MARC4 – 4-bit Universal Microcontroller

The M44C510 is a member of the TEMIC family of 4-bit single-chip microcontrollers. It contains ROM, RAM, up to 32 digital I/O pins, up to 10 maskable external interrupt sources, 6 maskable internal interrupts, a watchdog timer, 32-kHz oscillator with programmable watch timer, 2 x 8-bit multifunction timer/counter module and a versatile on-chip system clock generation module.

Features

- 4 K x 8-bit application ROM
- 256 x 4-bit RAM
- 8 hardware and software interrupt priority levels
- Bitwise maskable prioritized interrupts
- Up to 10 external and 4 internal interrupts
- Up to 32 I/O lines
- High drive ports (20 mA, V_{DD} = 5 V)
- I/O ports – bitwise configurable with combined interrupt handling (for serial I/O applications)
- 2 x 8-bit multifunction timer/counters
- 32-kHz on chip oscillator with programmable prescaler/interval timer
- User definable on-chip system clock generation
- 4-MHz crystal, 4-MHz ceramic resonator or fully integrated RC oscillator **

Benefits

- Extremely low-power consumption
- Minimal external components
- Coded reset and watchdog timer **
- Power-on reset, “brown out” function
- Power-down mode
- 2.4 V to 6.2 V supply voltage
- Data retention down to 2 V in SLEEP mode
- Efficient, hardware-controlled interrupt handling
- High-level programming language in qFORTH
- Comprehensive library of useful routines
- PC based development tools

(** mask option)

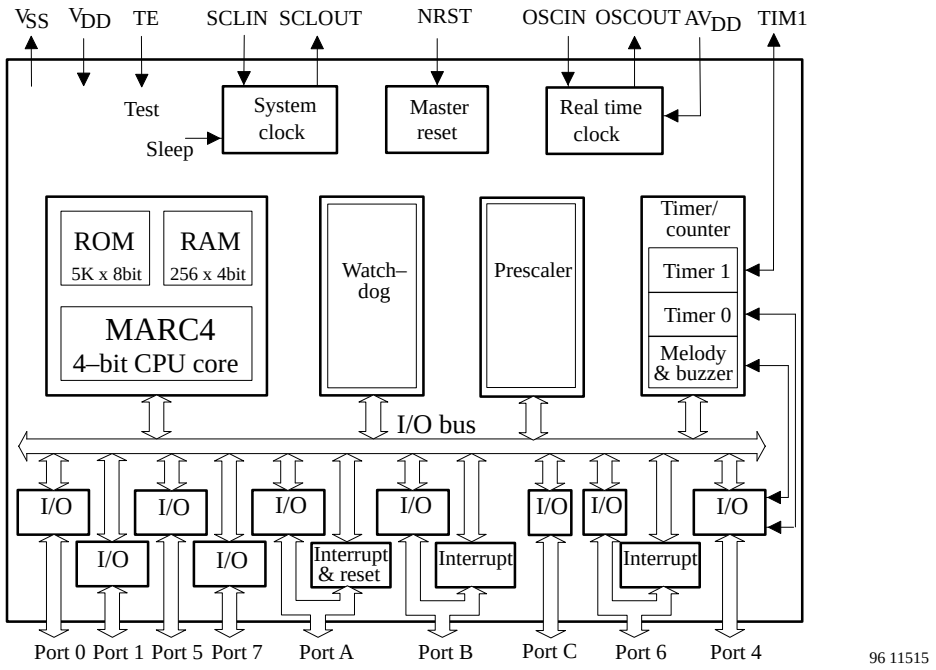


Figure 1. Block diagram

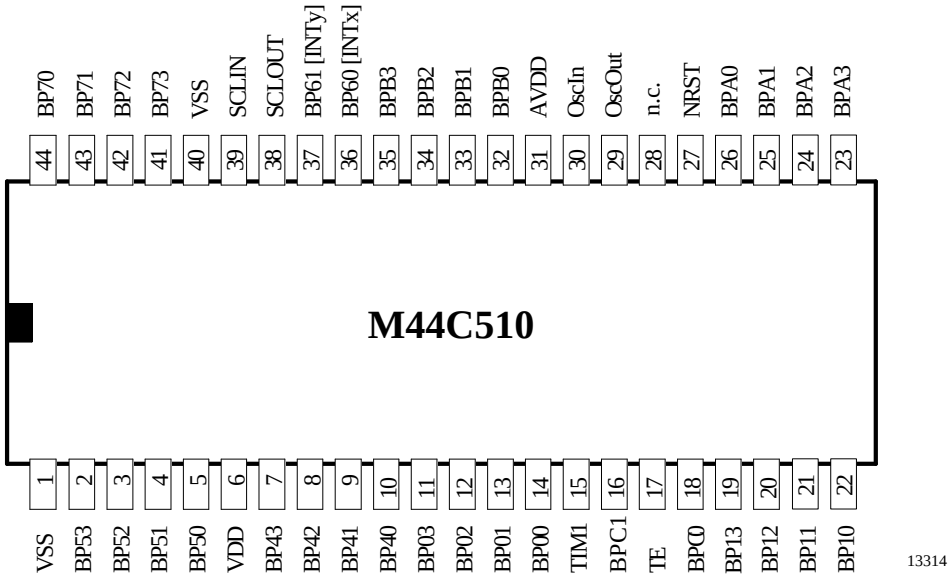


Figure 2. Pin connections SSO44-package

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 +6.2V
V_{SS}	Circuit ground
BP00 – BP03	4 bidirectional I/O lines of Port 0 – automatic nibblewise configurable I/O
BP10 – BP13	4 bidirectional I/O lines of Port 1(*) – automatic nibblewise configurable I/O
BP50 – BP53	4 bidirectional I/O lines of high current Port 5(*) – bitwise configurable I/O
BP70 – BP73	4 bidirectional I/O lines of high current Port 7(*) – bitwise configurable I/O
BPA0 – BPA3	4 bidirectional I/O lines of Port A(*) – bitwise configurable I/O and as inputs to a port monitor module. Optional coded reset inputs (*)
BPB0 – BPB3	4 bidirectional I/O lines of Port B(*) – bitwise configurable I/O and as inputs to a port monitor module
BPC0 – BPC1	2 bidirectional I/O lines of Port C (*) – bitwise configurable I/O
BP60 – BP61	2 bidirectional I/O lines of Port 6 (*) – bitwise configurable I/O or as 2 external programmable interrupts
BP40-T0OUT0	I/O line BP40 of Port 4(*) – configurable I/O or timer/counter 0 I/O T0OUT0
BP41-T0OUT1	I/O line BP41 of Port 4(*) – configurable I/O or timer/counter 0 I/O T0OUT1
BP42-BUZ	High current I/O line BP42 of Port 4(*) – configurable I/O or buzzer output BUZ
BP43-NBUZ	High current I/O line BP43 of Port 4(*) – configurable I/O or buzzer output NBUZ
TIM1	Dedicated bidirectional I/O for Timer 1
SCLIN	4-MHz quartz crystal/ceramic resonator or trimming resistor pin (mask-option dependent)
SCLOUT	4-MHz quartz crystal/ceramic resonator pin (mask-option dependent)
OSCIN	32-kHz quartz crystal pin (mask-option dependent)
OSCOUT	32-kHz quartz crystal pin (mask-option dependent)
TE	Testmode input. This input is used to control the test modes (internal pull-down)
NRST	Reset input (/output), a logic low on this pin resets the device. An internal watchdog or coded reset is indicated by a low pulse on this pin.

(*) For mask options, please see the order information.

