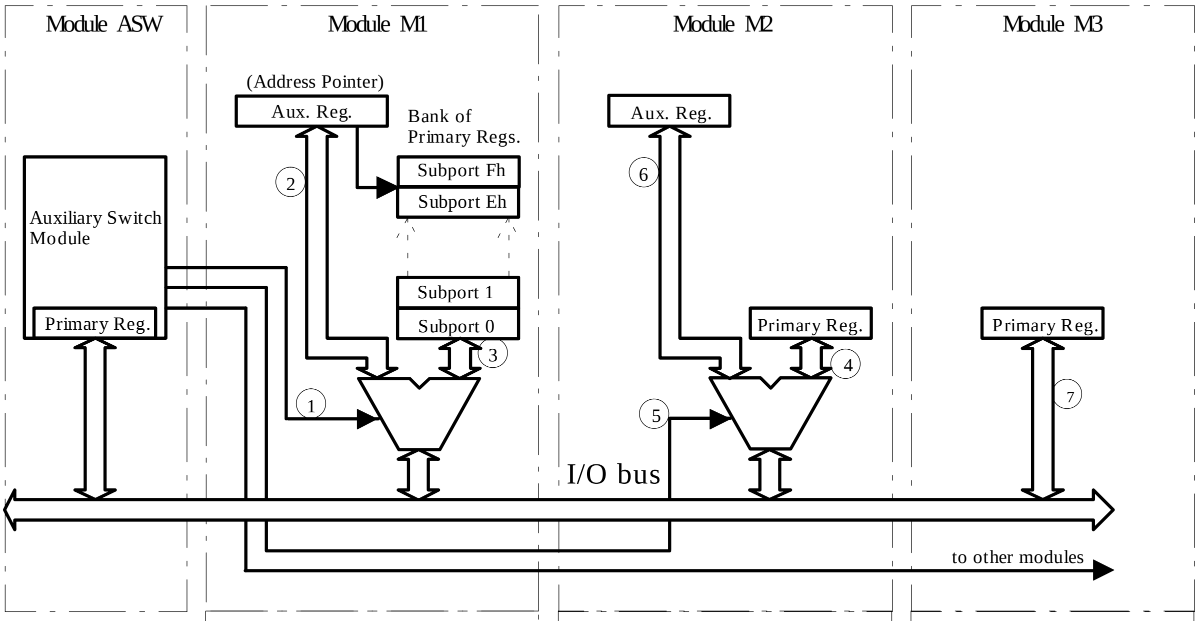
Figure 17. Clock monitoring

For trimming purposes, the T48C510 can be put into a clock monitor mode. By forcing the test input (TE) high, the SYSCL clock will appear on BP11 (Port 1, bit 1) and SUBCL clock on Port BP10 (Port 1, bit 0). On releasing the TE pin, the BP10 and BP11 will resume their normal function (see figure 17).

2 Peripheral Modules

2.1 Addressing Peripherals

Accessing the peripheral modules takes place via the I/O bus (see figure 18). The IN or OUT instructions allow direct addressing of up to 16 I/O modules. A dual register addressing scheme has been adopted which addresses the “primary register” directly. To address the “auxiliary register”, the access must be switched with an “auxiliary switching module”. Thus, a single IN (or OUT) to the module address will read (or write) into the module primary register. Accessing the auxiliary register is performed with the same instruction preceded by writing the module address into the auxiliary switching module. Byte-wide registers are accessed by multiple IN (or OUT) instructions. Extended addressing is used for more complex peripheral modules, with a larger number of registers. In this case, a bank of up to 16 subport registers are indirectly addressed with the subport address being initially written into the auxiliary register. Please refer to the ’HARDC510.SCR’ hardware interface file as a programming guideline.



Example of qFORTH Program Code

Indirect Subport Access

```
(Subport Register Write)
1  Addr.(M1)      Addr.(ASW) OUT
2  Addr.(SPort)   Addr.(M1)  OUT
3  SPort_Data     Addr.(M1)  OUT

(Subport Register Read)
1  Addr.(M1)      Addr.(ASW) OUT
2  Addr.(SPort)   Addr.(M1)  OUT
3                 Addr.(M1)  IN

(Subport Register Write Byte)
1  Addr.(M1)      Addr.(ASW) OUT
2  Addr.(SPort)   Addr.(M1)  OUT
3  SPort_Data(lo) Addr.(M1) OUT
3  SPort_Data(hi)  Addr.(M1) OUT

(Subport Register Read Byte)
1  Addr.(M1)      Addr.(ASW) OUT
2  Addr.(SPort)   Addr.(M1)  OUT
3                 Addr.(M1)  IN
3                 Addr.(M1)  IN

(Auxiliary Register Read)
1  Address(M1) Address(ASW) OUT
2              Address(M1)  IN
```

Dual Register Access

```
(Primary Register Write)
4  Prim._Data   Address(M2)  OUT

( Auxiliary Register Write )
5  Address(M2) Address(ASW) OUT
6  Aux._Data    Address(M2)  OUT

(Primary Register Read)
4               Address(M2)    IN

(Auxiliary Register Read)
5
   Address(M2) Address(ASW) OUT
6              Address(M2)  IN

(Auxiliary Register Write Byte)
5  Address(M2) Address(ASW) OUT
6  Aux._Data(lo) Address(M2)  OUT
6  Aux._Data(hi) Address(M2)  OUT
```

Single Register Access

```
(Primary Register Write)
7  Prim._Data Address(M3) OUT

(Primary Register Read)
7               Address(M3)  IN
```

Addr.(ASW) = Auxiliary Switch Module Address
Addr.(Mx) = Module Mx Address
Addr.(SPort) = Subport Address
Prim._Data = data to be written into Primary Register.
Aux._Data = data to be written into Auxiliary Register
Aux._Data (lo)= data to be written into Auxiliary Register (low nibble)
Aux._Data (hi) = data to be written into Auxiliary Register(high nibble)
SPort_Data(lo) = data to be written into SubPort (low nibble)
SPort_Data(hi) = data to be written into Subport (high nibble)

96 11522

Figure 18. Example of I/O addressing

Table 7 T48C510 Peripheral addresses

Port Address			Name	Write /Read	Reset Value	Register Function	Module Type	See Page
0			P0DAT	W/R	1111b	Port 0 – data register/input data	M3	21
1			P1DAT	W/R	1111b	Port 1 – data register/input data	M3	21
2			PAIPR	W	1111b	Port A – interrupt priority register	M2	22
	Aux.		PAICR	W	1111b	Port A – interrupt control register		22
3			CWD	R	——	Watchdog timer reset	M3	29
			PBIPR	W	1111b	Port B – interrupt priority register	M2	22
	Aux.		PBICR	W	1111b	Port B – interrupt control register		22
4			P4DAT	W/R	1111b	Port 4 – data register/pin data	M2	20
	Aux.		P4DDR	W	1111b	Port 4 – data direction register		21
5			P5DAT	W/R	1111b	Port 5 – data register/pin data	M2	20
	Aux.		P5DDR	W	1111b	Port 5 – data direction register		21
6			P6DAT	W/R	0011b	Port 6 – data register/pin data	M2	25
	Aux		P6CR	W	1111 1111b	Port 6 – control register (byte)		25
7			P7DAT	W/R	1111b	Port 7 – data register/pin data	M2	20
	Aux.		P7DDR	W	1111b	Port 7 – data direction register		21
8			ASW	W	1111b	Auxiliary switch register	ASW	18
9			TCM	W/R	1111b	Data to/from subport addressed by TCSUB	M1	18
	Aux.		T0SR	R	0000b	Timer 0 interrupt status register	M1	35
			TCSUB	W	1111b	Timer/counter subport address pointer	M1	30/31
		Subport address						
		0	T0MO	W	1111b	Timer 0 mode register	M1	35
		1	T0CR	W	1111b	Timer 0 control register	M1	36
		2	T1MO	W	1111b	Timer 1 mode register	M1	43
		3	T1CR	W	1111b	Timer 1 control register	M1	43/44
		4	TCMO	W	1111b	Timer/counter mode register	M1	33
		5	TCIOR	W	1111b	Timer/counter I/O control register	M1	32
		6	TCCR	W	1111b	Timer/counter control register	M1	31
		7	TCIP	W	1111b	Timer/counter interrupt priority	M1	32
		8	T1CP	W	xxxx xxxxb	Timer 1 compare register (byte)	M1	44
			T1CA	R	xxxx xxxxb	Timer 1 capture register (byte)	M1	
		9	T0CP	W	xxxx xxxxb	Timer 0 compare register (byte)	M1	37
			T0CA	R	xxxx xxxxb	Timer 0 capture register (byte)	M1	
		A	BZCR	W	1111b	Buzzer control register	M1	47
		B-F			——	Reserved		
A			PADAT	W/R	1111b	Port A – data register/pin data	M2	20
	Aux.		PADDR	W	1111b	Port A – data direction register		21
B			PBDAT	W/R	1111b	Port B – data register/pin data	M2	20
	Aux.		PBDDR	W	1111b	Port B – data direction register		21
C			PCDAT	W/R	1111b	Port C – data register/pin data	M2	20
	Aux.		PCDDR	W	1111b	Port C – data direction register		21
D			—–	—–	——	Reserved		
E			SC	W	1111b	System configuration register	M2	16
	Aux.		CM	W/R	1111b	Clock management register		15
F			ITFSR	W	1111b	Interval timer frequency select register	M2	28
	Aux.		ITIPR	W	1111b	Interval timer interrupt priority register		

2.2 Bidirectional Ports

Table 8 Overview of Port Features

Port Address	0	1	4	5	6	7	A	B	C
Number of bits	4	4	4	4	2	4	4	4	4
Bitwise programmable direction	no	no	yes	yes	yes	yes	yes	yes	yes
Output drivers configurable [1]	no [2]	yes	yes	yes	yes	yes	yes	yes	yes
Dynamic pullup/ down typ. (Ohm) [3]	500k	500k	500k	500k	500k	500k	500k	500k	500k
Static pullup/ down typ. (Ohm) [4]	none	none	30k	30k	4k	30k	30k	30k	30k
Schmitt trigger inputs	yes	yes	yes	no	yes	no	yes	yes	no
Additional functions			Timer 0		External interrupt		Port monitor/ coded reset	Port monitor	

[1] Either ”open drain down”, ”open drain up” or CMOS output configuration.

[2] This output must always be CMOS.

[3] The Dynamic pullup/down transistors are configurable and if selected, are only activated when the associated complementry driver transistor is off. ie. A dynamic pull up transistor is only active when the port is either in input mode (both drivers off) or when a logical 1 is written to the port pad (low driver off) in output mode. (figure 20)

[4] The Static Pullup/down transitors are configurable and if selected, are always active independant of the port direction or driven state. (figure 20)

For further data see section 3.2 .

All Ports (0, 1, 4, 5, 7, A, B and C with the exception of Port 6) are 4 bits wide. Port 6 has a data width of 2 bits (bit 0 and bit 1) only. The ports may be used for data input or output. All ports that can either directly or indirectly generate an interrupt are equipped with Schmitt-trigger inputs. A variety configurable options are available such as open drain, open source and full complementary outputs as well as different types of pull-up and pull-down transistors. All Port Data Registers (PxDAT) are I/O mapped to the primary address register of the respective port address, and the Port Data Direction Register (PxDDR) to the corresponding auxiliary register.

All bidirectional ports except Port 0 and Port 1, include a bitwise- programmable Data Direction Register (PxDDR) which allows the individual programming of each port bit as input or output. It is also possible to read the pin condition when in output mode. This is a useful feature for self testing and for collision detection on wired-OR bus systems.

There are five different types of bidirectional ports:

- Ports 0 and 1 – 4-bit wide, bidirectional ports with automatic full bus width direction switching.
- Port 4 – 4-bit wide, bitwise programmable bidirectional port also provides the I/O interface to Timer 0 and the Buzzer.
- Ports 5, 7 and C – 4-bit wide, bitwise programmable high drive I/O port.
- Port 6 – 2-bit wide, bitwise programmable bidirectional ports with optional static (4 kΩ) pull-up/-down and programmable interrupt logic.
- Ports A and B – 4-bit wide, bitwise programmable bidirectional ports with optional port monitor function.

Port Data Register (PxDAT)

Primary register address: ’Port address’hex

	Bit 3	Bit 2	Bit 1	Bit 0	
PxDAT	**PxDAT3**	**PxDAT2**	**PxDAT1**	**PxDAT0**	**Reset value: 1111b**

Bit 3 → MSB, bit 0 → LSB, x → Port address

Port Data Direction Register (PxDDR)

Auxiliary register address: ’Port address’hex

	Bit 3	Bit 2	Bit 1	Bit 0	
PxDDR	**PxDDR3**	**PxDDR2**	**PxDDR1**	**PxDDR0**	**Reset value: 1111b**

Table 9 Port Data Direction Register (PxDDR)

Code: 3 2 1 0	Function
x x x 1	BPx0 in input mode
x x x 0	BPx0 in output mode
x x 1 x	BPx1 in input mode
x x 0 x	BPx1 in output mode
x 1 x x	BPx2 in input mode
x 0 x x	BPx2 in output mode
1 x x x	BPx3 in input mode
0 x x x	BPx3 in output mode

2.2.1 Bidirectional Port 0 and Port 1

In this port type, the data direction register is not independently software programmable because the direction of the complete port is switched automatically when an I/O instruction occurs (see figure 19). The port can be switched to output mode with an OUT instruction and to input with an IN instruction. The data written to a port will be stored in the output data latches and appears immediately at the port pin following the OUT instruction. After RESET, all output latches are set to ’1’ and the ports are switched to input mode. An IN instruction reads the condition of the associated pins.

Note: Care must be taken when switching these bidirectional ports from output to input. The capacitive pin loading at this port, in conjunction with the high resistance pull-ups, may cause the CPU to read the contents of the output data register rather than the external input state. This can be avoided by using either of the following programming techniques:

- Use two IN instructions and DROP the first data nibble. The first IN switches the port from output to input and the DROP removes the first invalid nibble. The second IN reads the valid pin state.
- Use an OUT instruction followed by an IN instruction. With the OUT instruction, the capacitive load is charged or discharged depending on the optional pull-up /pull-down configuration. Write a “1” for pins with pull-up resistors, and a “0” for pins with pull-down resistors.

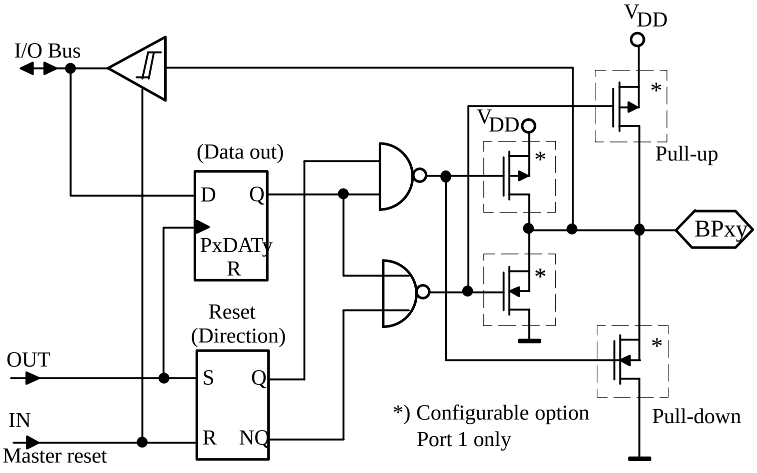


Figure 19. Bidirectional Ports 0 and 1

2.2.2 Bidirectional Port 5, Port 7 and Port C

All bidirectional ports except Port 0 and Port 1, include a bitwise-programmable Data Direction Register (PxDDR) which allows the individual programming of each port bit as input or output. It also enables the reading of the pin condition in output mode.

The bidirectional Ports 5, 7 and C as well as Port A and Port B are equipped with the same standard I/O logic. However, Port 5, Port 7 and Port C include standard CMOS input stages, whereas Port A, Port B and all other digital signal pins have Schmitt-trigger inputs. Port 5 and Port 7 have high current output drive capability for up to 20 mA @ 5 V. Whereby the instantaneous sum of the output currents should not exceed 100 mA.

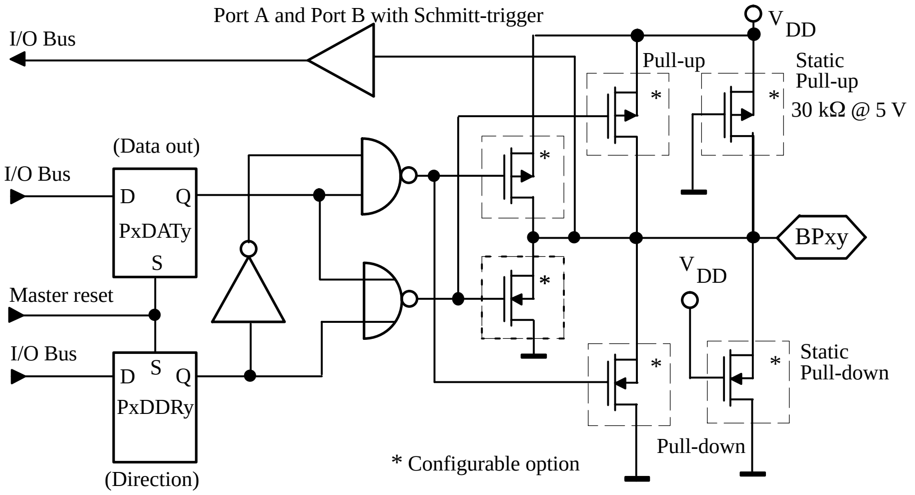
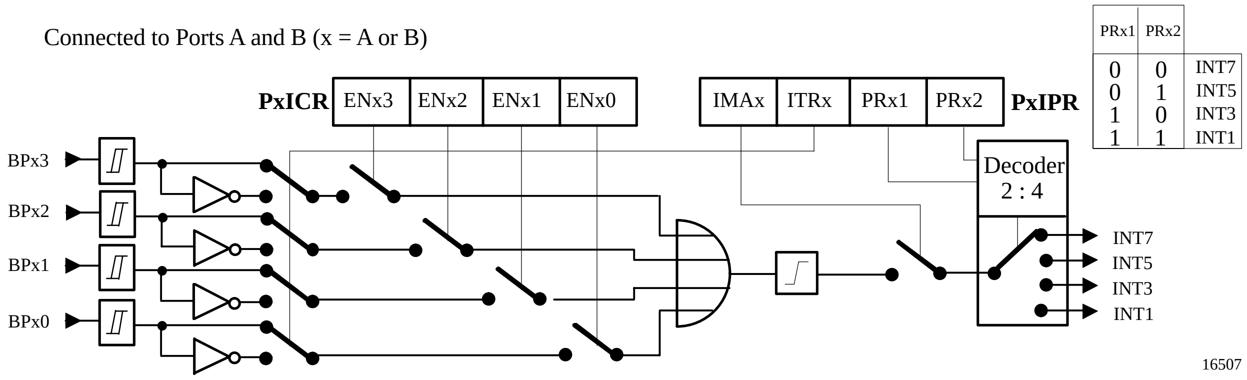


Figure 20. Bidirectional Ports 5, 7, A, B and C

2.2.3 Bidirectional Port A and Port B with Port Monitor Function



PRx1	PRx2	
0	0	INT7
0	1	INT5
1	0	INT3
1	1	INT1

Figure 21. Port monitor module of Port A and Port B

In addition to the standard I/O functions described in section 2.2.2, both Port A (BPA3 – BPA0) and Port B (BPB3 – BPB0) are equipped with Schmitt-trigger inputs and a port monitor module. This module is connected across all four port pins (see figure 21) and is intended for monitoring those pins selected by control bits Enx3 – Enx0 and generating an interrupt when the first pin leaves a preselected logical default idle state. This state is defined by control bit ITRx . Transitions on other pins will only cause an interrupt if the other pins have first returned to the idle state. This, for example is useful for interrupt initiated port scanning without the power consuming task of continuously polling for port activity.