Contents

1 MARC4 Architecture

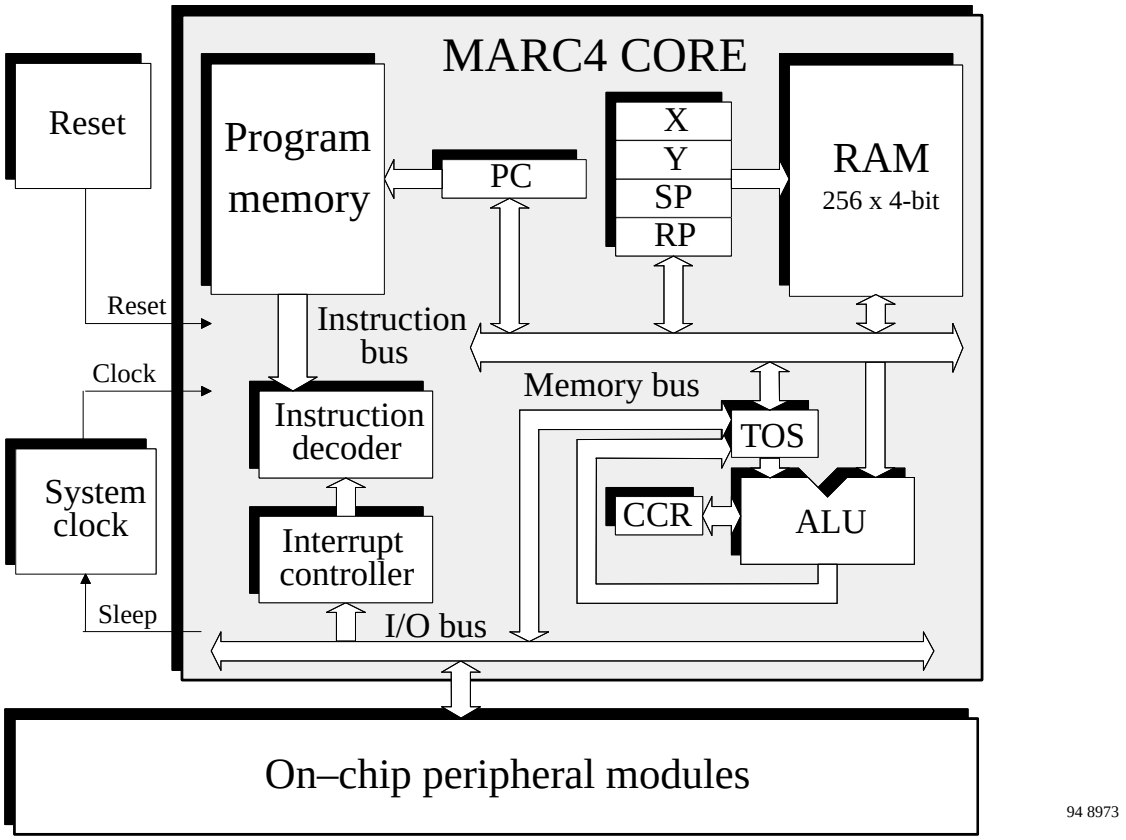


Figure 3. MARC4 core

1.1 General Description

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 (ROM) and data memory (RAM). Three independent buses, the instruction bus, the memory bus and the I/O bus are used for parallel communication between ROM, 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 ROM, 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 ROM

The program memory (ROM) is mask programmed with the customer application program during the fabrication of the microcontroller. The ROM is addressed by a 12-bit wide program counter, thus predefining a maximum program bank size of 4 Kbytes. An additional 1 Kbyte of ROM exists which is used partly for a quality control self-test program. The remaining space is available for the application program. The access to this additional ROM section is done by using a ROM-bank switch.

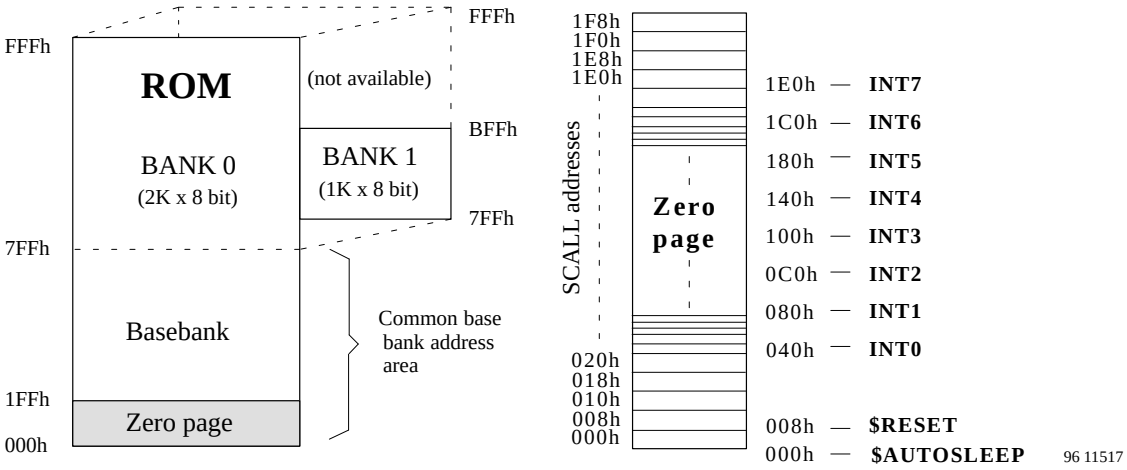


Figure 4. ROM map of M44C510

The lowest user ROM 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 ROM and are accessed via the MARC4's built-in TABLE instruction.

ROM Banking

Bank switching is fully supported by the compiler for customers programming with qFORTH. The MARC4 switches from one ROM bank to another by writing the new bank number to the ROM Bank Register (RBR). Conventional program space (power-up bank) resides in ROM bank 0. Each ROM bank consists of a 4-KByte address space whereby the lowest 2 KByte is common to all banks, so that addresses between 000h and 7FFh always accesses the same ROM data (see figure 4). When ROM banking is used, the compiler will, if necessary, insert the program code to save and restore the condition of the RBR on bank switching.

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 ROM. 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 ROM constants.

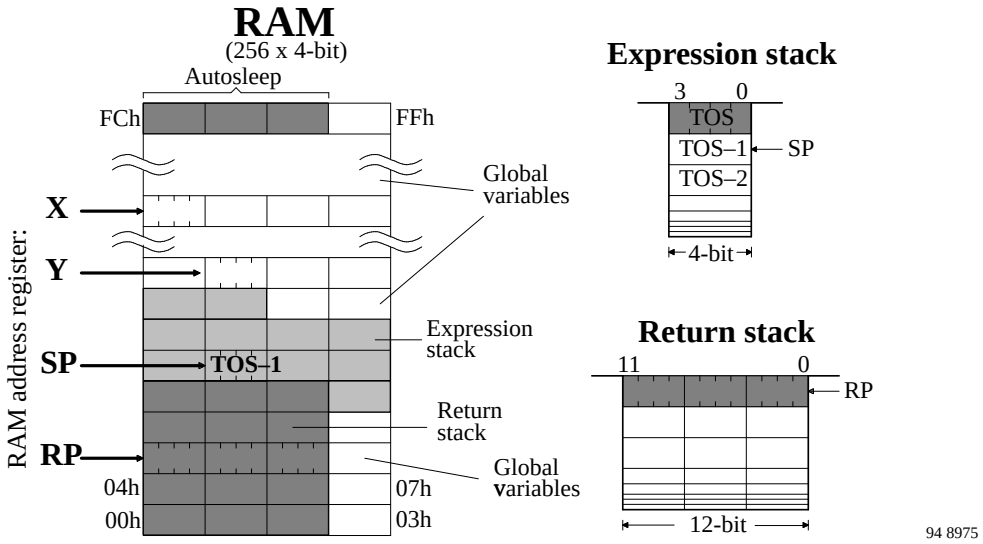


Figure 5. RAM map

ROM Banking Register (RBR)

The ROM banking register is a 4-bit register whereby in the M44C510, only bit 2 is used. This indicates which ROM bank is presently being addressed. The RBR is accessed with a standard qFORTH peripheral read or write instruction (IN or OUT, port address ‘D’ hex).

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, ROM, 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.

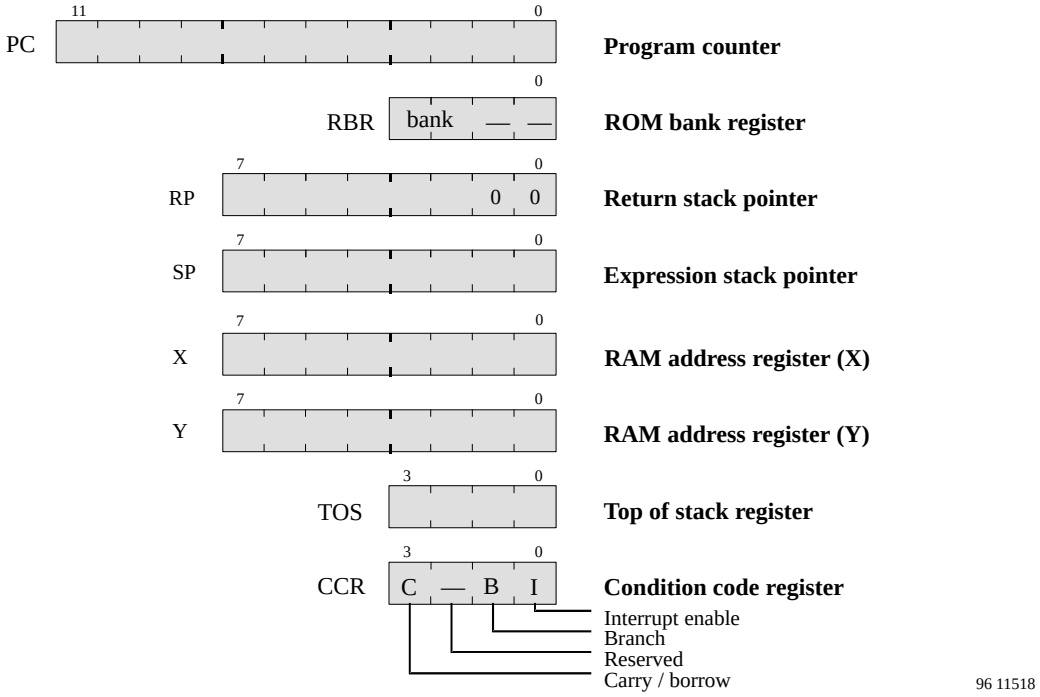


Figure 6. Programming model

1.2.4 ALU

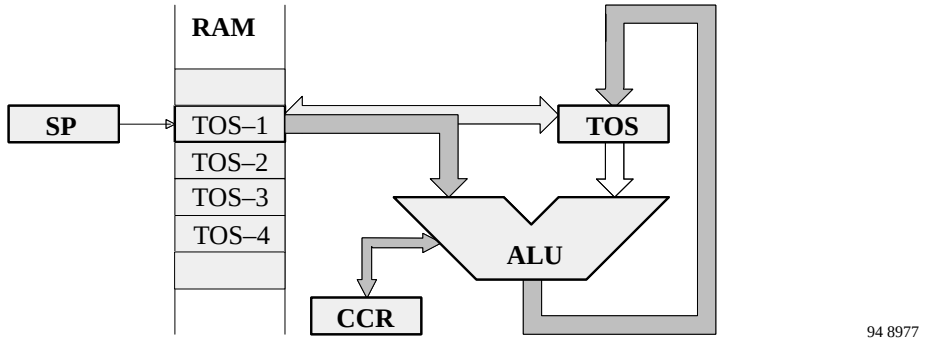


Figure 7. ALU zero-address operations

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 ROM 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 (see also the section “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 ROM (see table 2, page 10). 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”).

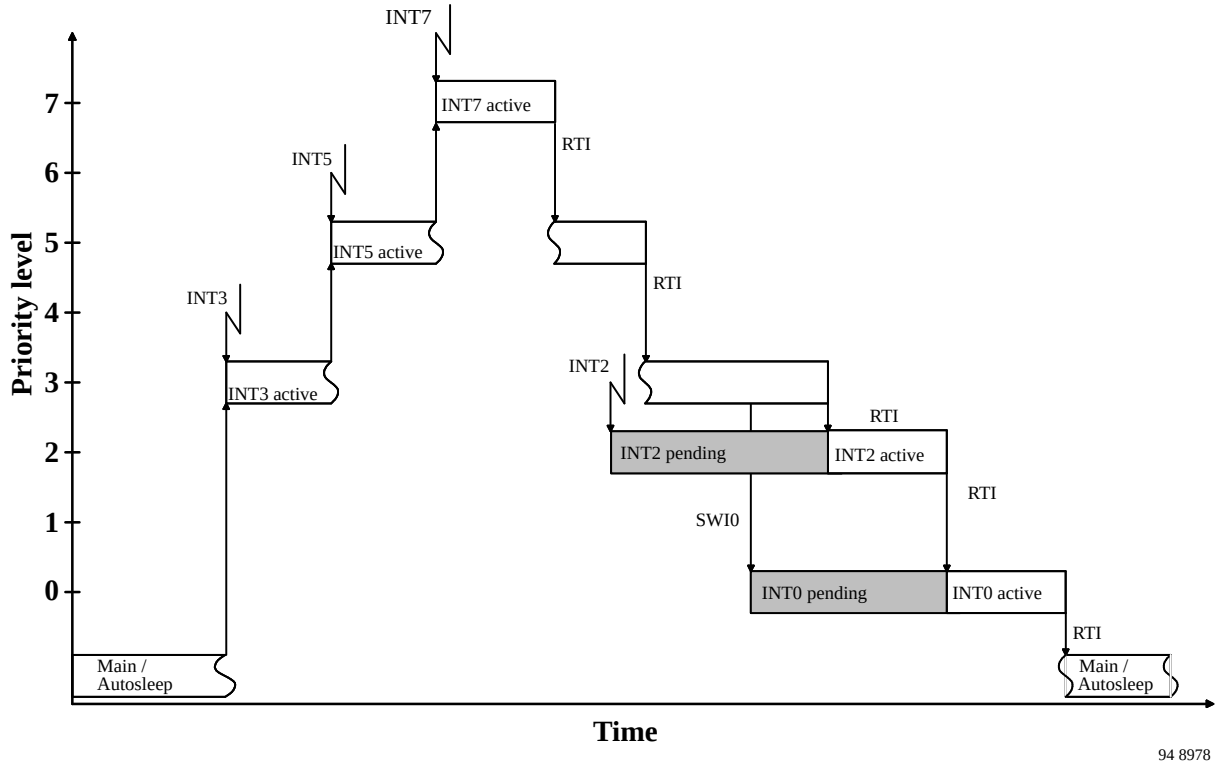


Figure 8. Interrupt handling

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 execution, 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 be 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).

It should also be realized that automatic stacking of the RBR is not carried out by the hardware and so if ROM banking is used, the RBR must be stacked on the expression stack by the application program and restored before the RTI. 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	ROM 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 Source	Possible Interrupt Priorities								RST	Interrupt Mask		Function
	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)

= customer mask option (see “Ordering Information”)

* = software configurable (see “Peripheral Modules” section for further details)

In the M44C510, there are eleven hardware interrupt sources which can be programmed to occupy a variety of priority levels. 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 programmer 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 register and the interrupt active register. During the reset phase, the I/O bus control signals are set to 'reset mode' thereby initializing all on-chip peripherals.

Releasing the reset results in a short call instruction (opcode C1h) to the ROM address 008h. This activates the initialization routine $RESET which in turn initializes all necessary RAM variables, stack pointers and 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 two machine cycles.

Coded Reset (Port A)

The coded reset circuit is connected directly to the Port A terminals. By using a mask option, 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
RST3	BPA0 & BPA1 & BPA2
RST4	BPA0 & BPA1 & BPA2 & BPA3

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 mask 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 register address (WDRES).

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

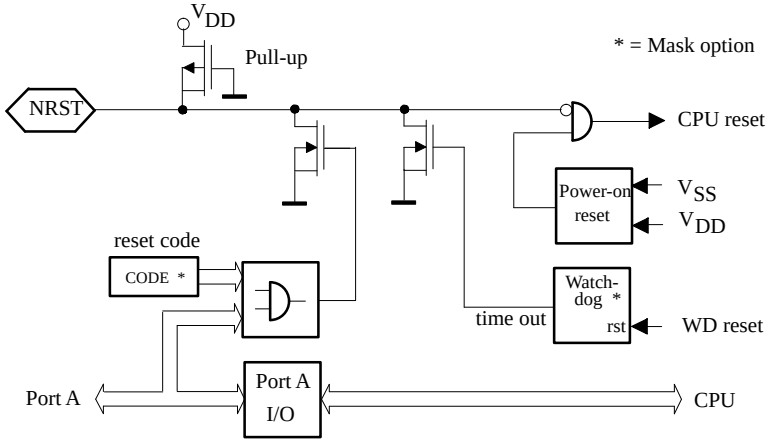


Figure 9. Reset configuration

1.5 Clock Generation

The M44C510 has a dual clock system, a 2-MHz system clock (SYSCL) for the core and a 32-kHz subclock (SUBCL) for the time-keeping peripheral modules (see figure 10). Each clock can be generated from independent on-chip oscillators or they can both be derived from the same high frequency SYSCL oscillator. This is mask selectable - allowing a choice of either an 4-MHz crystal, 4-MHz ceramic resonator or a RC oscillator. All the necessary oscillator circuitry, with the exception of the actual crystal or resonator, are integrated on chip. Therefore, if no exact timing is required, for example, it is possible to use the fully integrated RC oscillator, thus operating without any external components.

An additional mask option enables a high resolution trimmable RC oscillator whereby the SYSCL can be trimmed with an external resistor between SCLIN and V_{DD}. In this configuration, the SYSCL frequency can be maintained stable to within a tolerance of ±10% over the full operating temperature range. A SYSCL frequency of 2 MHz, for example, can be obtained by connecting a 220 kΩ resistor (see figures 44, 48 and 49).

Some applications require only long-term time keeping or low resolution timing. In this case, an on-chip, low-power 32-kHz crystal oscillator can be used to generate the SUBCL. This allows the core to go into SLEEP mode when not used, and therefore greatly reduces power consumption.

If the full 2-MHz timing resolution is required, then either the crystal or resonator oscillator should be used for SYSCL generation.

Should a suitable external 1...4-MHz or 32-kHz clock source be available, then SCLIN (Crystal oscillator configuration) or OSCIN respectively can be used as the input.

Note: A SYSCL frequency of 2 MHz leads an instruction cycle time of 1 µs.