Using the Port Interrupt Control Register (PxICR), pins can be individually selected. A non-selected pin cannot generate an interrupt. The Port Interrupt Priority Register (PxIPR) allows masking of each interrupt, definition of the interrupt edge and programming of the interrupt priority levels. When programming or reprogramming either of the port monitor control registers, any previously generated interrupt on that port which has not yet been acknowledged by the CPU or an interrupt generated by the reprogramming itself is automatically cleared. Port A can also be used for a configurable coded reset. For more information see section 1.4 ’Hardware Reset’.

The Port Interrupt Priority Registers PAIPR and PBIPR are I/O mapped to the the primary address registers of the Port Monitor Module addresses ’2’h and ’3’h respectively. The Port Interrupt Control Registers PAICR and PBICR are mapped to the corresponding auxiliary registers.

Port Monitor Interrupt Priority Register (PxIPR)

x = ’A’ (Port A) or ’B’ (Port B)

(Port A) Primary register address: ’2’hex
(Port B) Primary register address: ’3’hex

	Bit 3	Bit 2	Bit 1	Bit 0	
PxIPR	**IMx**	**ITRx**	**PRx2**	**PRx1**	**Reset value: 1111b**

IMx	– Interrupt Mask
ITRx	– Interrupt Transition
PRx2..1	– Interrupt Priority code

Table 10 Port Monitor Interrupt Priority Register (PxIPR)

Code 3 2 1 0	Function
x x 0 0	Port monitor interrupt priority 7
x x 0 1	Port monitor interrupt priority 5
x x 1 0	Port monitor interrupt priority 3
x x 1 1	Port monitor interrupt priority 1
x 0 x x	Port monitor interrupt on falling edge
x 1 x x	Port monitor interrupt on rising edge
0 x x x	Port monitor interrupt enabled
1 x x x	Port monitor interrupt disabled

Port Monitor Interrupt Control Register (PxICR)

x = ’A’ (Port A) or ’B’ (Port B)

(Port A) Auxiliary register address: ’2’hex
(Port B) Auxiliary register address: ’3’hex

	Bit 3	Bit 2	Bit 1	Bit 0	
PxICR	**ENx3**	**ENx2**	**ENx1**	**ENx0**	**Reset value: 1111b**

ENx3 ... 0 port monitor input ENable code

Table 11 Port Monitor Interrupt Control Register (PxICR)

Code 3 2 1 0	Function
x x x 0	Bit 0 can generate an interrupt
x x x 1	Bit 0 cannot generate an interrupt
x x 0 x	Bit 1 can generate an interrupt
x x 1 x	Bit 1 cannot generate an interrupt
x 0 x x	Bit 2 can generate an interrupt
x 1 x x	Bit 2 cannot generate an interrupt
0 x x x	Bit 3 can generate an interrupt
1 x x x	Bit 3 cannot generate an interrupt

2.2.4 Bidirectional Port 6

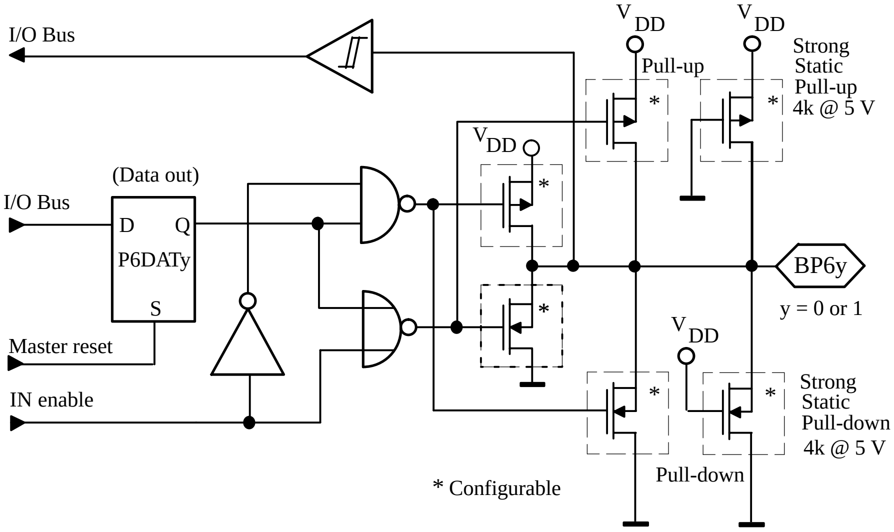


Figure 22. Bidirectional Port 6

This 2-bit bidirectional port can be used as bitwise-programmable I/O. The data is LSB aligned so that the two MSB's will not appear on the port pins when written. The port pins can also be used as external interrupt inputs (see figures 22 and 23). Both interrupts can be masked or independently configured to trigger on either edge. The interrupt priority levels are also configurable. The interrupt configuration and port direction is controlled by the Port 6 Control Register (P6CR). An additional low resistance pull-up transistor (configurable option) provides an internal bus pull-up for serial bus applications.

In output mode (PxDDR bit = 0), the respective Port Data Register (PxDAT) bit appears on the port pin, driven by an output port driver stage which can be configurable as open drain, or full complementary CMOS. With an IN instruction the actual pin state can be read back into the controller at any time without changing the port directional mode. If the output port is configured as an open drain driver, the controller is able to receive the external data on this pin without switching into input mode as long as the output transistor is switched off.

In input mode (PxDDR bit = 1), the output driver stage is deactivated, so that an IN instruction will directly read the pin state which can be driven from an external source. In this case, the state of the Port Data Register (PxDAT), although not appearing at the pin itself, remains unchanged. High resistance configurable pull-up or pull-down transistors are automatically switched onto the port pin in input mode. The Port Data Register is written to the respective port address with an OUT instruction.

The Port 6 Data Register (P6DAT) is I/O mapped to the primary address register of address '6'hex and the Port 6 Control Register (P6CR) to the corresponding auxiliary register. The P6CR is a byte wide register and is written by writing the low nibble first and then the high nibble (see section 2.1 “Addressing peripherals”).

Port 6 Data Register (P6DAT)

Primary register address: ’6’hex

	Bit 3	Bit 2	Bit 1	Bit 0	
P6DAT	**not used**	**not used**	**P6DAT1**	**P6DAT0**	**Reset value: xx11b**

The unused bits 2 and 3 are ’0’, if read.

Port 6 Control Register (P6CR)

Auxiliary register address: ’6’hex

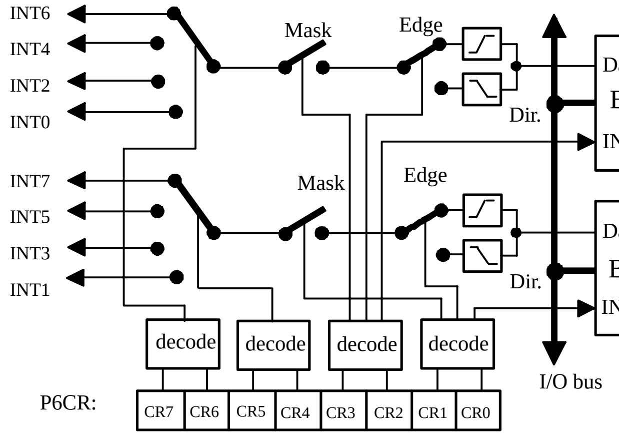
		Bit 3	Bit 2	Bit 1	Bit 0	
P6CR	**First write cycle**	**P61IM2**	**P61IM1**	**P60IM2**	**P60IM1**	**Reset value: 1111b**
		Bit 7	Bit 6	Bit 5	Bit 4	
	Second write cycle	**P61PR2**	**P61PR1**	**P60PR2**	**P60PR1**	**Reset value: 1111b**

P6xIM2, P6xIM1 – Port 6x Interrupt mode/direction code

P6xPR2, P6xPR1 – BP6x Interrupt priority code

Table 12 Port 6 control register (P6CR)

Auxiliary Address: ’6’hex First Write Cycle		**Second Write Cycle**	
Code 3 2 1 0	Function	Code 3 2 1 0	Function
x x 1 1	BP60 in input mode – interrupt disabled	x x 1 1	BP60 set to priority 1
x x 0 1	BP60 in input mode – rising edge interrupt	x x 1 0	BP60 set to priority 3
x x 1 0	BP60 in input mode – falling edge interrupt	x x 0 1	BP60 set to priority 5
x x 0 0	BP60 in output mode – interrupt disabled	x x 0 0	BP60 set to priority 7
1 1 x x	BP61 in input mode – interrupt disabled	1 1 x x	BP61 set to priority 0
0 1 x x	BP61 in input mode – rising edge interrupt	1 0 x x	BP61 set to priority 2
1 0 x x	BP61 in input mode – falling edge interrupt	0 1 x x	BP61 set to priority 4
0 0 x x	BP61 in output mode – interrupt disabled	0 0 x x	BP61 set to priority 6



CR7	CR6	
0	0	INT6
0	1	INT4
1	0	INT2
1	1	INT0

CR5	CR4	
0	0	INT7
0	1	INT5
1	0	INT3
1	1	INT1

CR1	CR0	Dir.	INT	INT
CR3	CR2		edge	disabled
0	0	out	–	yes
0	1	in	↑	no
1	0	in	↓	no
1	1	in	–	yes

96 11526

Figure 23. Port 6 external interrupts

2.2.5 Bidirectional Port 4

The bidirectional Port 4 is both a bitwise configurable I/O port and provides the external pins for both the Timer 0 and the internal buzzer generator. As an I/O port, it performs in exactly the same way as bidirectional Port 5, 7, A, B and C (see figure 20). Two additional multiplexers allow data and port direction control to be passed over to other internal modules (Timer 0 or Buzzer). Each of the four Port 4 pins can be individually switched by the Timer/Counter I/O Register (TCIO). Figure 24 shows the internal interfaces to Port 4.

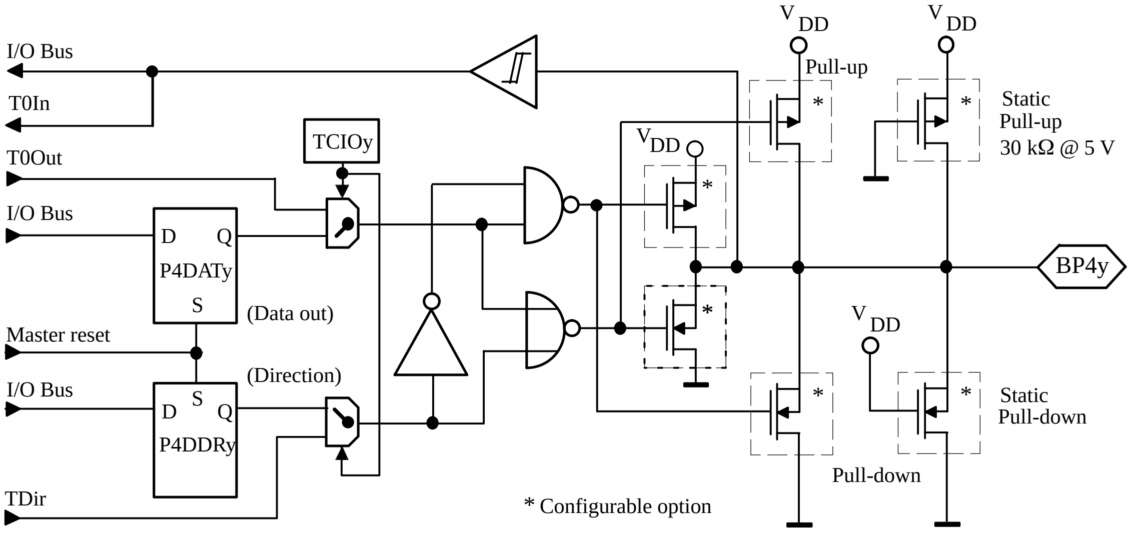


Figure 24. Bidirectional Port 4

2.2.6 TIM1 – Dedicated Timer 1 I/O Pin

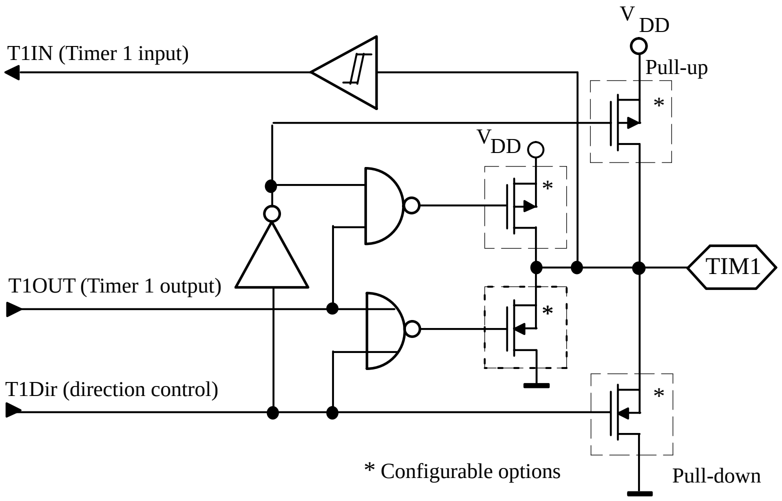


Figure 25. Bidirectional pin TIM1

TIM1 is a dedicated bidirectional I/O stage for signal communication to and from the Timer 1 in the timer/ counter module (see figure 25). It has no I/O bus interface and is not directly accessible from the CPU. The direction control is performed from the timer/counter configuration registers.

2.3 Interval Timers / Prescaler

The interval timers are based on a frequency divider for generating two independent time base interrupts. It is driven by SUBCL generated by the clock module (see figure 11) and consists of a 15-stage binary divider and two programmable multiplexers for selecting the appropriate interrupt frequencies for each interrupt source (see figure 26). Each multiplexer is completely independent and is controlled by the common Interval Timer Frequency Select Register (ITFSR). Buffer registers store the respective frequency select codes and ensure complete programming independence of each interrupt channel.

Interrupt masking and programming of the interrupt priority levels is performed with the aid of the Interval Timer Interrupt Priority Register (ITIPR).

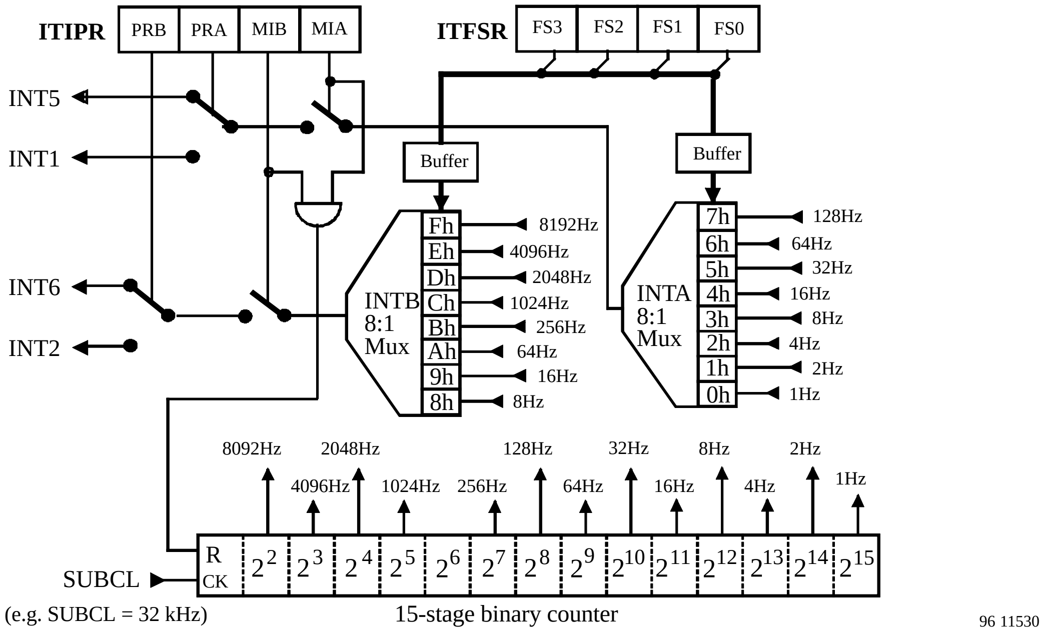


Figure 26. Interval timers / prescaler

2.3.1 Interval Timer Registers

The Interval Timer Frequency Select Register (ITFSR) is I/O mapped to the primary address register of the prescaler/ interval timer address ('F'hex) and the Interval Timer Interrupt Priority Register (ITIPR) to the corresponding auxiliary register. The interrupt masks MIA and MIB enable interrupt masking of INTA and INTB respectively. Each interrupt source can be programmed with PRA and PRB to one of two interrupt priority levels. Disabling both interrupts resets the interval timer.

Interval Timer Interrupt Priority Register (ITIPR)

Auxiliary register address (write only): 'F'hex

	Bit 3	Bit 2	Bit 1	Bit 0	
ITIPR	**PRB**	**PRA**	**MIB**	**MIA**	**Reset value: 1111b**

PRB – Priority select Interval Timer Interrupt INTB
PRA – Priority select Interval Timer Interrupt INTA
MIB – Mask Interval Timer Interrupt INTB
MIA – Mask Interval Timer Interrupt INTA

Table 13 Interval Timer Interrupt Priority Register (ITIPR)

Code 3 2 1 0	Function
x x 1 1	Reset prescaler and halt
x x x 1	Interrupt A disabled
x x x 0	Interrupt A enabled
x x 1 x	Interrupt B disabled
x x 0 x	Interrupt B enabled
x 1 x x	Interrupt A => priority 1
x 0 x x	Interrupt A => priority 5
1 x x x	Interrupt B => priority 2
0 x x x	Interrupt B => priority 6

Interval Timer Frequency Select Register (ITFSR)

Primary register address (write only): 'F'hex

	Bit 3	Bit 2	Bit 1	Bit 0	
ITFSR	**FS3**	**FS2**	**FS1**	**FS0**	**Reset value: 1111b**

FS3 ... 0 – Frequency select code

Table 14 Interval Timer Frequency Select Register (ITFSR)

Code 3 2 1 0	Function	SUBCL divide by	SUBCL = 32 kHz
0 0 0 0	INTA	2^{15}	Select 1 Hz
0 0 0 1		2^{14}	Select 2 Hz
0 0 1 0		2^{13}	Select 4 Hz
0 0 1 1		2^{12}	Select 8 Hz
0 1 0 0		2^{11}	Select 16 Hz
0 1 0 1		2^{10}	Select 32 Hz
0 1 1 0		2^{9}	Select 64 Hz
0 1 1 1		2^{8}	Select 128 Hz
1 0 0 0	INTB	2^{12}	Select 8 Hz
1 0 0 1		2^{11}	Select 16 Hz
1 0 1 0		2^{9}	Select 64 Hz
1 0 1 1		2^{7}	Select 256 Hz
1 1 0 0		2^{5}	Select 1024 Hz
1 1 0 1		2^{4}	Select 2048 Hz
1 1 1 0		2^{3}	Select 4096 Hz
1 1 1 1		2^{2}	Select 8192 Hz

The control bit FS3 determines whether the INTA or the INTB buffer register is loaded with the select code (FS2–FS0). This allows independent programming of interval times for INTA and INTB.

2.4 Watchdog Timer

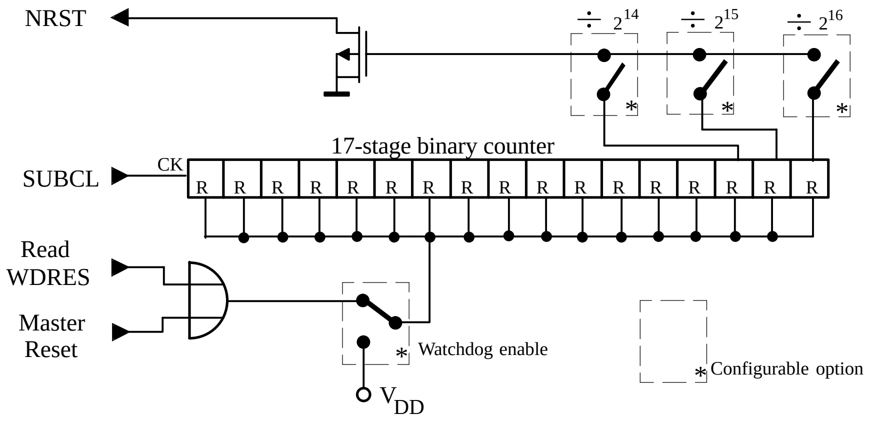


Figure 27. Watchdog timer

The watchdog timer is a 17-stage binary divider clocked by SUBCL generated within the clock module (see figures 11 and 27). It can only be enabled as a configurable option whereby it must be periodically reset from the application program. The program cannot disable the watchdog. If the CPU find itself for an extended length of time in SLEEP mode or in a section of program that includes no watchdog reset, then the watchdog will overflow, thus forcing the NRST pin low. This initiates a master reset. The timeout period can be set to 0.5, 1 or 2 seconds (if SUBCL = 32 kHz) by using a configurable option.

To reset the watchdog, the program must perform an IN-instruction on the address CWD ('3'hex). No relevant data is received. The operation is therefore normally followed by a DROP to flush the data from the stack.

2.5 Timer/Counter Module (TCM)

The TCM consists of two timer/counter blocks (Timer 0 and Timer 1) which can be used separately, or together as a single 16-bit counter/timer (see figures 28 and 30). Each timer can be supplied by various internal or external clock sources. These can be selected and divided under program control using the Timer/Counter Control Register (TCCR), the Timer 0 Control Register (T0CR) and the Timer 1 Control Register (T1CR). Capture and compare registers (T0CA,T1CA,T0CP and T1CP) not only allow event counting, but also the generation of various timed output waveforms including programmable frequencies, modulated melody tones, Pulse Width Modulated (PWM) and Pulse Density Modulated (PDM) output signals. When in one of these signal generation modes, the capture register acts as timer shadow register, the current timer state is freezed whenever read by the CPU. The Timer 0 is further equipped for performing a variety of time measurement operations. In this mode the capture register is used together with the gating logic for performing asynchronous, externally triggered snapshot measurements. These measurements include single input pulse width and period measurements and also dual input phase and positional measurement. The mode configuration is set in the Timer 0 and Timer 1 Mode Registers (T0MO and T1MO).

Each timer represents a single maskable interrupt source (T0INT and T1INT), the priority of which can be configured under program control. A Timer 0 interrupt can be caused by any of three conditions (overflow, compare or end-of-measurement). The associated status register (T0SR) differentiates between these. A status register is not necessary in the Timer 1 as an interrupt is caused only on a compare condition.

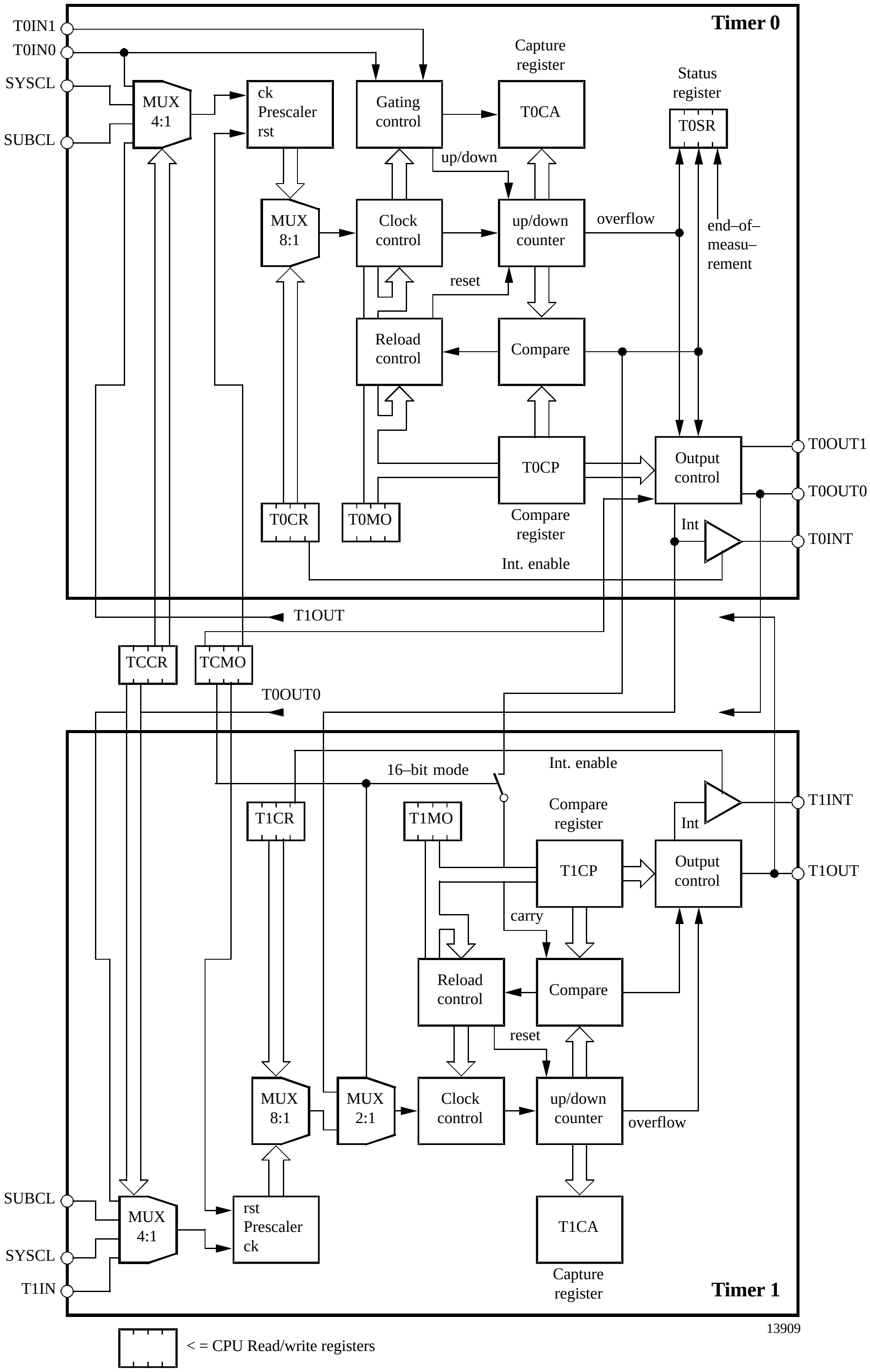


Figure 28. Timer/counter module

2.5.1 General Timer/Counter Control Registers

With the exception of the Timer 0 Interrupt Status Register (T0SR), all the timer/counter registers are indirectly addressed using extended addressing as described in the section “Addressing peripherals”. An overview of all register and subport addresses is shown in table 7. The Timer/Counter auxiliary register (TCSUB) holds the subport address of the particular register about to be accessed.

Care has to be taken to ensure that this subport access sequence is not interrupted. Please refer to the ’HARDC510.SCR’ hardware interface file as a programming guideline.

Timer/Counter Clock Control Register (TCCR)

Subport address (indirect write access): ’6‘hex of Port address ’9‘hex

	Bit 3	Bit 2	Bit 1	Bit 0	
TCCR	**T1CL2**	**T1CL1**	**T0CL2**	**T0CL1**	**Reset value: 1111b**

T0CL2, T0CL1 – Timer 0 Clock source select

T1CL2, T1CL1 – Timer 1 Clock source select

Table 15 Timer/Counter Clock Control Register (TCCR)

Code 3 2 1 0	Function	Direction (TDir) BP40*	 TIM1
x x 0 0	Timer 0 clock = SUBCL	out	x
x x 0 1	Timer 0 clock = SYSCL	out	x
x x 1 0	Timer 0 clock = Timer1 output (T1OUT connected internally)	out	x
x x 1 1	Timer 0 clock = T0IN0 (BP40*)	in	x
0 0 x x	Timer 1 clock = SUBCL	x	out
0 1 x x	Timer 1 clock = SYSCL	x	out
1 0 x x	Timer 1 clock = Timer 0 output (T0OUT0 connected internally)	x	out
1 1 x x	Timer 1 clock = TIM1	x	in

* if TCIO0 = low (connects Timer 0 to Port 4)

The Timer/Counter Clock Control Register (TCCR) controls the clock source to both Timer 0 and Timer 1 prescalers. If an external clock source (on BP40 or TIM1) is selected, then the corresponding port direction is automatically switched to input mode (see figure 27).

Note: The TCIO0 bit must be set low for the BP40 external timer/counter access.

Timer/Counter Interrupt Priority Register (TCIP)

The Timer/Counter Interrupt Priority register (TCIP) is used to configure the Timer 0 and Timer 1 interrupt priority levels.

Subport address (indirect write access): ’7’hex of Port address ’9‘hex

	Bit 3	Bit 2	Bit 1	Bit 0	
TCIP	**T1IP2**	**T1IP11**	**T0IP2**	**T0IP1**	**Reset value: 1111b**

T0IP2, T0IP1 – Timer 0 Interrupt Priority code

T1IP2, T1IP1 – Timer 1 Interrupt Priority code

Table 16 Timer/Counter Interrupt Priority Register (TCIP)

Code 3 2 1 0	Function
x x 1 1	Timer 0 interrupt priority 1
x x 1 0	Timer 0 interrupt priority 3
x x 0 1	Timer 0 interrupt priority 5
x x 0 0	Timer 0 interrupt priority 7
1 1 x x	Timer 1 interrupt priority 0
1 0 x x	Timer 1 interrupt priority 2
0 1 x x	Timer 1 interrupt priority 4
0 0 x x	Timer 1 interrupt priority 6

Timer/Counter I/O Control Register (TCIOR)

Subport address (indirect write access): ’5’hex of Port adddress ’9‘hex

	Bit 3	Bit 2	Bit 1	Bit 0	
TCIOR	**TCIO3**	**TCIO2**	**TCIO1**	**TCIO0**	**Reset value: 1111b**

TCIO3...0 – Timer / Counter I/0 mode select

Table 17 Timer/Counter I/O Control Register (TCIOR)

Code 3 2 1 0	Function
x x x 1	BP40 – standard port mode
x x x 0	BP40 – Timer 0 clock input (T0IN0) or Timer 0 output (T0OUT0)
x x 1 x	BP41 – standard port mode
x x 0 x	BP41 – Timer 0 gate input (T0IN1) or Timer 0 output (T0OUT1)
x 1 x x	BP42 – standard port mode
x 0 x x	BP42 – Buzzer output (BUZ)
1 x x x	BP43 – standard port mode
0 x x x	BP43 – Buzzer output (NBUZ)

By using the Timer/Counter I/O Control Register (TCIOR) the program can configure the respective Port 4 pins as either standard data I/O ports or as external signal ports for the Timer 0 and Buzzer. The Timer 1 uses a dedicated I/O pin TIM1, whose direction is controlled solely by the TCCR (see figure 29). It should be noted that if a TCIOR bit is set low, then the corresponding port data direction register (P4DDR) bit no longer influences the port direction. In the case of BP40 and BP41, the port direction is then controlled entirely by the timer/counter configuration registers (TCCR,T0MO), while pins BP42 and BP43 become unidirectional buzzer outputs.

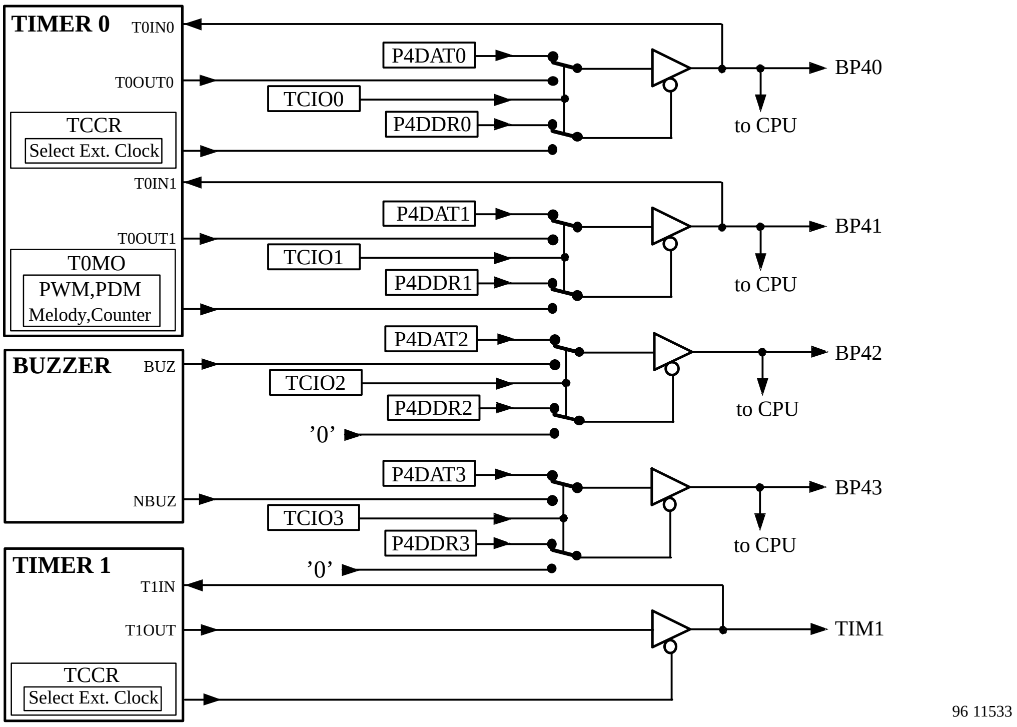


Figure 29. Timer/counter and buzzer external interface

Timer/Counter Mode Register (TCMO)

Subport address (indirect write access): '4'hex of Port address '9'hex

	Bit 3	Bit 2	Bit 1	Bit 0	
TCMO	**T0NINV**	**TC8**	**T1RST**	**T0RST**	**Reset value: 1111b**

T0NINV – Timer 0 output (BP41) appears non-inverted at BP40

TC8 – Timer/Counter in 8-/16-bit mode

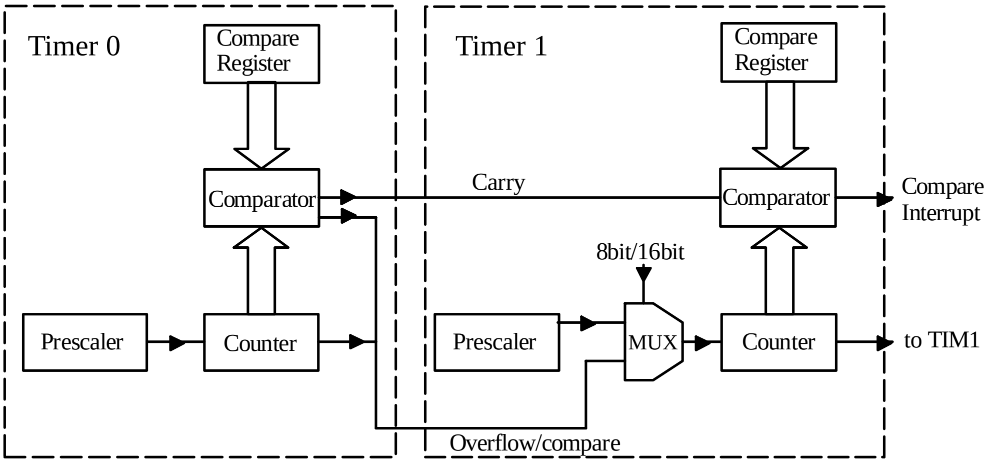
T1STP – Timer 1 Stop/Run

T0STP – Timer 0 Stop/Run

Table 18 Timer/Counter Mode Register (TCMO)

Code 3 2 1 0	Function
x x x 0	Timer 0 running
x x x 1	Timer 0 halted
x x 0 x	Timer 1 running
x x 1 x	Timer 1 halted
x 0 x x	Timer/counter in 16-bit mode
x 1 x x	Timer/counter in 8-bit mode
0 x x x	Inverted output BP41 appears on BP40 (BP40 = NOT BP41)
1 x x x	Non-inverted output BP41 appears on BP40 (BP40 = BP41)

2.5.2 Timer/Counter in 16-bit Mode



96 11549

Figure 30. 16-bit mode

In 16-bit mode, Timer 0 and Timer 1 are cascaded thus forming a 16-bit counter (see figure 30) whereby, irrespective of the state of Timer 0 interrupt mask bit (T0IM), the Timer 1 counts both Timer 0 overflow and compares interrupt events. These are generated according to the state of the Timer 0 Mode Register as described in the T0MO table. The comparators are also cascaded so that when both Timer 0 and Timer 1 match their respective compare registers, the Timer 1 generates both an output signal and a compare interrupt (if unmasked).

In measurement modes, only Timer 0 capture register is loaded with Timer 0's contents on an end-of-measurement event. Timer 1 capture register operates solely as a shadow register. There is no 16-bit capture operation, so the user program must check if Timer 1 has incremented between reading the lower and higher byte. Likewise, there is no automatic suppression of spurious interrupts which could conceivably be generated between writing Timer 0 and Timer 1 compare registers.