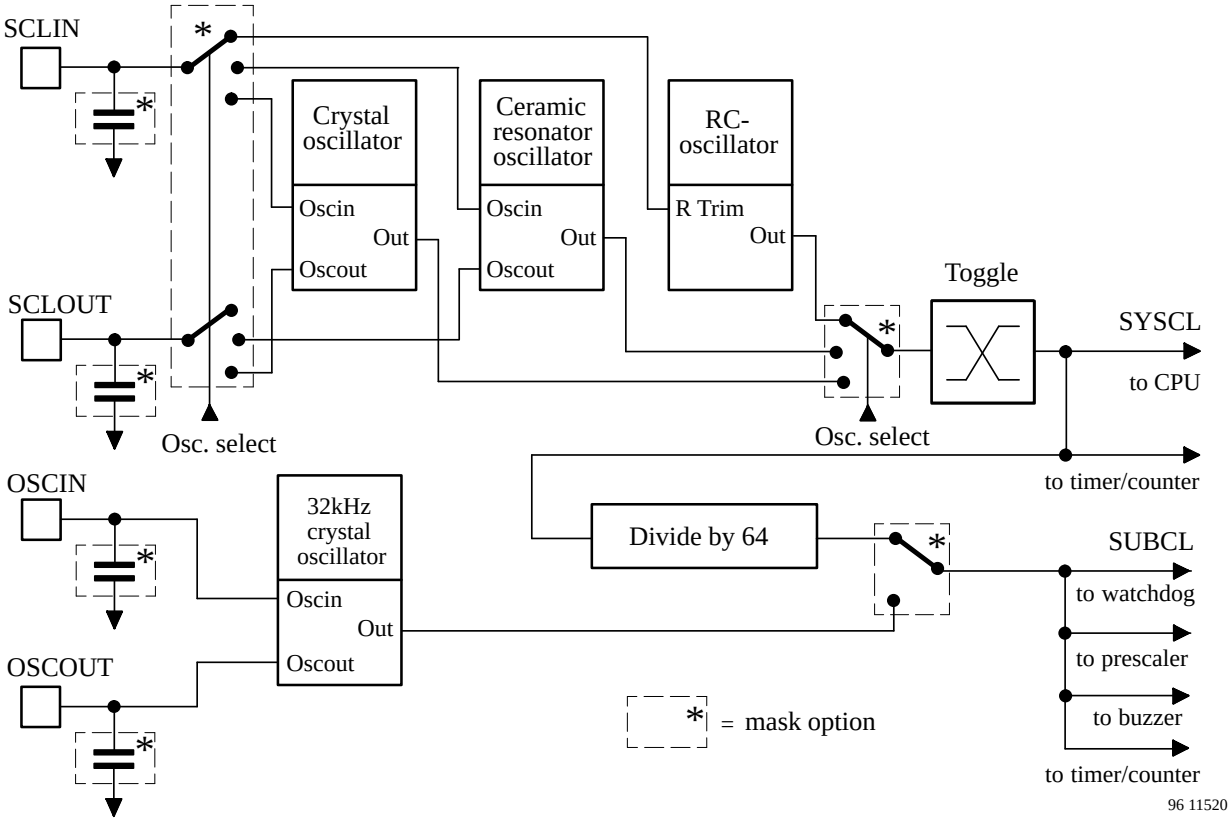


Figure 10. Clock module

1.5.1 Clock Monitor Mode

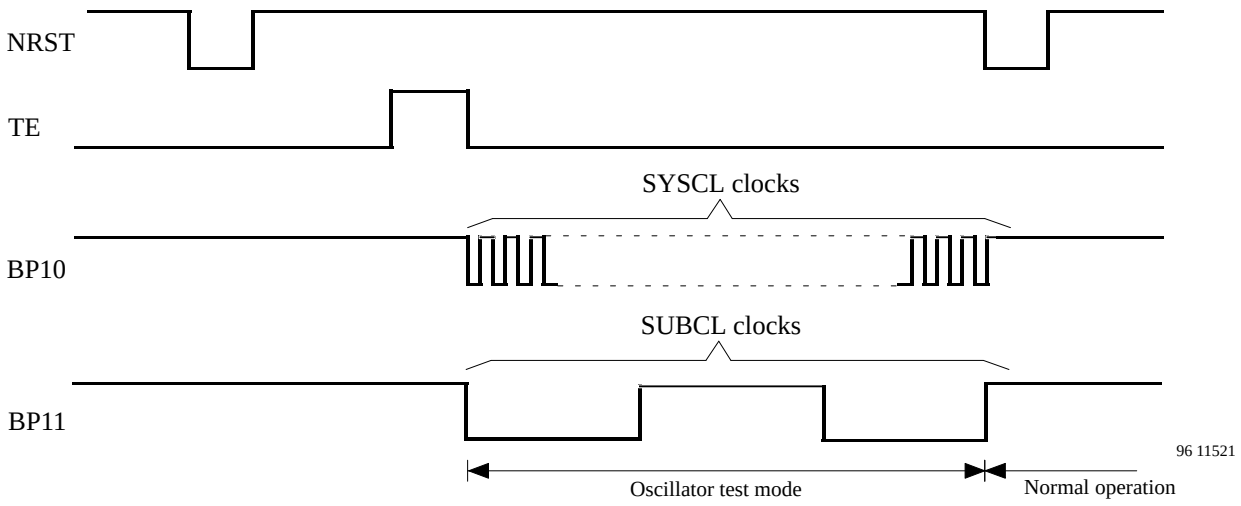


Figure 11. Clock monitoring

For trimming purposes, the M44C510 can be put into a clock monitor mode. The test input (TE) must therefore be pulsed high once, whereupon the SYSCL clock will appear on BP10 (Port 1, bit 0) and SUBCL clock on Port BP11 (Port 1, bit 1). To put BP10 and BP11 back into normal operation, the reset must be reapplied (see figure 11).

1.6 Sleep Mode

The sleep mode is a shutdown condition which is used to reduce the average system power consumption in applications where the µC is not fully utilized. In this mode, the system clock is stopped. The sleep mode is entered with the SLEEP instruction. This instruction sets the condition code register interrupt enable bit (I) to enable all interrupts and stops the core. During the sleep mode, the peripheral modules remain active and are able to generate interrupts. The µC exits the sleep mode with any interrupt or a reset. The sleep mode can only be maintained when no interrupt pending or active register bits are set. The application of the $AUTOSLEEP routine ensures the correct function of the sleep mode. The total power consumption is directly proportional to the active time of the µC. For a rough estimation of the expected average system current consumption, the following formula should be used:

$$I_{total}\,(V_{DD},f_{syscl}) = I_{Sleep} + (I_{DD} * T_{active} / T_{total})$$

I_{DD} depends on V_{DD} and f_{syscl}. Using a 32-kHz crystal, the SLEEP current (I_{Sleep}) is typically less than 1 µA. The active time of the core and the total emulation time are displayed in a separate window of the MARC4 emulator software.

2 Peripheral Modules

2.1 Addressing Peripherals

Accessing the peripheral modules takes place via the I/O bus (see figure 12). 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.