2.5.3 Timer 0 Modes

The Timer 0 mode configuration is defined in the Timer 0 Mode Register (T0MO). The available modes and the effect on the Timer 0 interrupt and interrupt flags is shown below. In all modes except the position measurement mode, Timer 0 acts as an up-counter, the related clock frequency being defined by the selected clock source and the prescaler division factor. The counter can be reset and halted at any time by the T0RST bit of the TCMO register which also resets all the interrupt status flags and capture registers. Whenever Port 4 BP40 and BP41 pins are required for Timer 0 I/O, then the appropriate TCIOR enable bit must be set low. In this case, the port direction switching is handled automatically by the hardware. In modes where the BP40 is not used as a timer clock input or as a melody envelope output, the BP40 outputs the same signal as that appearing on BP41. With the help of the T0NINV bit of the Timer/Counter Mode Register (TCMO), the BP41 output can be inverted so that BP40 and BP41 form a differential output stage which can be used for directly driving piezo buzzers or small stepper motors.

Timer 0 Mode Register (T0MO)

Subport address (indirect write access): ’0’hex of Port adress ’9’hex

	Bit 3	Bit 2	Bit 1	Bit 0	
T0MO	**T0MO3**	**T0MO2**	**T0MO1**	**T0MO0**	**Reset value: 1111b**

T0MO3 ... 0 – Timer 0 Mode Code

Table 19 Timer 0 Mode Register (T0MO)

Code 3 2 1 0	Function	Assuming TCIOR1=TCIOR0=low		Interrupt set / T0SR affected		
		BP40 (*3)	BP41	cmp	ofl	eom
0 0 0 0	reserved			–	–	–
0 0 0 1	reserved			–	–	–
0 0 1 0	Modulated melody mode	Envelope (out)	Tone (out)	y/y	y/y	n/n
0 0 1 1	Melody mode	Tone (out)	Tone (out)	y/y	y/y	n/n
0 1 0 0	Counter-auto reload (50% duty cycle)	Toggle (out) /Clock (in)	Toggle (out)	y/y	y/y	n/n
0 1 0 1	Counter-free running (50% duty cycle)	Toggle (out) /Clock (in)	Toggle (out)	n/y	y/y	n/n
0 1 1 0	Pulse density modulation	PDM (out) /Clock (in)	PDM (out)	n/y	y/y	n/n
0 1 1 1	Pulse width modulation	PWM (out) /Clock (in)	PWM (out)	n/y	y/y	n/n
1 0 0 0	Phase measurement	Signal 1 (in)	Signal 2 (in)	n/n	y/y	y/y
1 0 0 1	Position measurement	Signal 1 (in)	Signal 2 (in)	(*1)	(*2)	n/n
1 0 1 0	Low pulse width measurement	Clock (in)	Signal (in)	n/y	y/y	y/y
1 0 1 1	High pulse width measurement	Clock (in)	Signal (in)	n/y	y/y	y/y
1 1 0 0	Counter- auto reload (strobe)	Strobe (out) /Clock (in)	Strobe (out)	y/y	y/y	n/y
1 1 0 1	Counter-free running (strobe)	Strobe (out) /Clock (in)	Strobe (out)	n/y	y/y	n/y
1 1 1 0	Period measurement (rising edge)	Clock (in)	Signal (in)	n/y	y/y	y/y
1 1 1 1	Period measurement (falling edge)	Clock (in)	Signal (in)	n/y	y/y	y/y

*1 **Note:** The compare interrupt/status flag can only be set when counting up.

*2 **Note:** The overflow interrupt/status flag is set on both an overflow or an underflow.

*3 **Note:** The BP40 signals can be inverted if T0NINV=0 (TCMO register)

Timer 0 Interrupt Status Register (T0SR)

Auxiliary register address (read access): ’9’hex

	Bit 3	Bit 2	Bit 1	Bit 0	
T0SR	**not used**	**T0EOM**	**T0OFL**	**T0CMP**	**Reset value: x000b**

Note: The status register is reset automatically when read and also when Timer 0 is reset.

T0EOM– Timer 0 End Of Measurement status flag

T0OFL – Timer 0 OverFLow status flag

T0CMP – Timer 0 CoMPare status flag

Table 20 Timer 0 Interrupt Status Register (T0SR)

Code 3 2 1 0	Function
x x x 1	Timer 0 compare has occurred (Timer 0 = T0CP)
x x 1 x	Timer 0 overflow or underflow has occurred
x 1 x x	Timer 0 measurement completed

The interrupt flags will be set whenever the associated condition occurs irrespective of whether the corresponding interrupt is triggered. Therefore, the status flags are still set if the interrupt condition occurs when the interrupt is masked. To see exactly when the flags are set, see T0MO control code table 18.

Reading from the timer/counter auxiliary register will access the Timer 0 Interrupt Status Register (T0SR).

Timer 0 Control Register (T0CR)

The T0CR is responsible for the predivision of the selected Timer 0 input clock (see TCCR). It can be divided or used directly as clock for the up/down counter. Bit 0 is the mask bit for the Timer 0 interrupt.

Subport address (indirect write access): ’1’hex of Port address ’9‘hex

	Bit 3	Bit 2	Bit 1	Bit 0	
T0CR	**T0FS3**	**T0FS2**	**T0FS1**	**T0IM**	**Reset value: 1111b**

T0FS3 ... 1 – Timer 0 prescaler division factor code

T0IM – Timer 0 Interrupt Mask

Table 21 Timer 0 Control Register (T0CR)

Code 3 2 1 0	Function
x x x 1	Timer 0 interrupt disabled
x x x 0	Timer 0 interrupt enabled
0 0 0 x	Timer 0 prescaler divide by 256
0 0 1 x	Timer 0 prescaler divide by 128
0 1 0 x	Timer 0 prescaler divide by 64
0 1 1 x	Timer 0 prescaler divide by 32
1 0 0 x	Timer 0 prescaler divide by 16
1 0 1 x	Timer 0 prescaler divide by 8
1 1 0 x	Timer 0 prescaler divide by 4
1 1 1 x	Timer 0 prescaler bypassed

Timer 0 Compare Register (T0CP) – Byte Write

Subport address (indirect write access): ’9’hex of Port address ’9‘hex

		Bit 3	Bit 2	Bit 1	Bit 0	
T0CP	**First write cycle**	**T0CP3**	**T0CP2**	**T0CP1**	**T0CP0**	**Reset value: xxxxb**
		Bit 7	Bit 6	Bit 5	Bit 4	
	Second write cycle	**T0CP7**	**T0CP6**	**T0CP5**	**T0CP4**	**Reset value: xxxxb**

T0CP3 ... T0CP0 – Timer 0 Compare Register Data (low nibble) – first write cycle

T0CP7 ... T0CP4 – Timer 0 Compare Register Data (high nibble) – second write cycle

The compare register T0CP is 8-bit wide and must be accessed as byte wide subport (see section ”Addressing Peripherals). First of all, the data is written low nibble and is then followed by the high nibble. Any timer interrupts are automatically suppressed until the complete compare value has been transferred.

Timer 0 Capture Register (T0CA) – Byte Read

Subport address (indirect read access): ’9’hex of Port address ’9‘hex

		Bit 7	Bit 6	Bit 5	Bit 4	
T0CA	**First read cycle**	**T0CA7**	**T0CA6**	**T0CA5**	**T0CA4**	**Reset value: xxxxb**
		Bit 3	Bit 2	Bit 1	Bit 0	
	Second read cycle	**T0CA3**	**T0CA2**	**T0CA1**	**T0CA0**	**Reset value: xxxxb**

T0CA7. .. T0CA4 – Timer 0 Capture Register Data (high nibble) – first read cycle

T0CA3 ... T0CA0 – Timer 0 Capture Register Data (low nibble) – second read cycle

Note: If the timer is read (in PDM mode only) the bit order will appear reversed, so that T0CA0 =MSB, T0CA1=MSB-1 .... T0CA6=LSB+1, T0CA7 = LSB.

The 8-bit capture register T0CA is read as byte wide subport. Note, however, unlike the writing to the compare register, the high nibble is read first followed by the low nibble. The 8-bit timer state is captured on reading the first nibble and held until the complete byte has been read. During this transfer, the timer is free to continue counting.

Timer 0 Free Running Counter Modes (Strobe and 50% Duty Cycle)

In the free running counter mode, Timer 0 can be used as an event counter for summing external event pulses on BP40, or as a timer with an internal time-based clock. When enabled, the counter will count up generating an output signal on BP41 whenever the counter contents match the compare register (see figure 31). This signal can appear either as a strobe pulse or as a simple toggling of the output state (50% duty cycle) depending on the timer mode. Interrupts (if not masked) are generated every 256 clocks on the overflow condition. The current counter state can be read at any time by reading the capture register,. The compare register has no effect on the counter cycle time and will not influence interrupts.

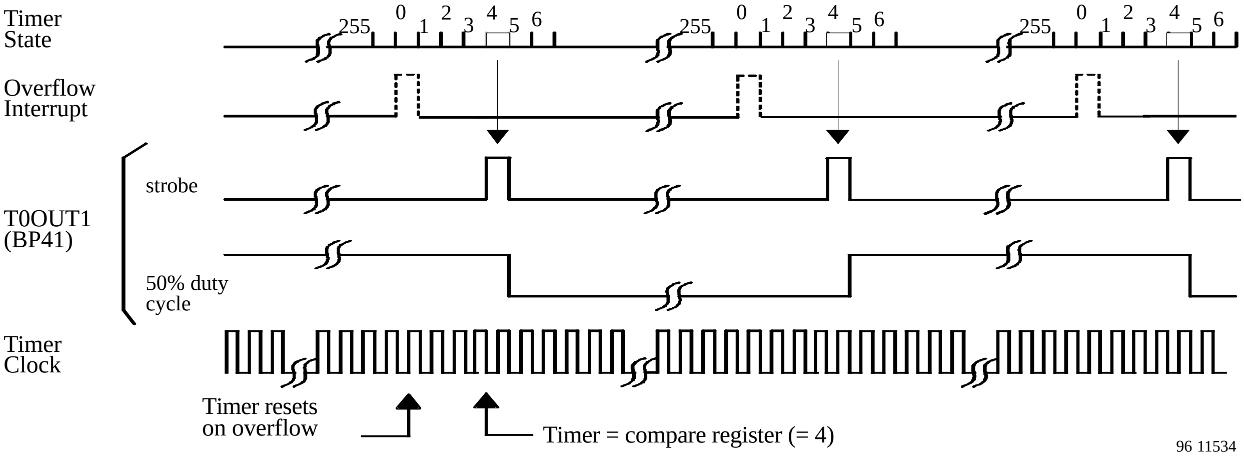


Figure 31. Timer 0 free running counter mode

Timer 0 Counter Reload Modes (Strobe and 50% Duty Cycle)

As in the free running mode, the counter can also be clocked from either an external signal on BP40 or from an internal clock source. In this mode, the counter repetition period is completely defined by the contents of the compare register (T0CP) (see figure 32). The counter counts up with the selected clock frequency. When it reaches the value held in the compare register, the counter then returns to the zero state. At the same time, depending on the selected timer mode, the BP41 either toggles or generates a strobe pulse. If the Timer 0 interrupt is unmasked, a compare interrupt is also generated.

The resultant output frequency $f_{OUT} = f_{IN}/2*(n+1)$ where
n = compare value (n = 1 – 255).

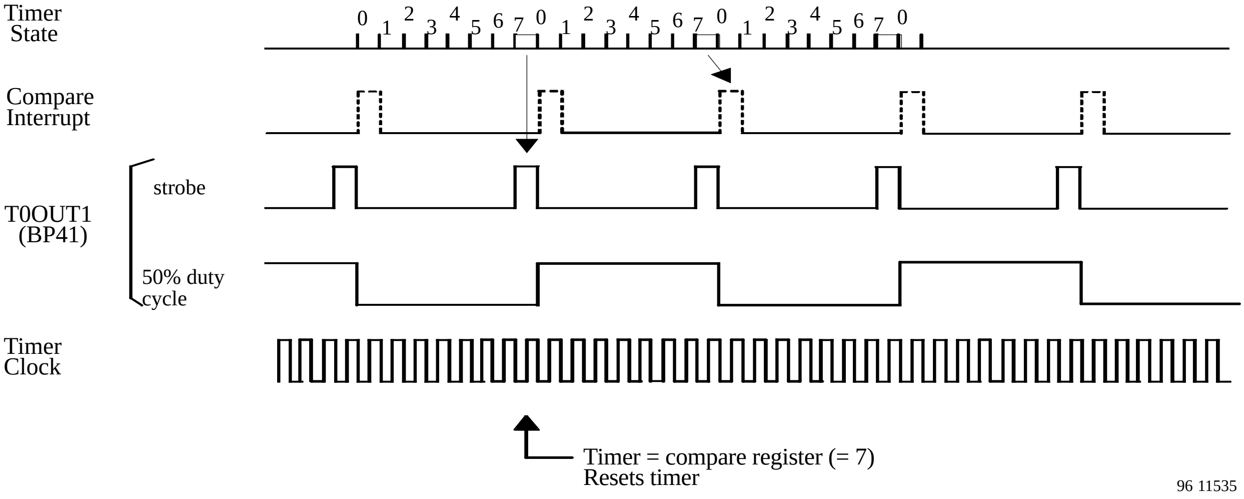


Figure 32. Timer 0 counter reload mode

Melody Mode (with/without Modulation)

The non-modulated melody mode is identical to the auto- reload counter (50% duty cycle) mode. The melody tone frequency appearing on BP41 and/or BP40 is determined in exactly the same way as the value written into the comparator register. In the modulated melody mode, the T48C510 generates two output signals, a melody tone and an envelope pulse (see figure 33). The tone frequency output on BP41 is generated in exactly the same way as in the simple melody mode. While the envelope pulse on BP40 is a single pulse, of a clock period in duration which appears shortly after loading the compare value into the compare register. In this mode, an analog switch is activated between the BP40 and BP41 outputs (see figure 34). With the external capacitor connected, the resultant signal on BP41 exhibits a melody chime effect with an exponential decay.

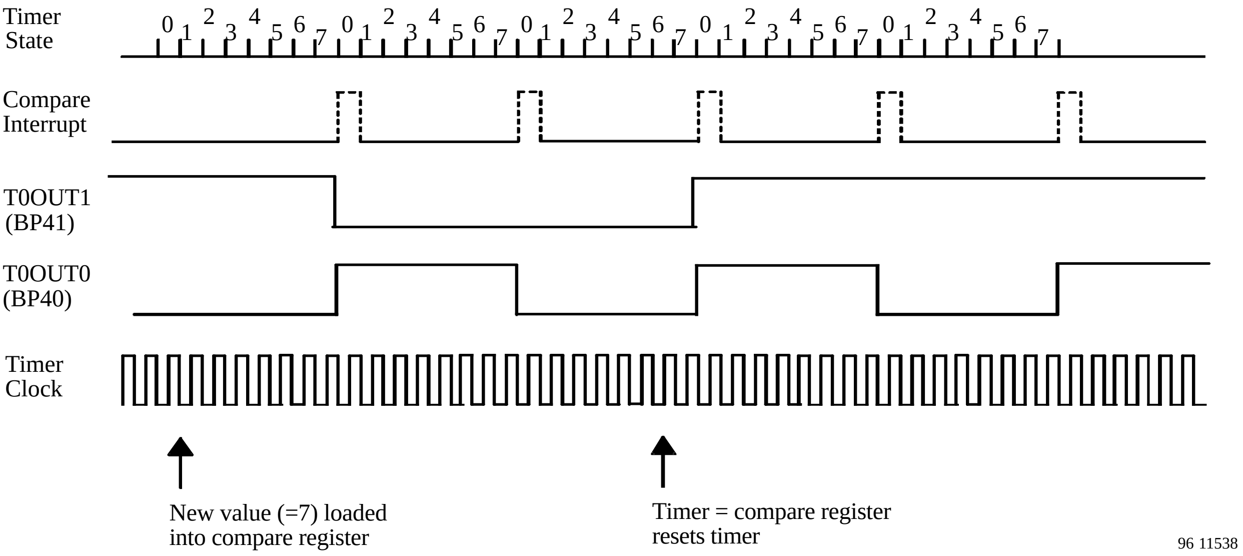


Figure 33. Modulated melody mode

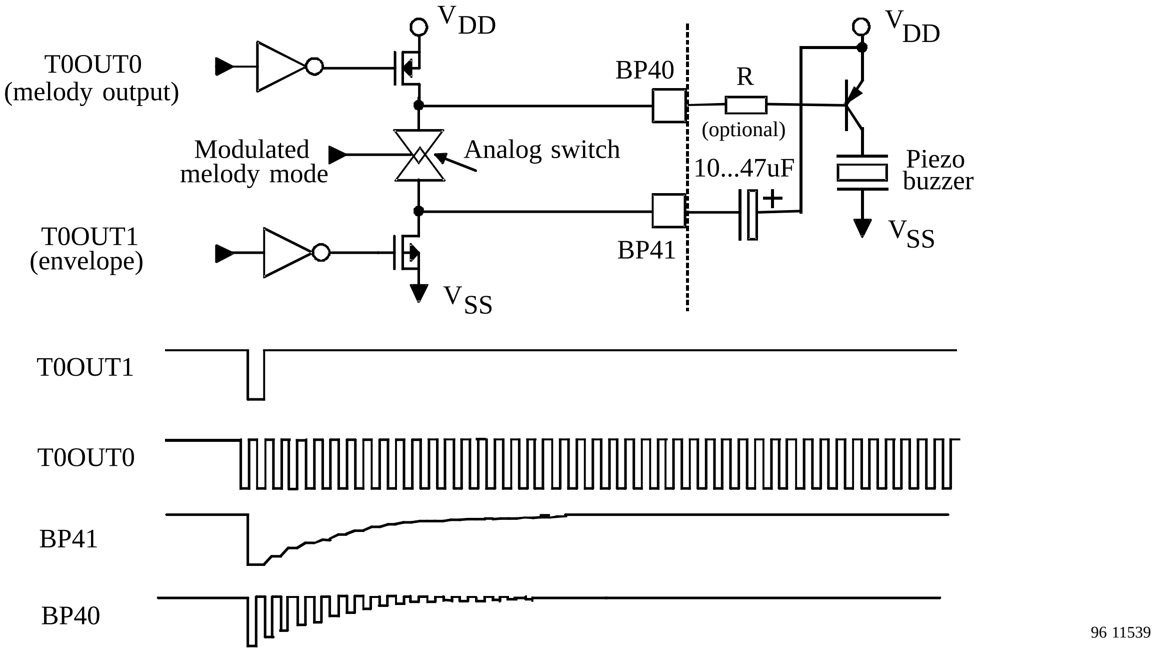


Figure 34. Modulated melody output circuit

Timer 0 Pulse Width Modulation Mode

A pulse width modulated (PWM) signal exhibits a fixed repetition frequency and a variable mark space ratio. It is often used as a simple method for D/A conversion, where the high period is proportional to the digital value to be converted. Therefore by connecting a simple low-pass RC network to the PWM signal, the analog value can be retrieved.

Timer 0 generates the PWM signal by comparing the state of the free running up counter with the contents of the compare register (see figure 35). If the result is less than the compare register value, then the BP41 output is high. If the result is greater or equal to the compare register value, then the BP41 output is set low. Thus, the high phase of the PWM signal is directly proportional to the compare register contents. A total of 256 possible discrete mark space ratios can be generated ranging from a continuous low signal over a variable pulse width signal to a continuous high signal. The PWM signal has a repetition period of 256 clocks, an interrupt (if unmasked) being generated on every overflow event. Care should be taken if the SYSCL clock is used as the PWM clock source because it may stop if the CPU goes into SLEEP mode (see Section 1.5.4 Power-Down Modes).

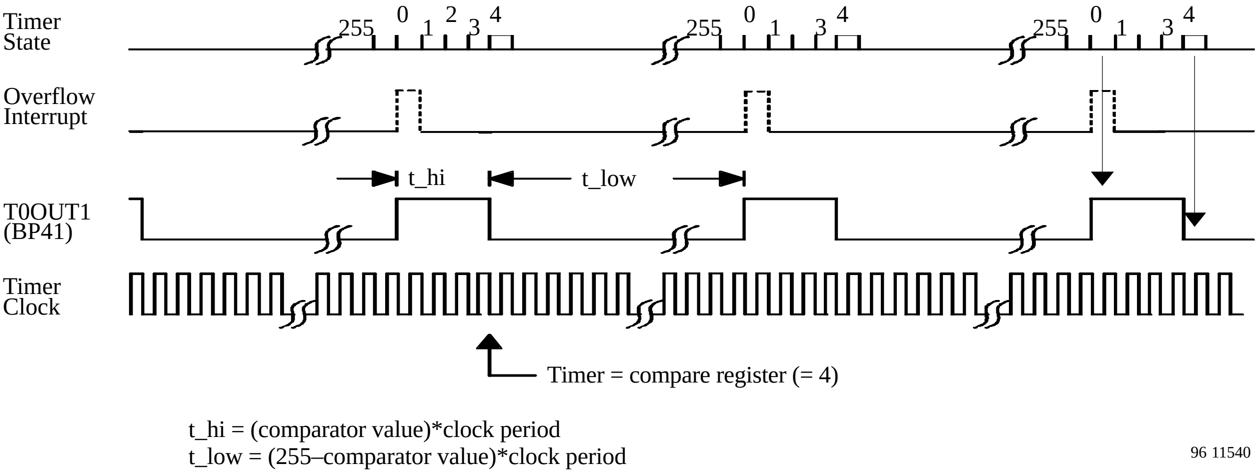

Timer State
255
0
1
2
3
4
Overflow Interrupt
t_hi
t_low
T0OUT1 (BP41)
Timer Clock
Timer = compare register (= 4)
t_hi = (comparator value)*clock period
t_low = (255–comparator value)*clock period
96 11540

Figure 35. Timer 0 pulse width modulation

Pulse Density Modulation Mode

Pulse density modulation (PDM) is also used for simple D/A conversion. Unlike the PWM signal,where the high and low signal phases are always continuous during a single repetition cycle, the PDM distributes these evenly as a series of pulses (see figure 36). This has the advantage that, if used together with an RC smoothing filter for D/A conversion, either the ripple is less than the PWM, or, for a corresponding ripple error, the filter components can be smaller or the clock frequency lower. To generate the PDM output on BP41, the pulse density is controlled by the contents of the compare register in the same way as the PWM generation. Each of the pulses has a width equal to the counter clock period.

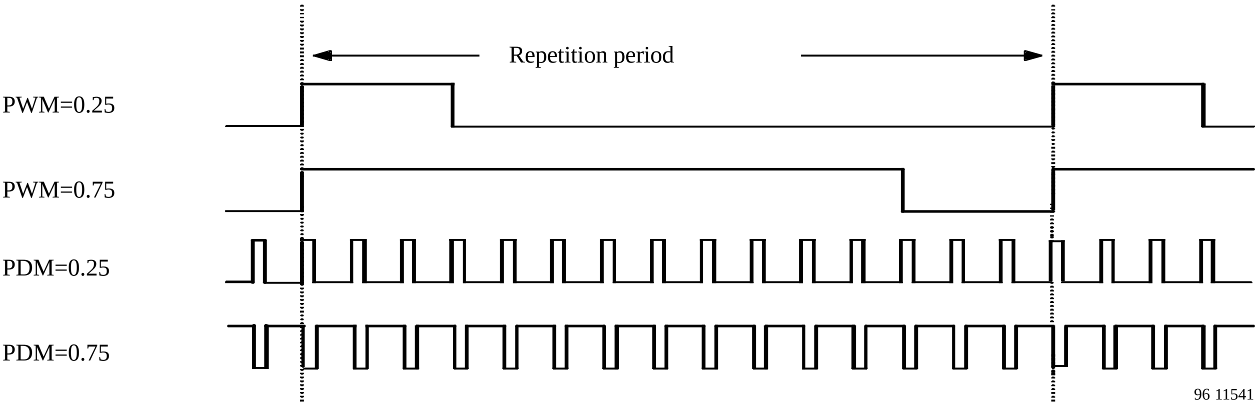


Figure 36. An example 4-bit PWM/PDM comparison

Period Measurement Modes (Rising and Falling Edge)

During the period measurement mode, the counter counts the number of either internal or external clocks in one period of the BP41 input signal (see figure 37). Dependent on the mode chosen, this will be from rising edge to the next rising edge or conversely, falling edge to the following falling edge. On the trigger edge, the counter state is loaded into the capture register and subsequently reset. The measured value remains in the capture register until overwritten by the following measured value. Interrupts can be generated by either an overflow condition or an end-of-measurement (eom) event. An 'eom' event signals the CPU that a new measured value is present in the capture register and can be read, if required.

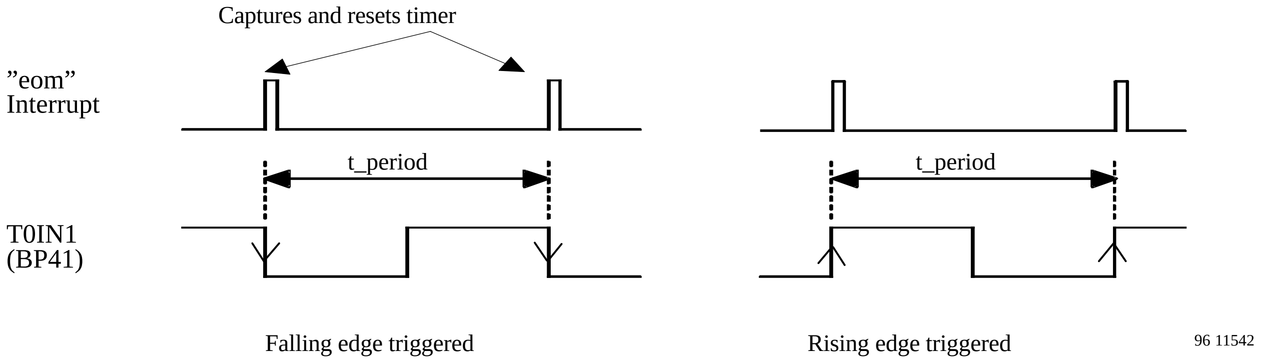


Figure 37. Period measurement

Pulse Width Measurement Modes (High and Low)

In this mode, the selected clock source is gated to the counter for the duration of each input pulse received on BP41 (see figure 38). Whether the measurement takes place during the high or low phase depends on the selected mode. At the end of each pulse, the counter state is loaded into the capture register and subsequently reset. Interrupts can be generated by either an overflow condition or an end-of-measurement (eom) event. An 'eom' event signals the CPU that a new measured value is present in the capture register can be read, if required.

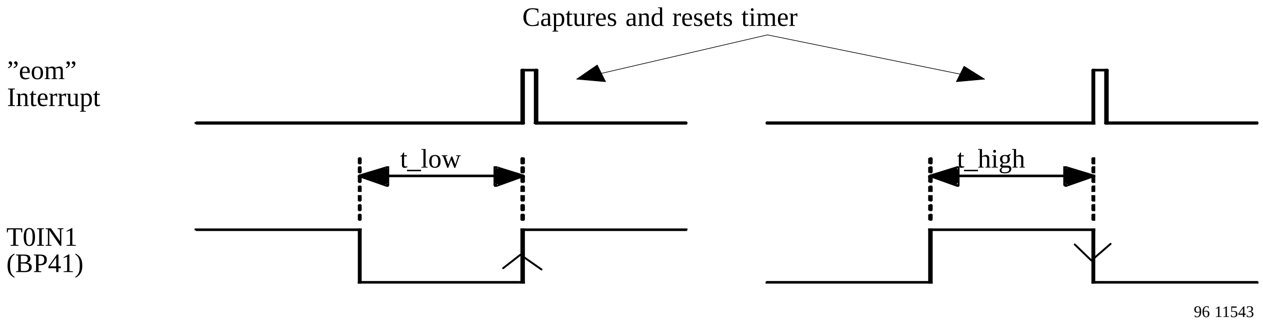


Figure 38. Pulse width measurement

Phase Measurement Mode

This mode allows the Timer 0 to measure the phase misalignment between two 1:1 mark space ratio input signals connected to the BP40 and BP41 pins (see figure 39). The counter clock is gated with the phase misalignment period (tp), during which time the counter increments with the selected clock frequency. This misalignment period is defined as the period during which BP40 is high and BP41 is low. Capturing and resetting of the counter always takes place on the rising edge of BP41. The measured value remains in the capture register until overwritten by the next measurement. Interrupts can be generated by either an overflow condition or an end-of-measurement ('eom') event. An 'eom' event signals the CPU that a new measured value is present in the capture register and can be read, if required.

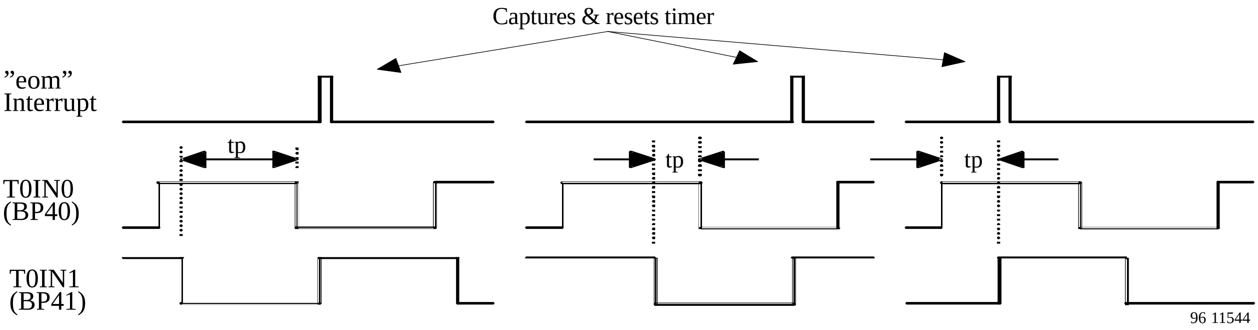


Figure 39. Phase measurement

Position Measurement Mode

This mode is intended for the evaluation of positional sensors with biphase output signals. Figure 40 illustrates a typical positional sensor system which delivers both incremental positional stepping signals and also directional information. The direction can be deduced from the relative phase of the two signals. Therefore if BP40 is high on the rising edge of BP41, the moving mask travels to the left and if it is low then it travels to the right. The direction (left/right) information is used to set the direction of the up/down counter which enables the BP40 pulses to be counted. Assuming that the system has been reset on a reference position, the counter will always hold the absolute current position of the moving mask. This can be read by the CPU if necessary. This mode is the only one in which the counter is allowed to decrement. Therefore, in this case it is possible for both an underflow or an overflow to occur. The overflow interrupt (if unmasked) will trigger on either of these conditions while the compare interrupt on the other hand will only trigger if the counter is counting upwards. To differentiate between an overflow or underflow, the compare value can be set to '0' hex, for example. An overflow would then set both the overflow and compare status flags while an underflow sets the overflow status flag only.

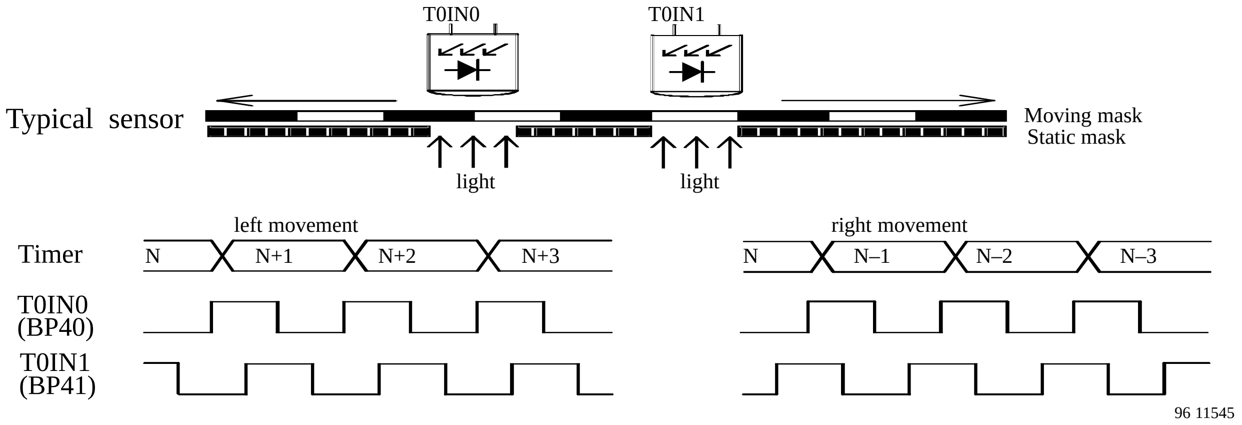


Figure 40. Position measurement mode

2.5.4 Timer 1 Modes

The Timer 1 is aimed at performing event counting and timing functions (see figure 28). It has, unlike the Timer 0, no gated clock or externally triggered capture modes. The counter counts up with an internal or external clock, depending on the state of the Timer 1 Control Register (T1CR) and the Timer/Counter Clock Control Register (TCCR) and generates a compare interrupt whenever the counter matches the Timer 1 compare register. This is the only Timer 1 interrupt source. Masking can be performed using the mask bit in the Timer 1 Control Register (T1CR) and priority can be defined in the Timer/Counter Interrupt Priority Register (TCIP). The TIM1 pin is used by the Timer 1 either as clock/event input or timer output. I/O control of the Timer 1 pin TIM1 is controlled entirely by the hardware, therefore if the TIM1 is selected as an external clock or event source (in the TCCR), there can be no Timer 1 signal output. In this case, the timer would be used solely to generate interrupts.

In autostop operation, the Timer 1 will halt both itself and Timer 0 whenever the Timer 1 compare value is reached. This feature can be used for example to generate an exact burst of pulses. Both timers will remain stopped until restarted. Restarting is performed in the normal way by setting the appropriate control bits in the Timer/Counter Mode Register (TCM0).

Timer 1 Mode Register (T1MO)

Subport address (indirect write address): ’2’hex of Port address ’9‘hex

	Bit 3	Bit 2	Bit 1	Bit 0	
T1MO	**T1MO3**	**T1MO2**	**T1MO1**	**T1MO0**	**Reset value: 1111b**

T1MO3 ... 0 – Timer 1 Mode Control

Table 22 Timer 1 Mode Register (T1MO)

Code 3 2 1 0	Function	Compare Interrupt
x x 0 0	Counter free running (50% duty cycle)	yes
x x 0 1	Counter auto reload (50% duty cycle)	yes
x x 1 0	Pulse width modulation	yes
x x 1 1	Counter auto-reload (strobe output)	yes
x 0 x x	Increment on falling edge of clock	–
x 1 x x	Increment on rising edge of clock	–
1 x x x	Normal operation (no autostop)	yes
0 x x x	Autostop operation (Timer 1 stops Timer 2)	yes

Timer 1 Control Register (T1CR)

The T1CR is responsible for the predivision of the selected Timer 1 input clock (see TCCR). It can be divided or used directly as clock for the up counter. Bit 0 is the mask bit for the Timer 1 interrupt.

Subport address (indirect write access): ’3’hex of Port adress ’9‘hex

	Bit 3	Bit 2	Bit 1	Bit 0	
T1CR	**T1FS3**	**T1FS2**	**T1FS1**	**T1IM**	**Reset value: 1111b**

T1FS3 ... 1 – Timer 1 Prescaler Division Factor Code

T1IM – Timer 1 Interrupt Mask

Table 23 Timer 1 Control Register (T1CR)

Code 3 2 1 0	Function
x x x 1	Timer 1 interrupt disabled
x x x 0	Timer 1 interrupt enabled
0 0 0 x	Timer 1 prescaler divide by 256
0 0 1 x	Timer 1 prescaler divide by 128
0 1 0 x	Timer 1 prescaler divide by 64
0 1 1 x	Timer 1 prescaler divide by 32
1 0 0 x	Timer 1 prescaler divide by 16
1 0 1 x	Timer 1 prescaler divide by 8
1 1 0 x	Timer 1 prescaler divide by 4
1 1 1 x	Timer 1 prescaler bypassed

Timer 1 Compare Register (T1CP) – Byte Write

Subport address (indirect write access): ’8’hex of Port address ’9‘hex

		Bit 3	Bit 2	Bit 1	Bit 0	
T1CP	**First write cycle**	**T1CP3**	**T1CP2**	**T1CP1**	**T1CP0**	**Reset value: xxxxb**
		Bit 7	Bit 6	Bit 5	Bit 4	
	Second write cycle	**T1CP7**	**T1CP6**	**T1CP5**	**T1CP4**	**Reset value: xxxxb**

T1CP3 ... T1CP0 – Timer 1 Compare Register Data (low nibble) – first write cycle

T1CP7. .. T1CP4 – Timer 1 Compare Register Data (high nibble) – second write cycle

The compare register T1CP is 8 bits wide and must be accessed as byte wide subport (see section “Addressing Peripherals”). The data is written low nibble first, followed by high nibble. Any timer interrupts are automatically suppressed until the complete compare value has been transferred.

Timer 1 Capture Register (T1CA) – Byte Read

Subport address (indirect read access): ’8’hex of Port address ’9‘hex

		Bit 7	Bit 6	Bit 5	Bit 4	
T1CA	**First read cycle**	**T1CA7**	**T1CA6**	**T1CA5**	**T1CA4**	**Reset value: xxxxb**
		Bit 3	Bit 2	Bit 1	Bit 0	
	Second read cycle	**T1CA3**	**T1CA2**	**T1CA1**	**T1CA0**	**Reset value: xxxxb**

T1CA7 ... T1CA4 – Timer 1 Capture Register Data (high nibble) – first read cycle

T1CA3 ... T1CA0 – Timer 1 Capture Register Data (low nibble) – second read cycle

The 8-bit capture register T1CA is read as byte-wide subport. Note, however, unlike the writing to the compare register, the high nibble is read first followed by low nibble. The 8-bit timer state is captured on reading the first nibble and held until the complete byte has been read. During this transfer, the timer is free to continue counting. The previous capture value will be held until the timer is restarted again.