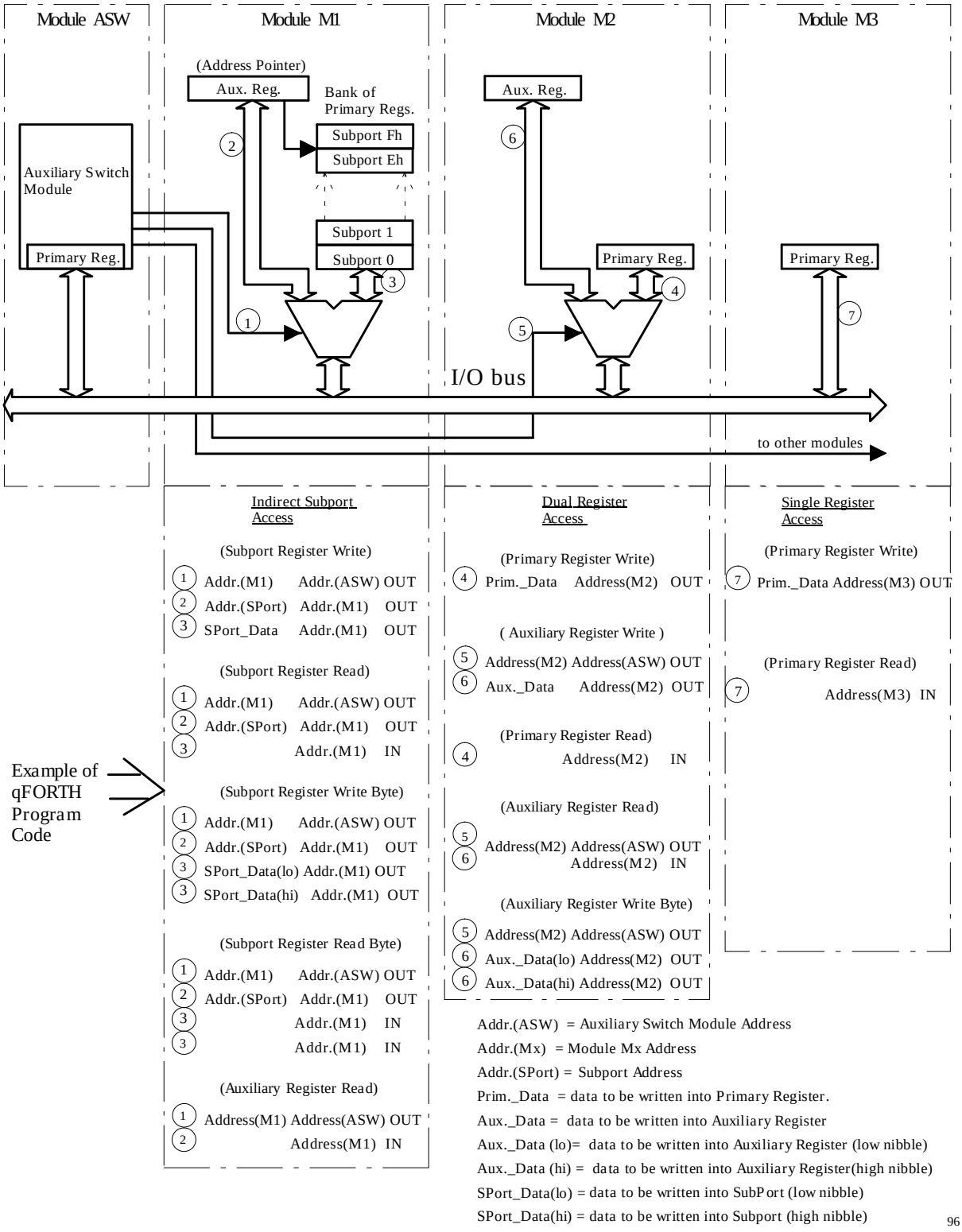


Figure 12. Example of I/O addressing

Table 5. M44C510 Peripheral addresses

Port Address			Name	Write /Read	Register Function	Module Type	See Page
0			P0DAT	W/R	Port 0 – data register/input data	M3	17
1			P1DAT	W/R	Port 1 – data register/input data	M3	17
2			PAIPR	W	Port A – interrupt priority register	M2	19
	Aux.		PAICR	W	Port A – interrupt control register		19
3			WDRES	R	Watchdog reset	M3	25
			PBIPR	W	Port B – interrupt priority register	M2	19
	Aux.		PBICR	W	Port B – interrupt control register		19
4			P4DAT	W/R	Port 4 – data register/pin data	M2	22
	Aux.		P4DDR	W	Port 4 – data direction register		18
5			P5DAT	W/R	Port 5 – data register/pin data	M2	18
	Aux.		P5DDR	W	Port 5 – data direction register		16
6			P6DAT	W/R	Port 6 – data register/pin data	M2	20
	Aux		P6CR	W	Port 6 – control register (byte)		21
7			P7DAT	W/R	Port 7 – data register/pin data	M2	18
	Aux.		P7DDR	W	Port 7 – data direction register		16
8			ASW	W	Auxiliary switch register	ASW	13
9			TCM	W/R	Data to/from subport addressed by TCX	M1	25
	Aux.		T0SR	R	Timer 0 interrupt status register	M1	31
			TCX	W	Timer/counter subport address pointer	M1	27
		Subport address					
		0	T0MO	W	Timer 0 mode register	M1	31
		1	T0CR	W	Timer 0 control register	M1	32
		2	T1MO	W	Timer 1 mode register	M1	40
		3	T1CR	W	Timer 1 control register	M1	41
		4	TCMO	W	Timer/counter mode register	M1	29
		5	TCIOR	W	Timer/counter I/O control register	M1	28
		6	TCCR	W	Timer/counter control register	M1	27
		7	TCIP	W	Timer/counter interrupt priority	M1	28
		8	T1CP	W	Timer 1 compare register (byte)	M1	41
			T1CA	R	Timer 1 capture register (byte)	M1	41
		9	T0CP	W	Timer 0 compare register (byte)	M1	33
			T0CA	R	Timer 0 capture register (byte)	M1	33
		A	BZCR	W	Buzzer control register	M1	44
		B-F			Reserved		
A			PADAT	W/R	Port A – data register/pin data	M2	18
	Aux.		PADDR	W	Port A – data direction register		16
B			PBDAT	W/R	Port B – data register/pin data	M2	18
	Aux.		PBDDR	W	Port B – data direction register		16
C			PCDAT	W/R	Port C – data register/pin data	M2	18
	Aux.		PCDDR	W	Port C – data direction register		16
D			RBR	W	Rom bank switch register	M3	5, 7
E			–	–	Reserved		
F			ITFSR	W	Interval timer frequency select register	M2	23
	Aux.		ITIPR	W	Interval timer interrupt priority register		24

2.2 Bidirectional Ports

With the exception of Port 6 and Port C, all other ports (0, 1, 4, 5, 7, A and B) are 4 bits wide. Port 6 and Port C have a data width of 2 bits (bit 0 and bit 1). All these ports may be used for data input or output. All ports are equipped with Schmitt-trigger inputs and a variety of mask options for open drain, open source and full complementary outputs and 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.

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

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**

Value: 1111b means all pins in input mode

Table 6. 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

There are five different types of bidirectional ports:

- Type 1 (Ports 0 and 1) – 4-bit wide, bidirectional ports with automatic full bus width direction switching.
- Type 2 (Ports 5 and 7) – 4-bit wide, Port C is a 2-bit wide, bitwise programmable high drive I/O port.
- Type 3 (Ports A and B) – 4-bit wide, bitwise programmable bidirectional ports with optional keyboard pull-ups.
- Type 4 (Port 6) – 2-bit wide, bitwise programmable bidirectional ports with optional bus pullups and programmable interrupt logic.
- Type 5 (Port 4) – 4-bit wide, bitwise programmable bidirectional port also provides the I/O interface to Timer 0 and the Buzzer.

2.2.1 Port 0, Port 1 – Bidirectional Ports Type 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 13). 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 into 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.

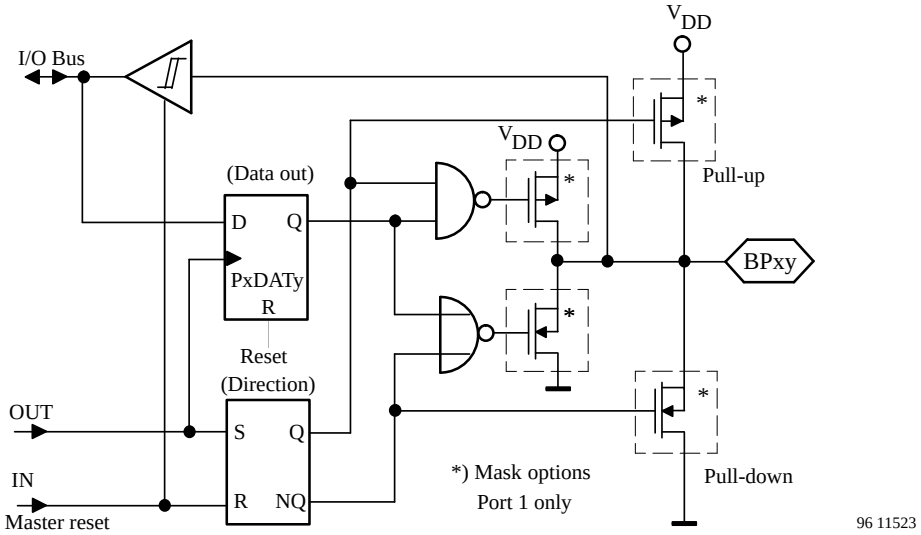


Figure 13. Bidirectional Ports 0 and 1

2.2.2 Port 5, Port 7, Port C – Bidirectional Ports Type 2

These, and all other bidirectional ports 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. This is a useful feature for self testing and for serial bus applications.

Both type 2 and type 3 bidirectional ports have the same I/O logic. Type 2, however, has an increased drive capability and type 3, an additional low resistance pull-up as customer mask option.

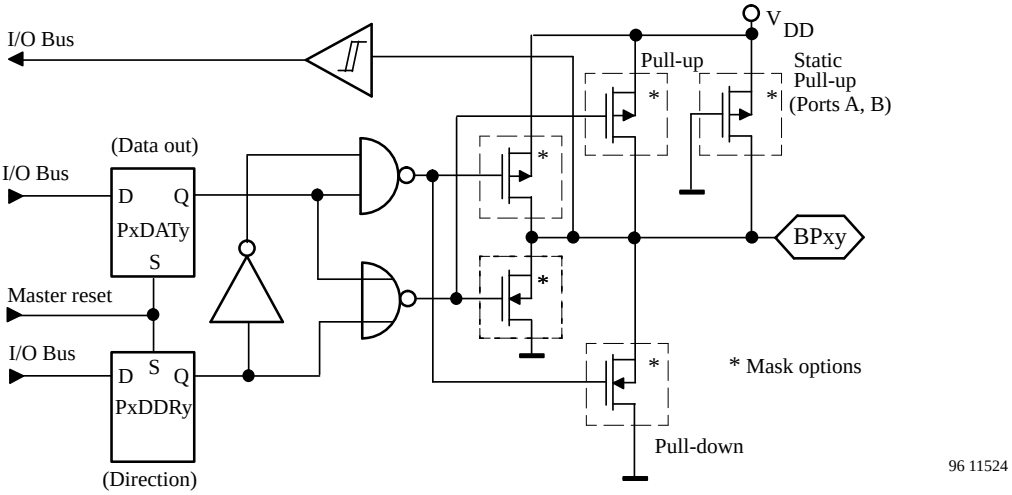


Figure 14. Bidirectional Ports 5, 7, A, B and C

2.2.3 Port A, Port B – Bidirectional Ports Type 3 – and Port Monitor Function

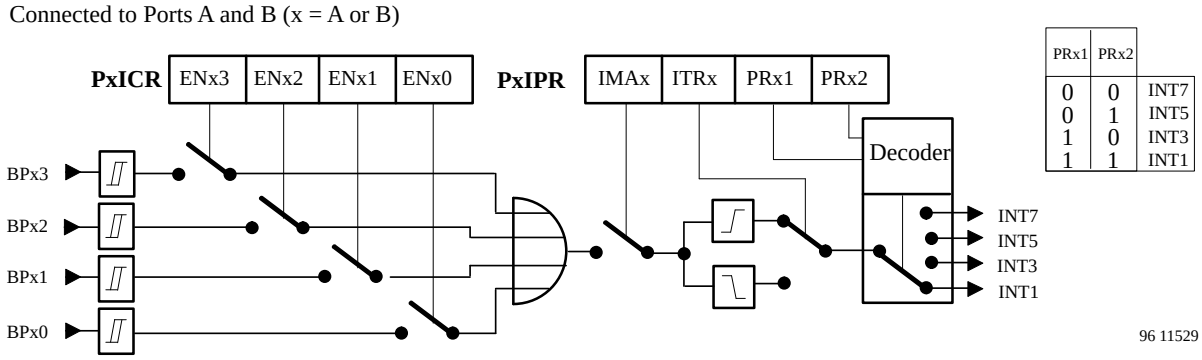


Figure 15. Port monitor module

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 port monitor modules. This module is connected across all four port pins (see figure 19) and generates an interrupt should a preprogrammed transition occur on any of the selected pins. This allows interrupt driven port scanning without the power consuming task of continuously polling the port inputs.