Timer 1 Counter Free Running (50% Duty Cycle)

In the free running counter mode, the counter counts up with either an internal or external clock and cycles through all 256 timer states. On the clock following a match between the compare register (T1CR) and the counter, a compare interrupt (if unmasked) is generated and the TIM1 pin is toggled (see figure 40).

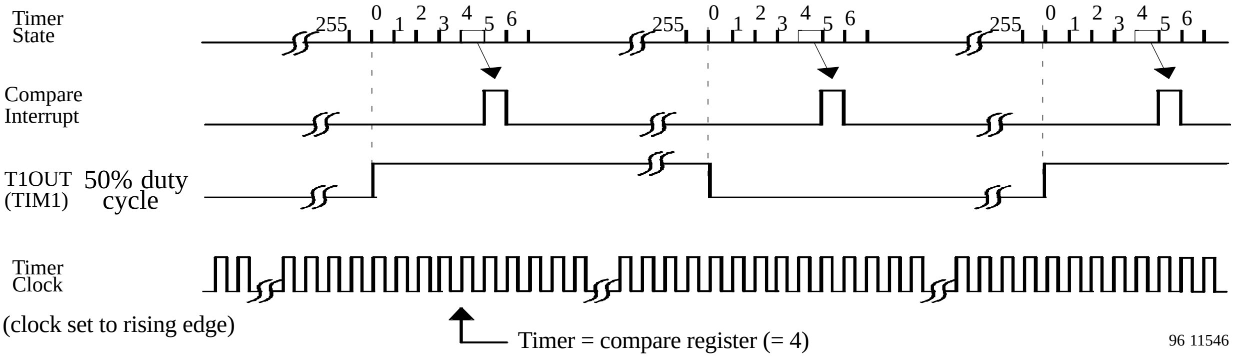


Figure 41. Timer 1 counter free running (50% duty cycle)

Timer 1 Counter Auto Reload (Strobe and 50% Duty Cycle)

In the auto-reload mode, the counter counts up with either an internal or external clock. On the clock cycle following a match between the compare register (T1CR) and the counter, a compare interrupt (if unmasked) is generated. The TIM1 output is either strobed or toggled and the counter reset (see figure 42). Therefore, the counter cycle period is defined by the contents of the compare register. In 50% duty cycle mode the frequency of TIM1 is:

$f_{TIM1} = f_{in}/2(n+1)$ where the compare value (n) =1 ... 255.

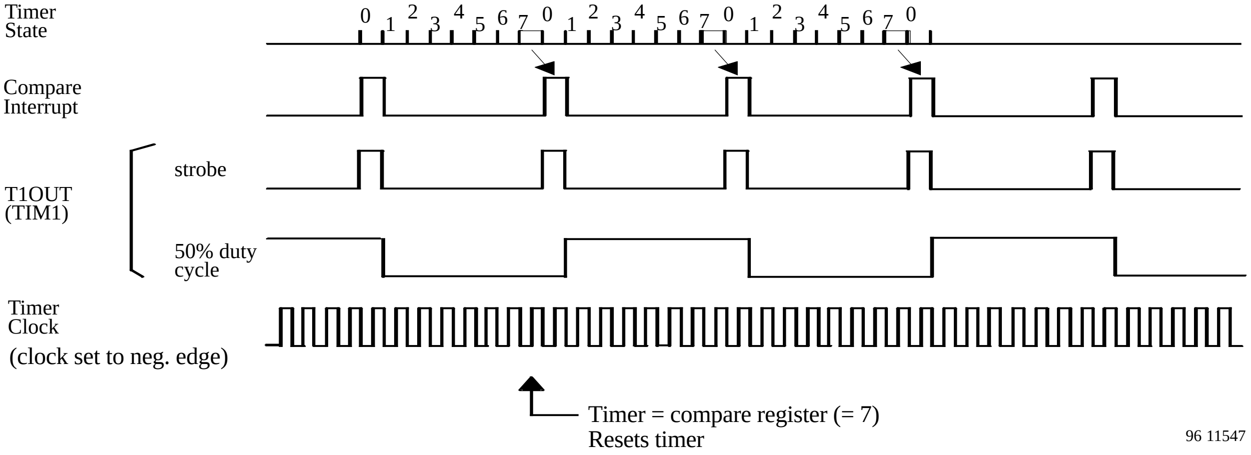


Figure 42. Timer 1 counter auto reload

Timer 1 Pulse Width Modulation

The Timer 1 generates the PWM signal by comparing the state of the free running up counter with the contents of the compare register (see figure 43). If the result is less or equal to the compare register value, then the TIM1 output is high. If the result is greater than the compare register value, then the TIM1 output is set low. Thus, the high phase of the PWM signal is directly proportional to the compare register contents. A total of 256 possible discrete mark space ratios can be generated ranging from a continuous low signal over a variable pulse width signal. The PWM signal has a repetition period of 256 clock periods, an interrupt (if unmasked) being generated on every compare event.

Care should be taken if SYSCL is used as the PWM clock source. The PWM output may stop if the CPU goes into SLEEP depending on the programming of the NSTOP bit in the CM-register. Using this mode of operation recommends to set the bit NSTOP =1.

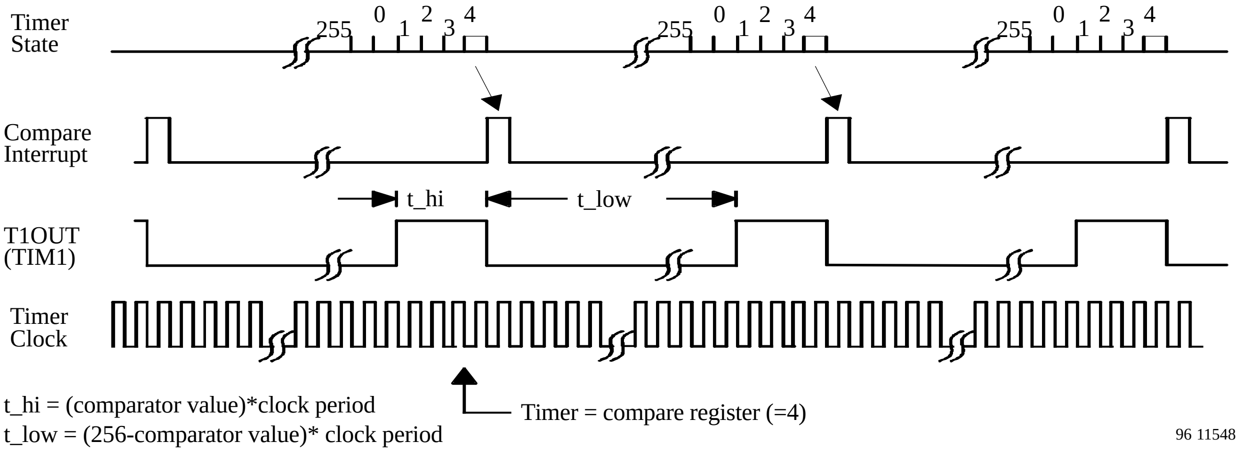

Timer State
255
0
1
2
3
4
Compare Interrupt
t_hi
t_low
T1OUT (TIM1)
Timer Clock
t_hi = (comparator value)*clock period
t_low = (256-comparator value)* clock period
Timer = compare register (=4)
96 11548

Figure 43. Timer 1 pulse width modulation

2.6 Buzzer Module

The buzzer is a 4 stage frequency divider which divides the SUBCL and depending on the state of the Buzzer Control Register (BZCR) can output one of four frequencies. An external piezo or buzzer can be driven by the complementary buzzer outputs (BUZ and NBUZ) which are directed to Port 4 (BP42 and BP43) under control of the Timer/Counter I/O Register (TCIOR) as shown in figure 28. When the buzzer is switched off, both of the buzzer outputs take up the same logical state. This is controlled by the BZOP bit of the BZCR.

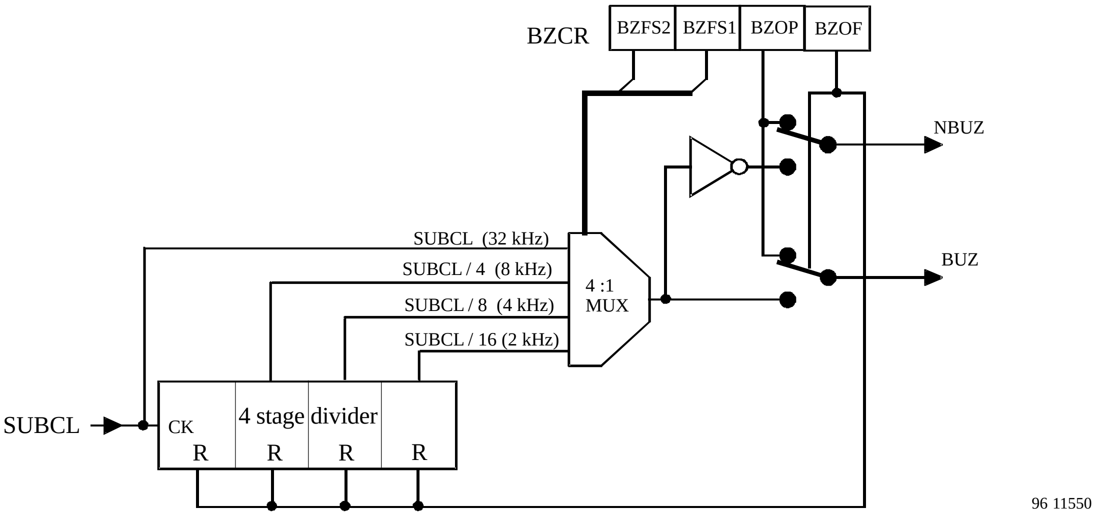


Figure 44. Buzzer module

Buzzer Control Register (BZCR)

Subport address (indirect write access): ’A‘hex of Port adress ’9‘hex

	Bit 3	Bit 2	Bit 1	Bit 0	
BZCR	**BZFS2**	**BZFS1**	**BZOP**	**BZOF**	**Reset value: 1111b**

BZFS2, BZFS2 – Buzzer Frequency Select code

BZOP – Buzzer Output Stop State

BZOF – Buzzer off/on

Table 24 Buzzer Control Register (BZCR)

Code 3 2 1 0	Function
x x x 0	Buzzer on
x x x 1	Buzzer off
x x 0 x	Buzzer output stop state: BP42 = BP43 = low
x x 1 x	Buzzer output stop state: BP42 = BP43 = high
0 0 x x	Buzzer frequency: 32 kHz (= SUBCL)
0 1 x x	Buzzer frequency: 8 kHz (= SUBCL / 4)
1 0 x x	Buzzer frequency: 4 kHz (= SUBCL / 8)
1 1 x x	Buzzer frequency: 2 kHz (= SUBCL / 16)

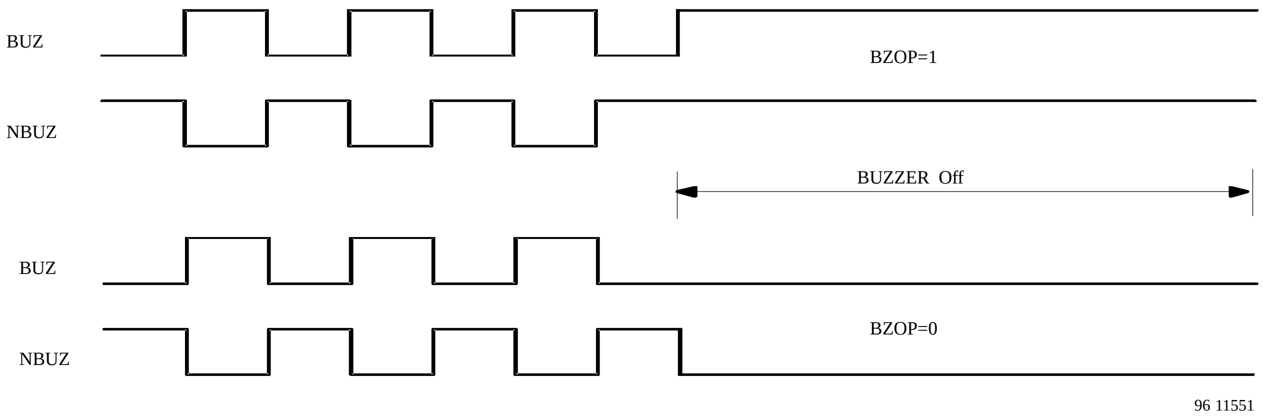


Figure 45. Buzzer waveform

2.7 MTP Programming

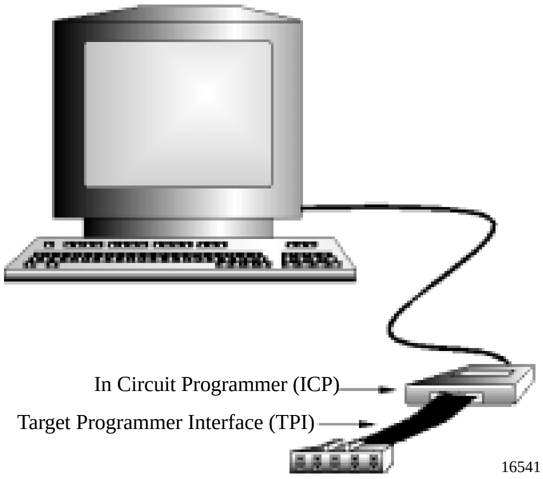

In Circuit Programmer (ICP)
Target Programmer Interface (TPI)
16541

Figure 46. Programmer System

To accomodate the application program and the associated hardware option configuration, the T48C510 is equipped with 2 on-chip EEPROM memory blocks. These are written via a 6-signal Target Programmer Interface (TPI), comprising of 2 power lines (VDD and VSS), a Programm Mode signal (PM) and 3 data lines which are multiplexed onto 3 of the T48C510 functional pins – BP00, BP01 and BP02 (see figure 47). For setting up the required hardware options and downloading these along with the application program into the T48C510, the customer can be supplied with a dedicated PC based programmer software operating under Windows95/98 or NT and an In Circuit Programmer unit (ICP). The ICP is connected to the PC via a standard PC serial interface port and to the target device or application board (for in system programming) via the TPI flat band cable.

Table 25 Target programmer interface signals

TPI Connector Pin	Pin Name	T48C510 Function
1	PM	Programming Mode Input
2	VDD	+5 volt Supply
3	BP02	Port02 (Clock) Input
4	BP01	Port01 (Data) Input
5	BP00	Port00 (Data) Output
6	VSS	Ground Supply
7	n/c	not connected
8	n/c	not connected
9	n/c	not connected
10	n/c	not connected

The state of the T48C510 PM pin defines the MTP operational mode ie. PM = high (Program Mode), PM = low (Normal operation Mode) while the 3 TPI data lines are used to serially load or read the customer's data into or out of the T48C510.

Application Program

The Programmer software requires only the customer's binary *.hex file which is generated by the MARC4 program compiler and also provides the primary data base for emulation. This is displayed on the screen as an editable hexadecimal memory map. Contents of an already programmed device can be read back and displayed an the same hex. form provided that the device's "Read Lock" has not been set. A "Read Lock" Protected device, if read will appear to be full of F hex .

Hardware Configuration

All hardware configurations are set up within the software's intuitive user interface by selecting the required options from the masks provided. The available configurable hardware options are similar to those of the M44C510E (see "Hardware Options" section). These effect primarily port configurations, watchdog and coded reset settings. The port driver strengths, although mask programmable in the M44C510E are not configurable in the MTP – all output drivers being internally "hardwired" to the default "standard drive" strength.

Read Lock Protection

The programmer software encorporates a so called "Read Lock" which can be set by the user. This is provided for customer security purposes and inhibits the reading of the customer's Application Program by unauthorized persons. If set, the "Read Lock" sets a hardware key in the MTP EEPROM which disables reading of the Program/ Configuration data. It should be noted that this is a "Read Lock" and not a "Write Lock", so even if the lock is set, it is still possible to overwrite the customer data with new program code.

In-System Programming

For "in-system programming", the application circuit board must be fitted with a 10-pin male connector to accomodate the TPI connector. To ensure conflict-free access to the target T48C510 TPI related Pins (BP00, BP01, BP02 and PM) it is recommended that these are equipped with jumpers (J5, J4, J3 and J1) to avoid signal contention with other on board drivers sources. (see figure 47). However, if these can be overdriven, or if the Port 0 is not used in the application, then the jumpers can be omitted or replaced by isolating resistors. Prior to connecting the TPI, all other application power supply sources should be disconnected from the application circuit board. Should other on board components either present an excessive power supply load or be unable to withstand the ICP 5-Volt supply voltage, then the VDD power line should also be jumpered (J2).

During the programming operation all ports are set into input mode, with the previously programmed pullup/pull-downs transitors deactivated.

In normal operational mode, the PM pin is strapped to ground and the Port 0 reverts to a port function as described in section 2.2.1.

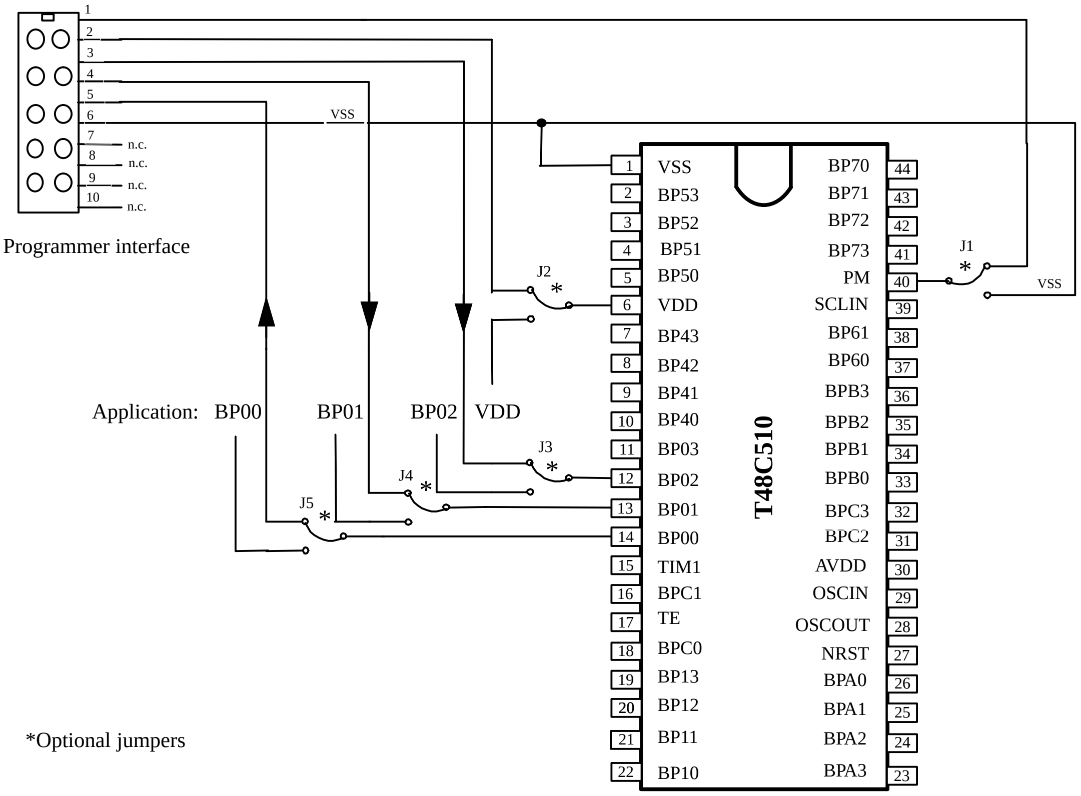


Figure 47. In-system programming

2.8 Noise Considerations

When designing the microcontroller based application, several factors should be taken into consideration to increase noise immunity and reduce electromagnetic emission (EME). Many such potential problems can be avoided by careful layout of the printed circuit board (PCB). The PCB contains many parasitic components which at first sight are not apparent. PCB tracks can act as antennas or as coupling capacitors. Long stretches of parallel tracks and long high frequency signal lines should thus be avoided wherever possible to minimise the chance of picking up or transmitting unwanted signals.