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. Port A can also be used for a mask programmable 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 7. 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 8. 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 Port 6 – Bidirectional Port Type 4

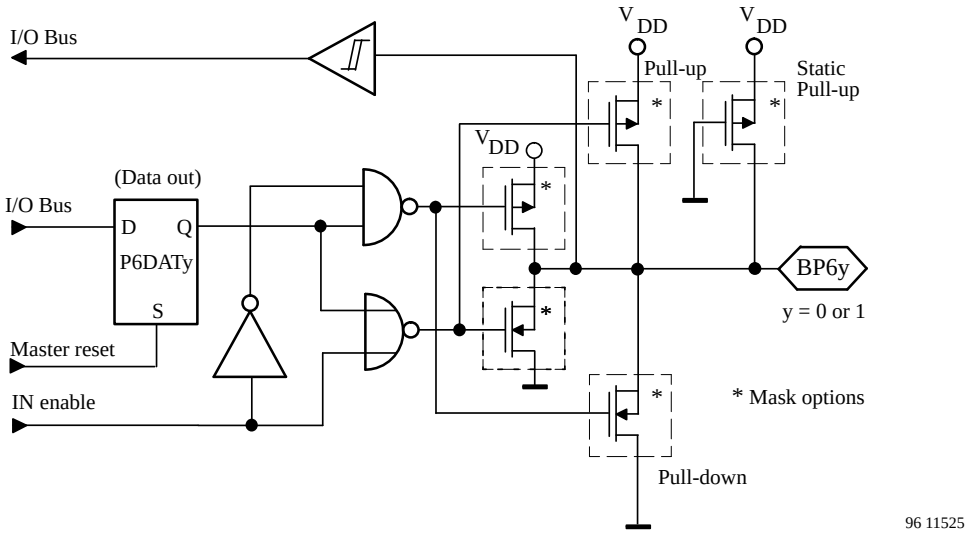


Figure 16. Bidirectional Port 6

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 15 and 16). 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 (mask 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 mask programmed 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 mask 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 mask selectable 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

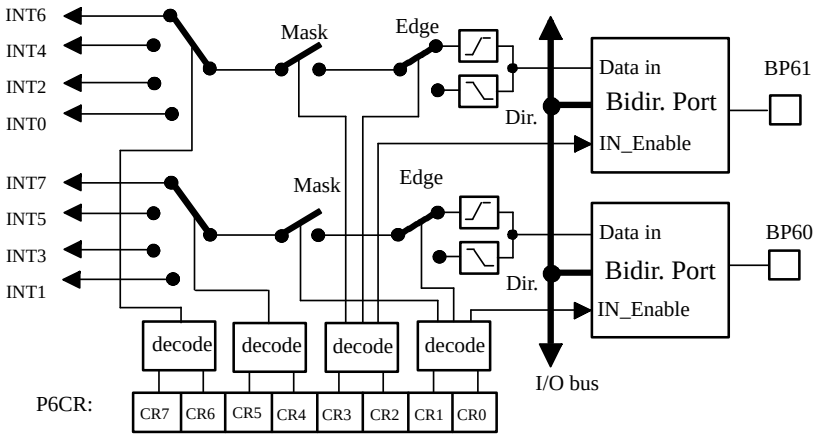
		Bit 3	Bit 2	Bit 1	Bit 0	
P6CR	**First write cycle**	**P61IM2**	**P61IM1**	**P60IM2**	**P60IM1**	**Reset value: 1111b**
		Bit 7	Bit 6	Bit 5	Bit 4	
	Second write cycle	**P61PR2**	**P61PR1**	**P60PR2**	**P60PR1**	**Reset value: 1111b**

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

P6xPR2, P6xPR1 – BP6x Interrupt priority code

Table 9. 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

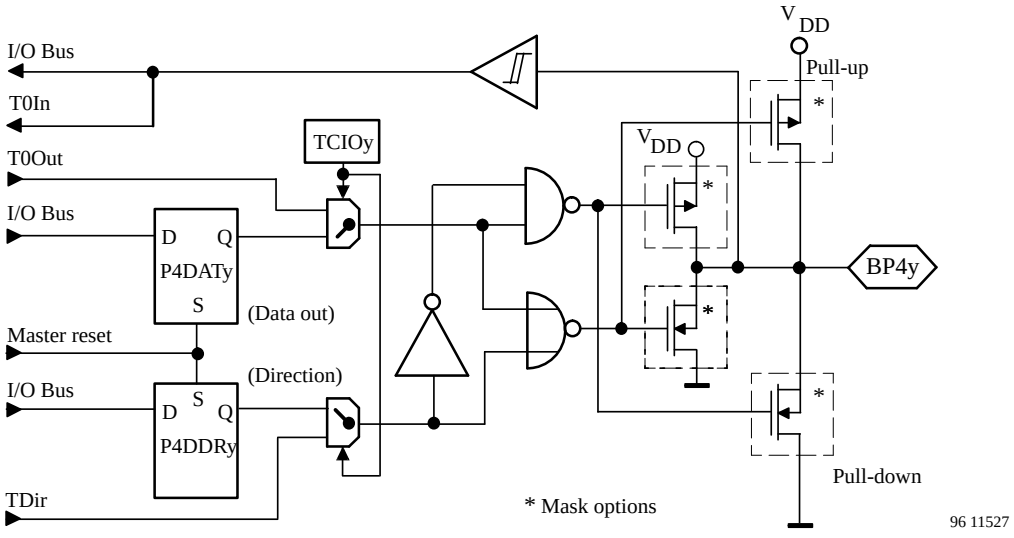
CR1 CR3	CR0 CR2	Dir.	INT edge	INT disabled
0	0	out	–	yes
0	1	in	⌐	no
1	0	in	¬	no
1	1	in	–	yes

96 11526

Figure 17. Port 6 external interrupts

2.2.5 Port 4 – Bidirectional Port Type 5

The type 5 bidirectional port is both a bitwise configurable I/O port and provides the external pins for both the Timer 0 and the internal buzzer generator. As a normal port, it performs in exactly the same way as bidirectional port type 2 (see figure 14). 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 17 shows the internal interfaces to Port 4.



96 11527

Figure 18. Bidirectional Port 4

2.2.6 TIM1 – Bidirectional Pin Timer 1

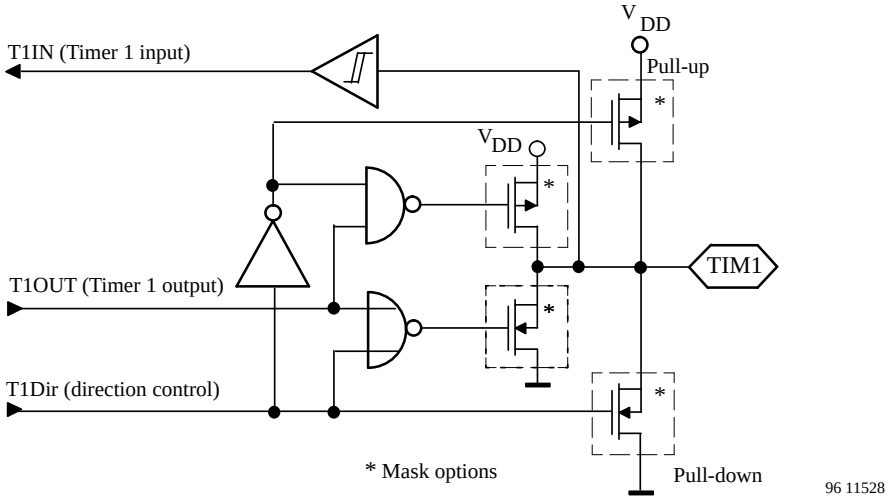


Figure 19. Bidirectional pin TIM1

TIM1 is a dedicated bidirectional I/O stage for signal communication to and from the Timer 1 in the timer/counter module (see figure 18). 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 10) and consists of a 15-stage binary divider and two programmable multiplexers for selecting the appropriate interrupt frequencies for each interrupt source (see figure 20). 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).