2.8.1 Noise Immunity

The following guidelines will increase system noise immunity:

- Unconnected inputs should not be left open. If port pins are not required then it is recommended to set pullup or pulldown option on these pins.
- Special care should be taken when laying out the PCB that interrupt, reset and clock signal lines are kept short and are carefully shielded or have sufficient spacing from other on board noise generating sources.
- A quartz crystal should always be directly located immediately next to the microcontroller crystal oscillator terminals (OSCIN and OSCOUT), the connections being always very short. This avoids not only signal coupling onto the clock source but can also reduces EME.

- PCB's should where economically possible be equipped with adequate ground planes.

- The microcontroller power supply should be decoupled with an electrolytic capacitance (approx. 10 µF) in parallel with a ceramic capacitance (approx.100 nF) situated as close to the microcontroller device as possible.

2.8.2 Electromagnetic Emission

Electromagnetic emmision is caused by rapidly changing electrical current (dI/dt) in long antenna like connection lines and cables. This can result in electrical interference on other telecommunication devices. These current spikes are more often than not present in the system power supply lines and driver signal lines.

The following guide will help to reduce EME:

- Keep the length of PCB current switching signal tracks to a minimum..
- Adopt a PCB star power routing system connected at one point.
- Many of the microcontroller port outputs can be configured with several drive strengths. This means that a high drive output will switch a signal faster than for example standard drive output. The resulting change in current in the signal and power lines will also increase, causing an increase in EME. So wherever speed and drive current is not necessary the ports should be configured with the lowest drive possible.
- If possible, write the application program to avoid multiple outputs switching at any instant.
- Cables can be equipped with ferrite rings to slow current spikes or the system can be encased in a grounded conducting casing.

3 Electrical Characteristics

3.1 Absolute Maximum Ratings

Voltages are given relative to V_{SS} .

Parameters	Symbol	Value	Unit
Supply voltage	V_{DD}	–0.3 to + 7.0	V
Input voltage (on any pin)	V_{IN}	$V_{SS} -0.3 \leq V_{IN} \leq V_{DD} +0.3$	V
Output short circuit duration	t_{short}	indefinite	s
Operating temperature range	T_{amb}	–40 to +85	°C
Storage temperature range	T_{stg}	–65 to +150	°C
Thermal resistance (SSO44)	R_{thJA}	110	K/W
Soldering temperature ($t \leq 10$ s)	T_{sld}	260	°C

Stresses greater than those listed under absolute maximum ratings may cause permanent damage to the device. This is a stress rating only and functional operation of the device at any condition above those indicated in the operational section of these specification is not implied. Exposure to absolute maximum rating condition for an extended period may affect device reliability. All inputs and outputs are protected against high electrostatic voltages (4kV, HBM) or electric fields. However, precautions to minimize the build-up of electrostatic charges during handling are recommended. Reliability of operation is enhanced if unused inputs are connected to an appropriate logic voltage level (e.g. V_{DD}).

3.2 DC Operating Characteristics

Supply voltage V_{DD} = 5 V, V_{SS} = 0 V, T_{amb} = –40 to 85°C unless otherwise specified.
Typical values relate to V_{DD} = 5 V, T_{amb} = 25°C and are for reference only.

Parameters	Test Conditions / Pins	Symbol	Min.	Typ.	Max.	Unit
Power supply						
Supply voltage		V_{DD}	2.2		6.2	V
Active current	CPU running TestROM @SYSCL_iRC3	I_{DD}		200	500	µA

Parameters	Test Conditions / Pins	Symbol	Min.	Typ.	Max.	Unit
Quotient I_{DD}/SYSCL_iR3	CPU running TestROM @SYSCL_iRC3	I_{DDQ}		0.25	0.5	µA/kHz
Halt current	CPU in sleep mode, NSTOP = 0	I_{Halt}		0.1	0.5	µA
Power-on reset threshold voltage						
POR threshold voltage		V_{POR}	0.8	1.0	1.5	V
Schmitt-trigger input voltage: (all inputs except Port 5, 7 and C)						
Negative-going threshold voltage	V_{DD} = 2.4 to 6.2 V	V_{T-}	V_{SS}		0.4×V_{DD}	V
Positive-going threshold voltage	V_{DD} = 2.4 to 6.2 V	V_{T+}	0.55× V_{DD}		V_{DD}	V
Hysteresis (VT+ – VT–)	V_{DD} = 2.4 to 6.2 V	V_H		0.1×V_{DD}		
Input Pins: NRST and TE						
Input voltage LOW	V_{DD} = 2.4 to 6.2 V	V_{IL}	V_{SS}		0.2 × V_{DD}	V
Input voltage HIGH	V_{DD} = 2.4 to 6.2 V	V_{IH}	0.8 × V_{DD}		V_{DD}	V
Input NRST with pull-up resistor						
Input LOW current	V_{DD} = 2.4 V, V_{IL}= V_{SS} V_{DD} = 5.0 V	I_{IL}	–1.0 –5	–1.5 –10	–3.0 –18	µA µA
Input TE with pull-down resistor						
Input HIGH current	V_{DD} = 5.0 V	I_{IH}	1	1.4	2	mA
All Bidirectional Ports and TIM1						
Input voltage LOW	V_{DD} = 2.4 to 6.2 V	V_{IL}	V_{SS}		0.2 × V_{DD}	V
Input voltage HIGH	V_{DD} = 2.4 to 6.2 V	V_{IH}	0.8 × V_{DD}		V_{DD}	V
Dynamic input LOW current (pull-up)	V_{DD} = 2.4 V, V_{IL}= V_{SS} V_{DD} = 5.0 V	I_{IL}	–1.0 –5	–1.5 –10	–3.0 –18	µA µA
Dynamic input HIGH current (pull-down)	V_{DD} = 2.4 V, V_{IH} = V_{DD} V_{DD} = 5.0 V	I_{IH}	1.0 5	1.5 10	2.5 18	µA µA
Output LOW current	V_{DD} = 2.4 V V_{OL} = 0.2*V_{DD} V_{DD} = 5.0 V	I_{OL}	1 6	2 9	4 13	mA mA
Output HIGH current	V_{DD} = 2.4 V V_{OH} = 0.8*V_{DD} V_{DD} = 5.0 V	I_{OH}	–1 –6	–2 –8	–4 –13	mA mA
Bidirectional Port BP4, BP5, BP7, BPA, BPB and BPC						
Input LOW current Static pull-up (30 kΩ)	V_{DD} = 2.4 V V_{DD} = 5.0 V	I_{IL} I_{IL}	–15 –100	–25 –150	–45 –220	µA µA
Input HIGH current static pull-down (30 kΩ)	V_{DD} = 2.4 V V_{DD} = 5.0 V	I_{IH} I_{IH}	15 100	25 150	45 220	µA µA
Bidirectional Port BP60 and BR61						
Input LOW current static pull-up (4 kΩ)	V_{DD} = 2.4 V V_{DD} = 5.0 V	I_{IL} I_{IL}	–0.2 –1	–0.3 –1.35	–0.5 –2	mA mA
Input HIGH current static pull-down (4 kΩ)	V_{DD} = 2.4 V, V_{IL} = V_{SS} V_{DD} = 5.0 V	I_{IH} I_{IH}	0.15 1	0.25 1.4	0.5 2	mA mA

Note: The total sum of all port static output currents must not exceed 100 mA.
The sum of all port currents switched at any instant (dI/ dt) must not exceed 30 mA.

3.3 AC Characteristics

Supply voltage V_{DD} = 2.4 to 6.2 V, V_{SS} = 0 V, T_{amb} = –40 to 85°C unless otherwise specified.
Typical values relate to V_{DD} = 5 V, T_{amb} = 25°C and are for reference only.

Parameters	Test Conditions / Pins	Symbol	Min.	Typ.	Max.	Unit
Reset timing						
Power-on reset delay	$V_{DD} > V_{POR}$	t_{POR}		80		ms
NRST input LOW time		t_{NRST}	4			µs
Interrupt request input timing						
Int. request LOW time		t_{IRL}	50			ns
Int. request HIGH time		t_{IRH}	50			ns
Internal RC oscillator (for additional characteristics see figures 55 to 57)						
Standby current of iRC0	CPU in SLEEP mode, SC = 0011b, CM = 1100b	I_{iRC0}		300	500	µA
SYSCL_iRC0	CPU active, SC = 0011b, CM = 1100b	f_{SYSCL}	3.5	7.0	10.5	MHz
Standby current of iRC1	CPU in SLEEP mode, SC = 0111b, CM = 1101b	I_{iRC1}		150	250	µA
SYSCL_iRC1	CPU active, SC = 0111b, CM = 1101b	f_{SYSCL}	1.9	3.0	4.5	MHz
Standby current of iRC2	CPU in SLEEP mode, SC = 1011b, CM = 1110b	I_{iRC2}		100	150	µA
SYSCL_iRC2	CPU active, SC = 1011b, CM = 1110b	f_{SYSCL}	1.4	2.0	3.0	MHz
Standby current of iRC3	CPU in SLEEP mode, SC = 1111b, CM = 1111b	I_{iRC3}		40	70	µA
SYSCL_iRC3	CPU active, SC = 1111b, CM = 1111b	f_{SYSCL}	0.60	0.80	1.3	MHz
Stability	ΔV_{DD} = 5 V ± 20 %	df/f_0			”5	%
System clock crystal/ceramic oscillator (for additional characteristics see figures 49)						
Standby current	CPU in SLEEP mode, 4-MHz crystal active	I_{xtal}			125	µA
Start-up time	V_{DD} = 2.4 V	$t_{startup}$		8	10	ms
Stability	ΔV_{DD} = 3 V to 5.5 V	df/f_0		0.3	0.5	ppm
RC oscillator – external resistor (for additional characteristics see figures 52 to 54)						
Standby current	CPU in SLEEP mode, R_{ext} = 150 kΩ (± 1 %)	I_{xRC}			125	µA
Frequency	CPU active, R_{ext} = 150 kΩ	f_{SYSCL}	1.8	2.0	2.2	MHz
Stability	V_{DD} = 2.4 V to 5.5 V	df/f_0			”10	%

AC Characteristics (continued)

Parameters	Test Conditions / Pins	Symbol	Min.	Typ.	Max.	Unit
32-kHz crystal oscillator						
Active current	CPU active/running	I_{DD32k}			10	µA
HALT current	CPU in SLEEP mode	I_{HALTx}		1.0	1.5	µA
Start-up time	V_{DD} = 2.4 V	$t_{startup}$			1.5	s
Stability	ΔAV_{DD} = 100 mV	df/f_0		0.1	0.3	ppm
External clock input at SCLIN, TIM1 and T0IN						
SCLIN input clock $f_{SCLIN} = 2 \times f_{SYSCL}$	CPU active, V_{DD} > 2.4 V rise/fall time < 50 ns, see figure 47	f_{SYSCL}		4	8	MHz
TIM1, T0IN input frequ.	rise/fall time < 30 ns	f_{IN}			10	MHz
EEPROM program/ configuration memory						
Number of programming cycles		n	1000			Cycles

Crystal Characteristics

Parameters	Test Conditions / Pins	Symbol	Min.	Typ.	Max.	Unit
32-kHz crystal						
Crystal frequency		f_X		32.768		kHz
Series resistance		RS		30	50	kΩ
Static capacitance		C0		1.5		pF
Dynamic capacitance		C1		3		fF
Load capacitance		C_L	8	10	12.5	pF
System clock crystal						
Crystal frequency		f_X	1.5	4	8	MHz
Series resistance		RS		30	50	Ω
Static capacitance		C0		2	4.5	pF
Dynamic capacitance		C1		3	15	fF

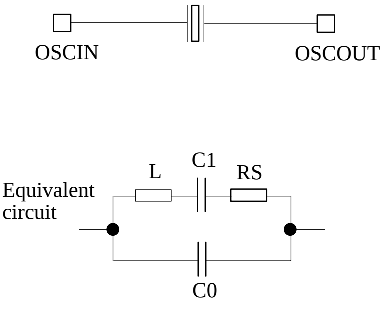


Figure 48. Crystal equivalent circuit

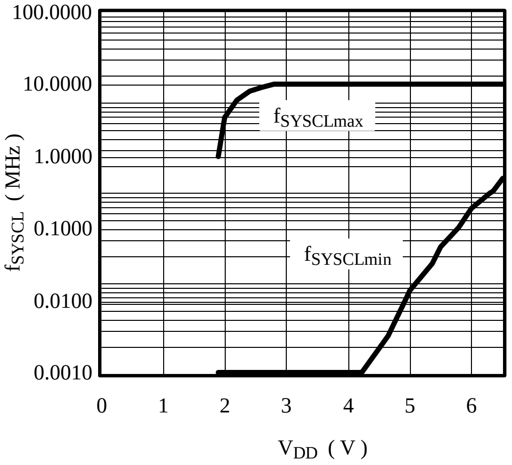


Figure 49. Worst case minimum/ maximum system frequency (using external RC or crystal oscillator)

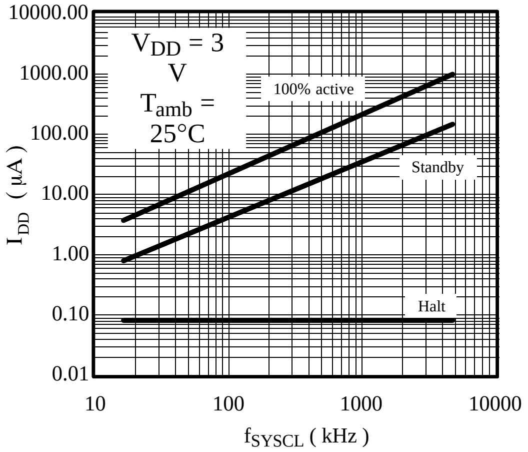


Figure 50. $I_{DD} = f (f_{SYSCL})$

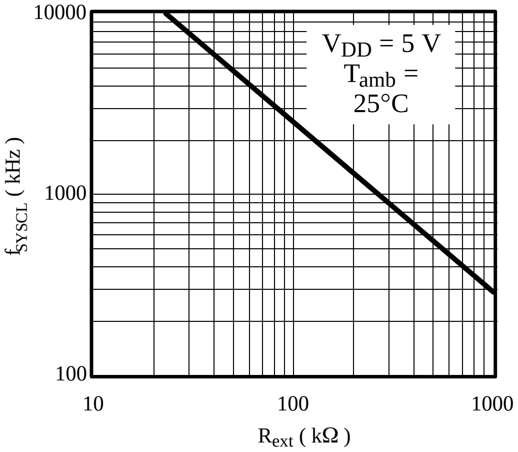


Figure 53. $f_{SYSCL} = f (R_{ext})$

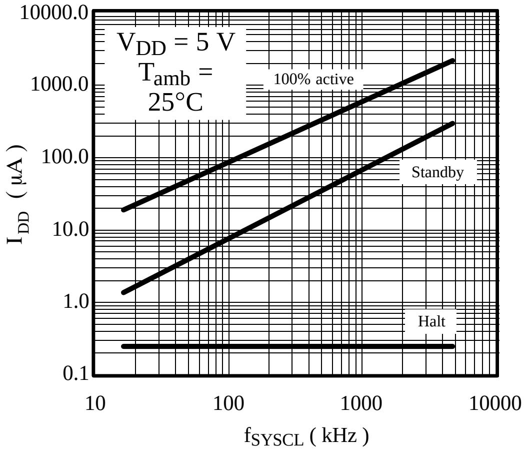


Figure 51. $I_{DD} = f (f_{SYSCL})$

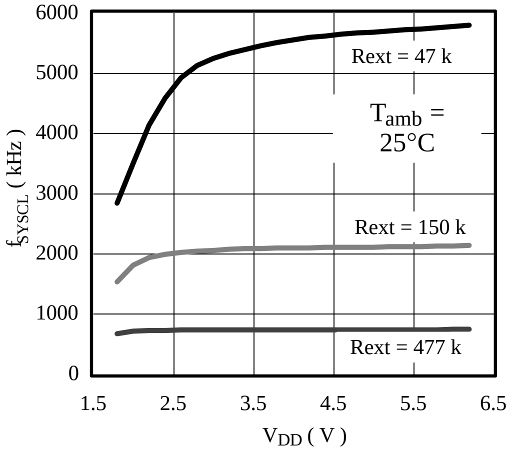
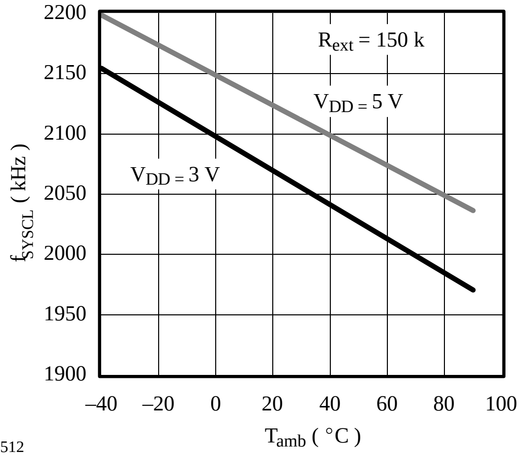


Figure 54. $f_{SYSCL} = f (V_{DD}, R_{ext})$



16512

Figure 52. $f_{SYSCL} = f (T_{amb})$; external RC

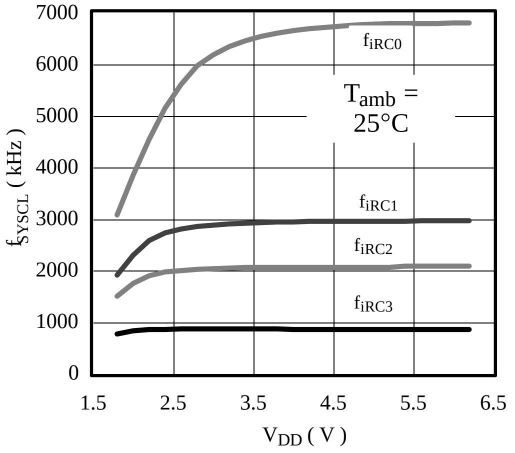


Figure 55. $f_{SYSCL} = f (V_{DD})$; internal RC

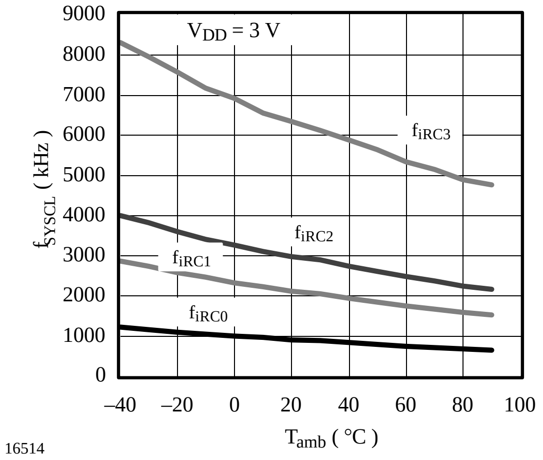


Figure 56. f_{SYSCL} = f (T_{amb})

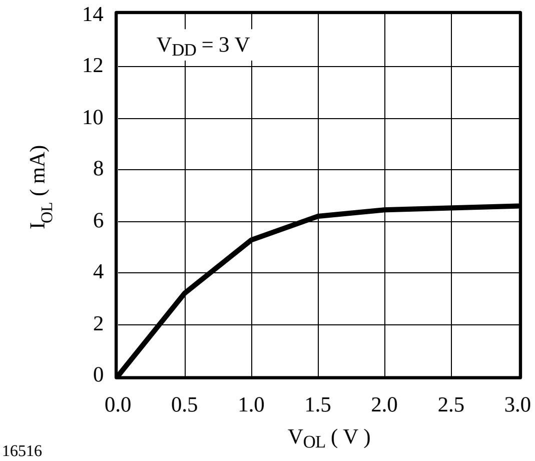


Figure 59. Typical low output driver

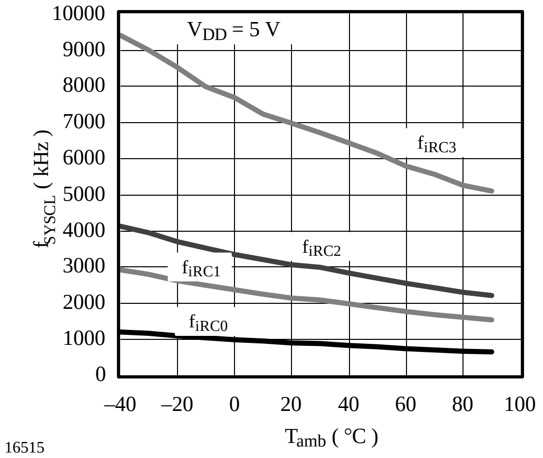


Figure 57. f_{SYSCL} = f (T_{amb})

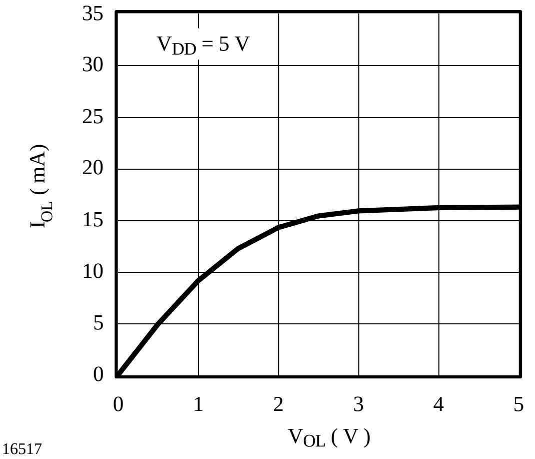


Figure 60. Typical low output driver

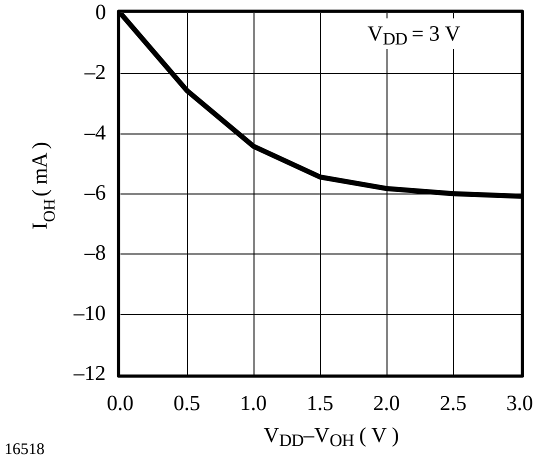


Figure 58. Typical high output driver

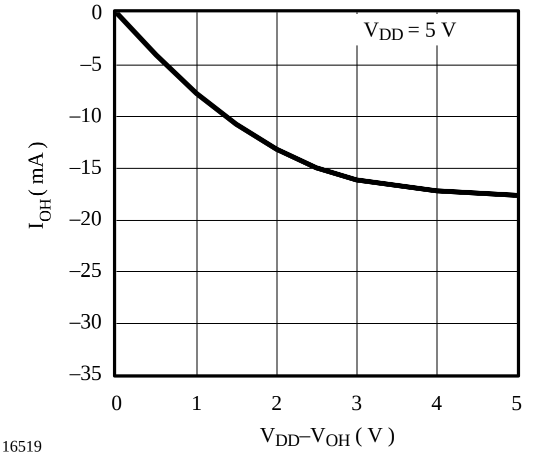
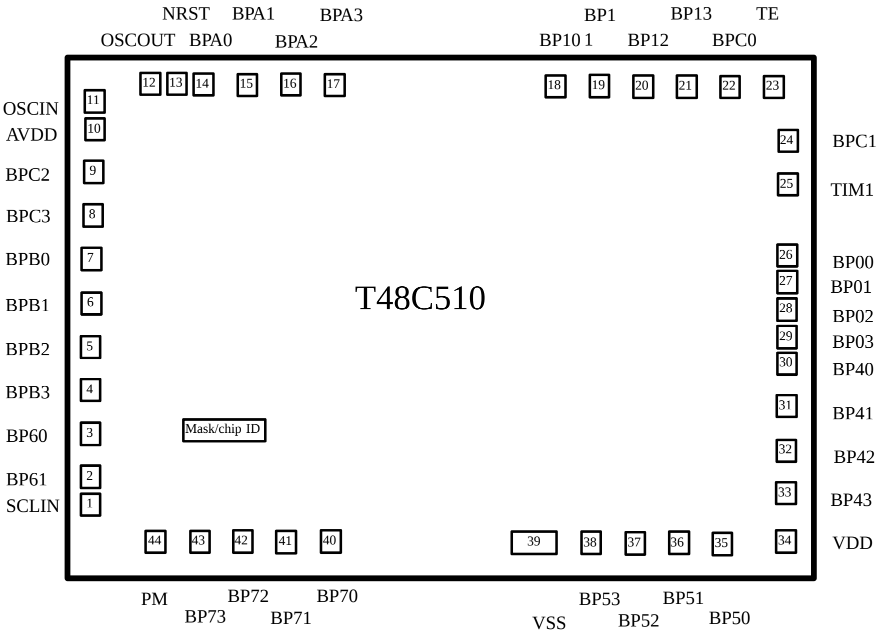


Figure 61. Typical high output driver

4 Device Information

4.1 Pad Layout



Die size: 2.99 x 4.30 mm

Standard pad size:
96 x 96 µm

Pad size (VSS):
246 x 96 µm

Thickness:
480 ± 25 µm
(19 ± 1 mil)

Figure 62. Pad layout

Table 26 Pad coordinates

Pad No.	Name	X-Coord.	Y-Coord		Pad No.	Name	X-Coord.	Y-Coord.
1	SCLIN	113.80	353.95		23	TE	3861.80	2678.70
2	BP61	113.80	503.95		24	BPC1	3939.70	2374.60
3	BP60	113.80	744.40		25	TIM1	3939.70	2134.15
4	BPB3	113.80	984.85		26	BP00	3939.70	1744.20
5	BPB2	113.80	1225.30		27	BP01	3939.70	1594.20
6	BPB1	113.80	1465.75		28	BP02	3939.70	1444.20
7	BPB0	113.80	1706.20		29	BP03	3939.70	1294.20
8	BPC3	113.80	1946.65		30	BP40	3939.70	1144.20
9	BPC2	113.80	2187.10		31	BP41	3939.70	903.75
10	AVDD	113.80	2426.65		32	BP42	3939.70	663.30
11	OSCIN	113.80	2576.65		33	BP43	3939.70	422.85
12	OSCOUT	421.80	2678.70		34	VDD	3939.70	147.90
13	NRST	571.80	2678.70		35	BP50	3590.95	146.45
14	BPA0	721.80	2678.70		36	BP51	3350.50	146.45
15	BPA1	962.25	2678.70		37	BP52	3110.05	146.45
16	BPA2	1202.70	2678.70		38	BP53	2869.60	146.45
17	BPA3	1443.15	2678.70		39	VSS	2474.15	146.45
18	BP10	2659.55	2678.70		40	BP70	1431.05	146.45
19	BP11	2900.00	2678.70		41	BP71	1190.60	146.45
20	BP12	3140.45	2678.70		42	BP72	950.15	146.45
21	BP13	3380.90	2678.70		43	BP73	709.70	146.45
22	BPC0	3621.35	2678.70		44	VSS	469.25	146.45

4.2 Packaging

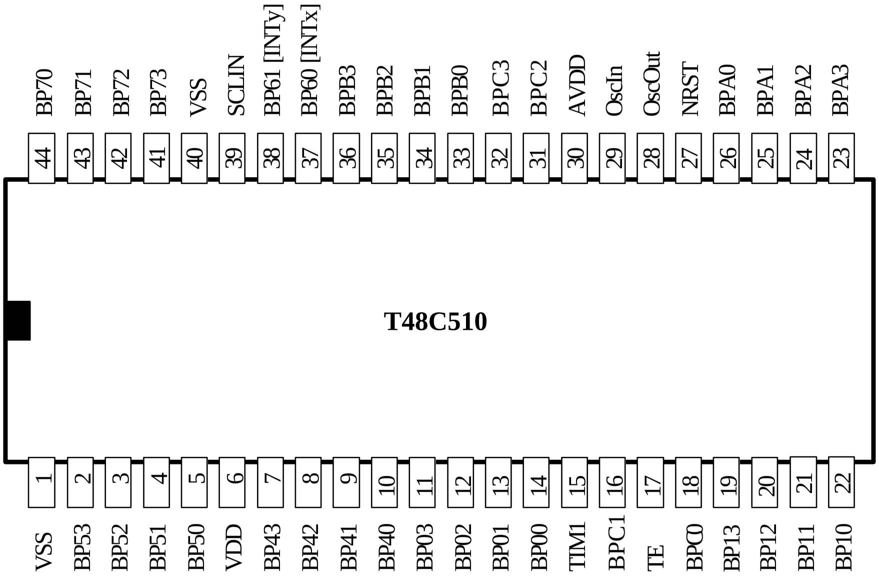
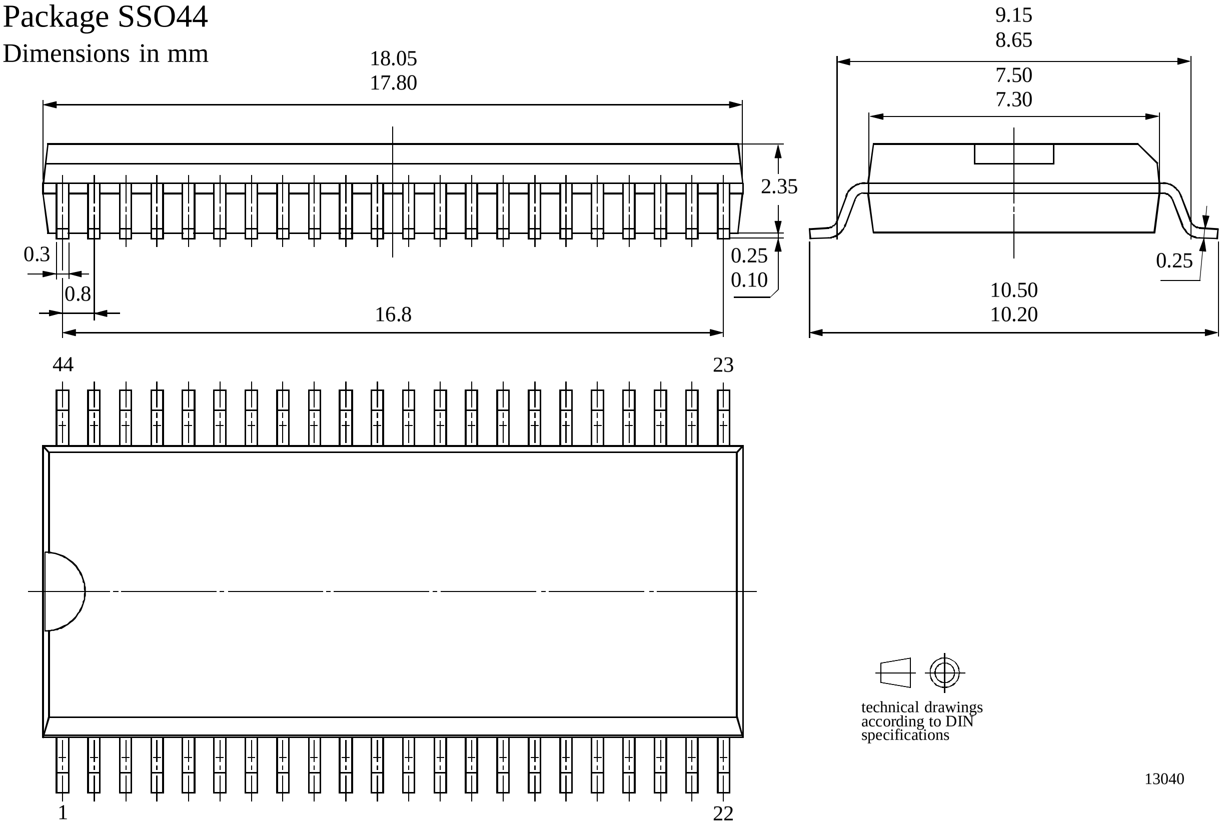


Figure 63. Pin connections SSO44-package

Package SSO44
Dimensions in mm



5 Hardware Options

The following list shows all the T48C510 hardware options that can be programmed into the configuration EEPROM.
SPD –> strong static pull-down, SPU –> strong static pull-up

Port 0	**Output**		Standard drive
	BP00	CMOS	Pull-up
	BP01	CMOS	Pull-up
	BP02	CMOS	Pull-up
	BP03	CMOS	Pull-up

Port 1	**Output**		Standard drive
	BP10	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	
	BP11	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	
	BP12	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	
	BP13	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	

Port 4	**Output**		Standard drive
	BP40	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)
	BP41	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)
	BP42	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)
	BP43	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)

Port 5	**Output**		Standard drive
	BP50	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)
	BP51	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)
	BP52	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)
	BP53	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)

Port 7	**Output**		Standard drive
	BP70	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)
	BP71	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)
	BP72	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)
	BP73	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)

Port 6	**Output**		Standard drive
	BP60	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (4 kΩ)
			□ SPD (4 kΩ)
	BP61	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (4 kΩ)
			□ SPD (4 kΩ)

Port A	**Output**	Standard	Drive
	BPA0	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)
	BPA1	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)
	BPA2	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)
	BPA3	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)

Port B	**Output**	Standard	Drive
	BPB0	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)
	BPB1	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)
	BPB2	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)
	BPB3	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)

TIM1	**Output** □	Standard	□ Drive
		□ CMOS	
		□ Open drain [N]	□Pull-up
		□ Open drain [P]	□Pull-down

Port C	**Output**	Standard	Drive
	BPC0	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)
	BPC1	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)
	BPC2	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)
	BPC3	□ CMOS	□ Pull-up
		□ Open drain [N]	□ Pull-down
		□ Open drain [P]	□ SPU (30 kΩ)
			□ SPD (30 kΩ)

BPA-Reset

□ No
□ BPA0 & BPA1 = low
□ BPA0 & BPA1 & BPA2 = low
□ BPA0 & BPA1 & BPA2 & BPA3 = low
□ BPA0 & BPA1 = high
□ BPA0 & BPA1 & BPA2 = high
□ BPA0 & BPA1 & BPA2 & BPA3 = high

Watchdog

□ $^1/_2$ s □ Disabled
□ 1 s
□ 2 s

OSCIN No integrated capacitance

OSCOUT No intergrated capacitance

Ozone Depleting Substances Policy Statement

It is the policy of **Atmel Germany GmbH** to

1. Meet all present and future national and international statutory requirements.
2. Regularly and continuously improve the performance of our products, processes, distribution and operating systems with respect to their impact on the health and safety of our employees and the public, as well as their impact on the environment.

It is particular concern to control or eliminate releases of those substances into the atmosphere which are known as ozone depleting substances (ODSs).

The Montreal Protocol (1987) and its London Amendments (1990) intend to severely restrict the use of ODSs and forbid their use within the next ten years. Various national and international initiatives are pressing for an earlier ban on these substances.

Atmel Germany GmbH has been able to use its policy of continuous improvements to eliminate the use of ODSs listed in the following documents.

1. Annex A, B and list of transitional substances of the Montreal Protocol and the London Amendments respectively
2. Class I and II ozone depleting substances in the Clean Air Act Amendments of 1990 by the Environmental Protection Agency (EPA) in the USA
3. Council Decision 88/540/EEC and 91/690/EEC Annex A, B and C (transitional substances) respectively.

Atmel Germany GmbH can certify that our semiconductors are not manufactured with ozone depleting substances and do not contain such substances.

We reserve the right to make changes to improve technical design and may do so without further notice. Parameters can vary in different applications. All operating parameters must be validated for each customer application by the customer. Should the buyer use Atmel Wireless & Microcontrollers products for any unintended or unauthorized application, the buyer shall indemnify Atmel Wireless & Microcontrollers against all claims, costs, damages, and expenses, arising out of, directly or indirectly, any claim of personal damage, injury or death associated with such unintended or unauthorized use.

Data sheets can also be retrieved from the Internet: http://www.atmel–wm.com

Atmel Germany GmbH, P.O.B. 3535, D-74025 Heilbronn, Germany
Telephone: 49 (0)7131 67 2594, Fax number: 49 (0)7131 67 2423