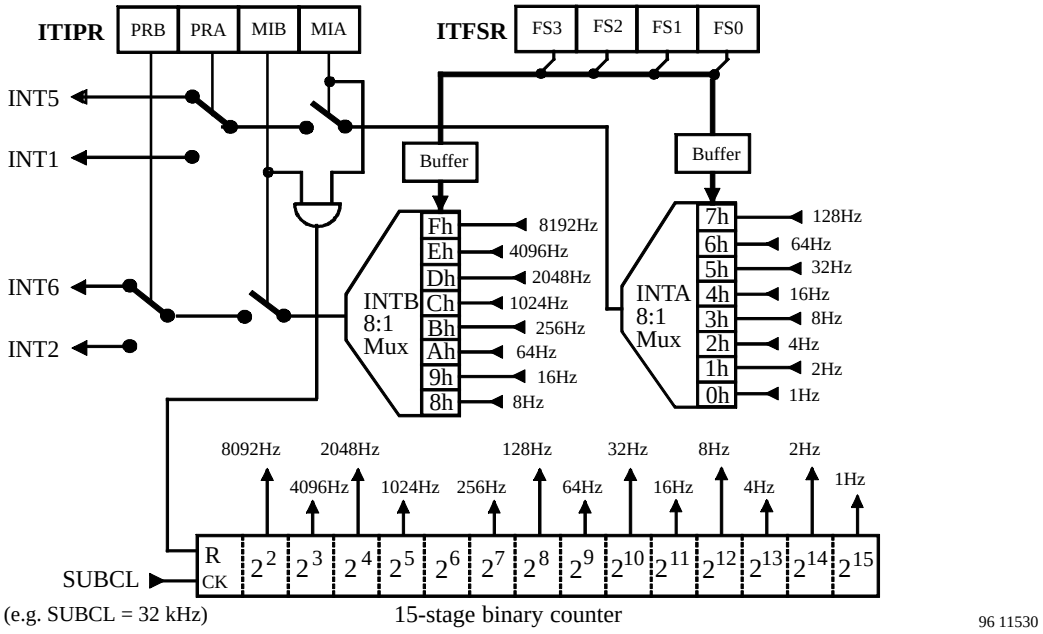


Figure 20. Interval timers / prescaler

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 watch 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 10. 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 11. Interval Timer Frequency Select Register (ITFSR)

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

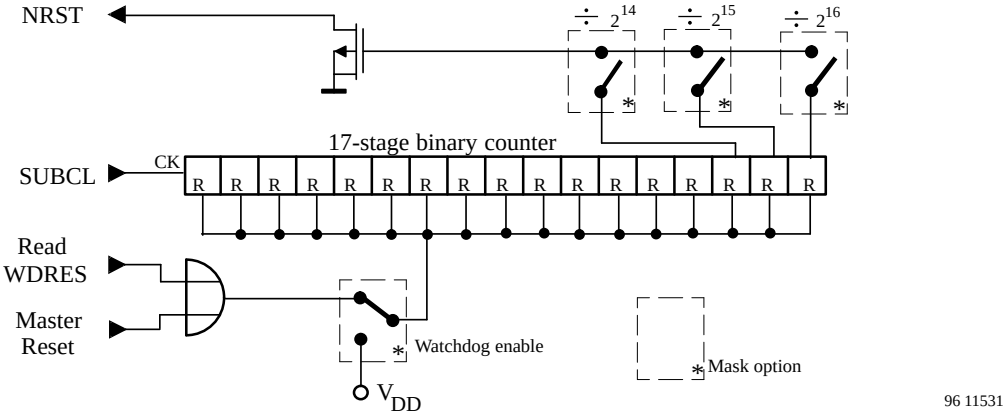


Figure 21. Watchdog timer

The Watchdog timer is a 17-stage binary divider clocked by SUBCL generated within the clock module (see figures 10 and 21). It can only be enabled as a mask 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 mask option.

To reset the watchdog, the program must perform an IN-instruction on the address WDRES (’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 figure 22). 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.

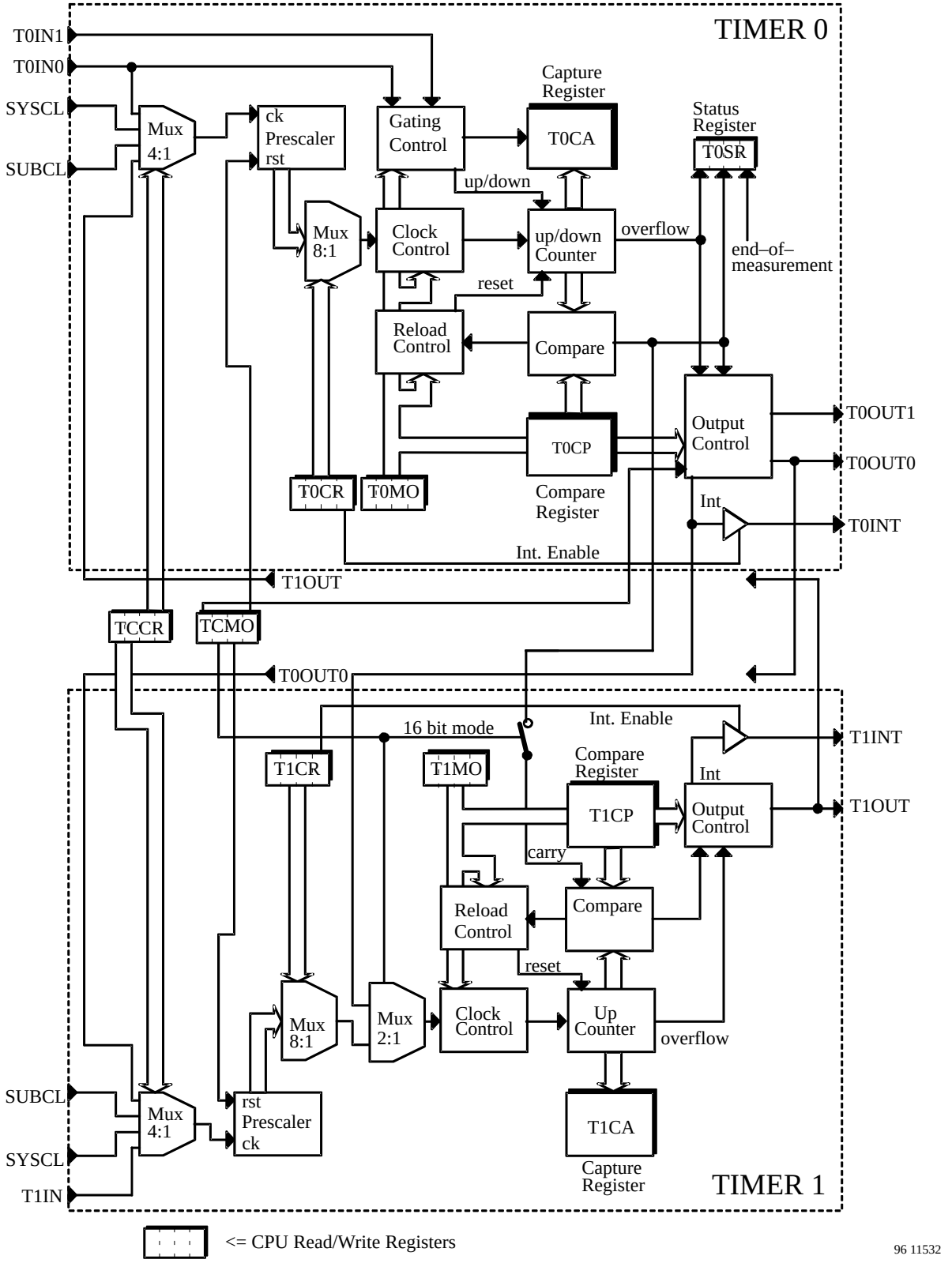


Figure 22. Timer/counter module

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 4. The Timer/Counter auxiliary register (TCX) 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.

Timer/Counter Clock Control Register (TCCR)

Subport address (indirect write access): ’6’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 12. 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 23).

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

	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 13. 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

	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 14. 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 23). 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.

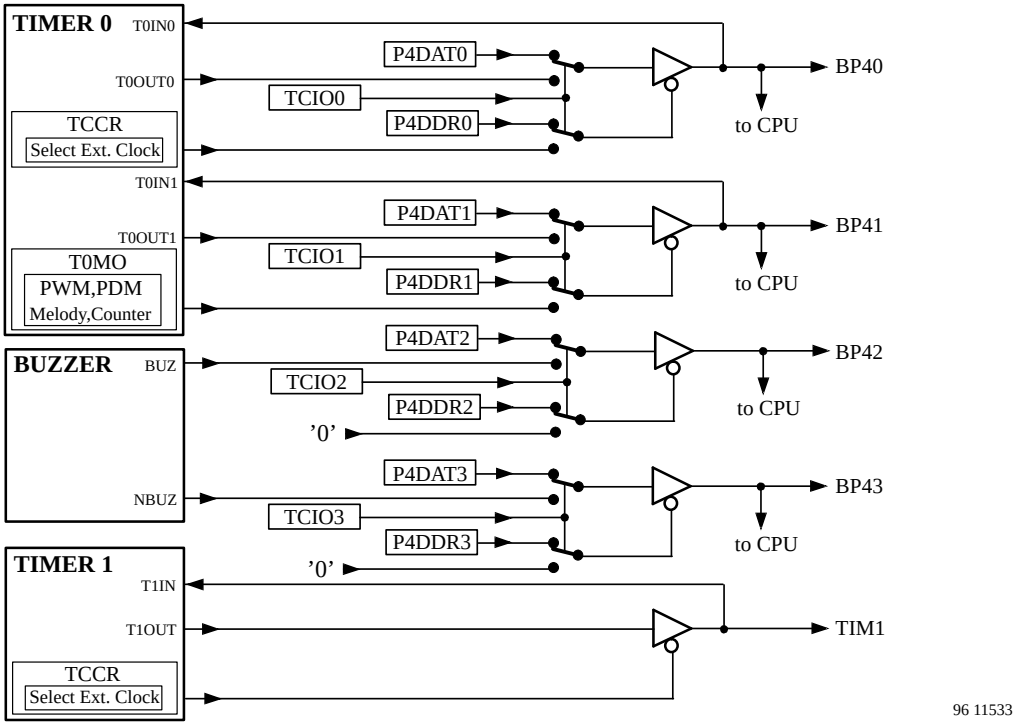


Figure 23. Timer/counter and buzzer external interface

Timer/Counter Mode Register (TCMO)

Subport address (indirect write access): '4'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

T1RST – Timer 1 Reset/Run

T0RST – Timer 0 Reset/Run

Table 15. Timer/Counter Mode Register (TCMO)

Code 3 2 1 0	Function
x x x 0	Timer 0 running
x x x 1	Timer 0 reset and halted
x x 0 x	Timer 1 running
x x 1 x	Timer 1 reset and 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

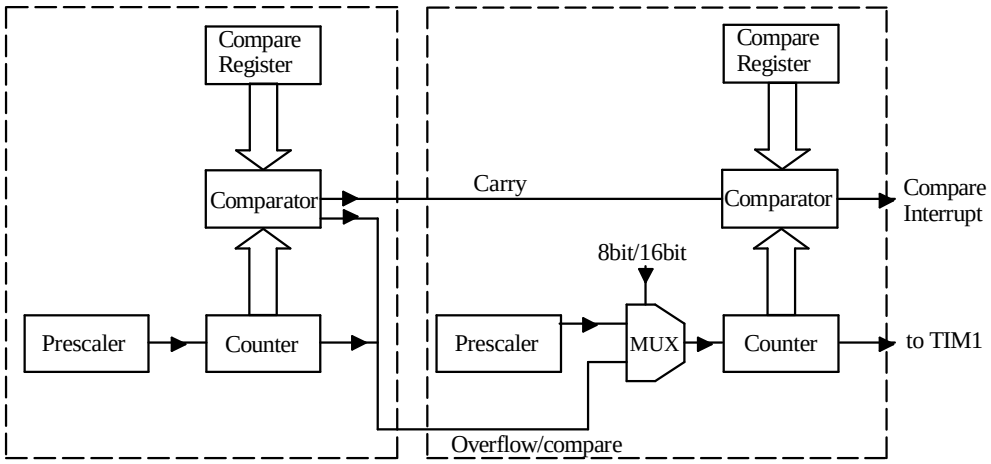


Figure 24. 16-bit mode

In 16-bit mode, Timer 0 and Timer 1 are cascaded thus forming a 16-bit counter (see figure 24) 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

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

T0MO3 ... 0 – Timer 0 Mode Code

Table 16. 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 17. 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 16, page 31.

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

	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 18. 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

*) Note: Emulation devices marked with C510 – 00x, with x = 4 ... 9, use a prescaler divide by 2.

Timer 0 Compare Register (T0CP) – Byte Write

Subport address (indirect write access): ’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

		Bit 7	Bit 6	Bit 5	Bit 4	
T0CA	**First read cycle**	**T0CA7**	**T0CA6**	**T0CA5**	**T0CA4**	**Reset value: 0000b**
		Bit 3	Bit 2	Bit 1	Bit 0	
	Second read cycle	**T0CA3**	**T0CA2**	**T0CA1**	**T0CA0**	**Reset value: 0000b**

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 25). 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.

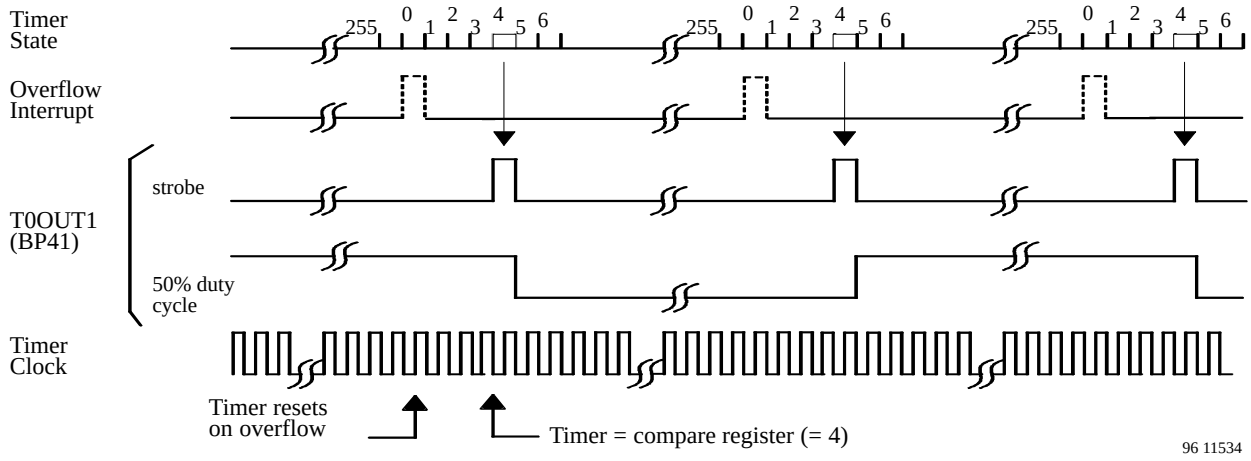


Figure 25. Timer 0 free running counter mode

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 26). 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).

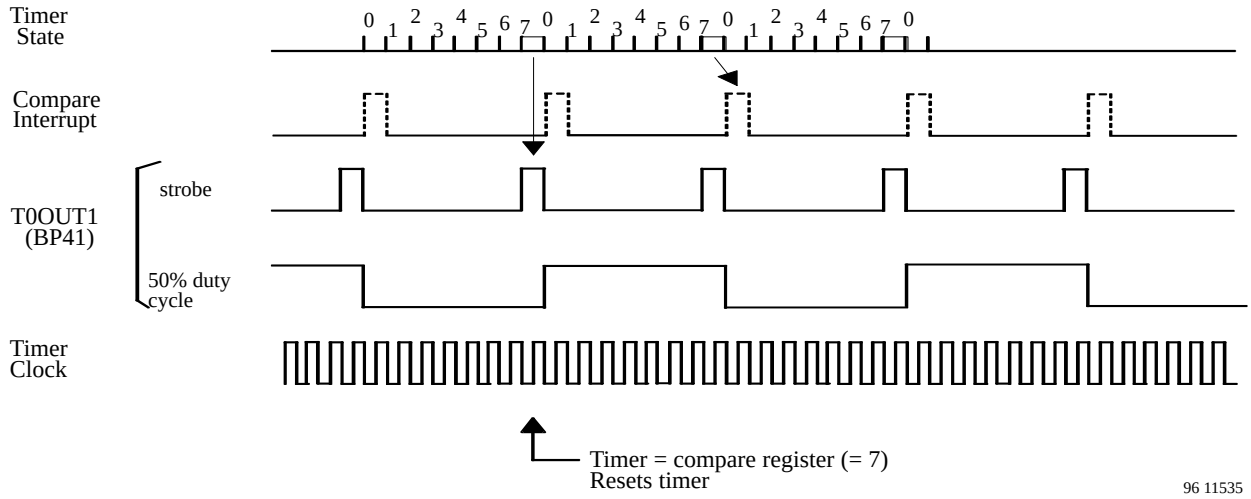
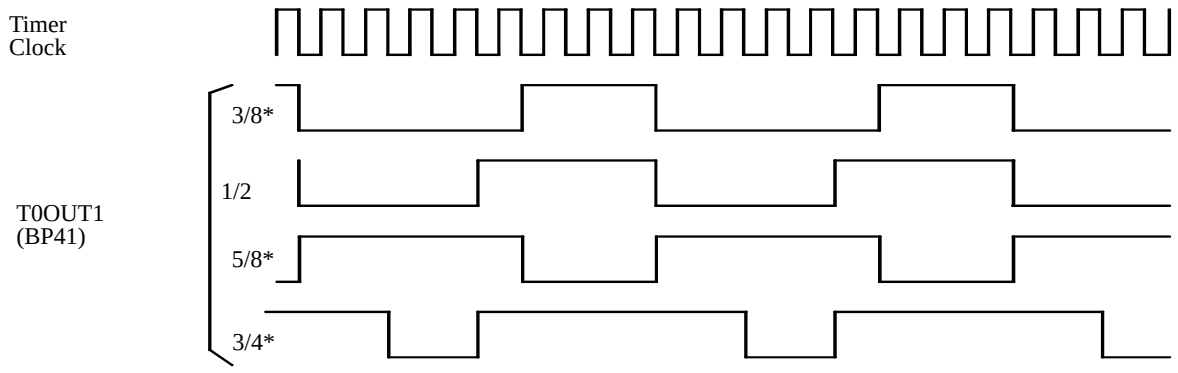


Figure 26. Timer 0 counter reload mode

Motor Chopping and Mask Options

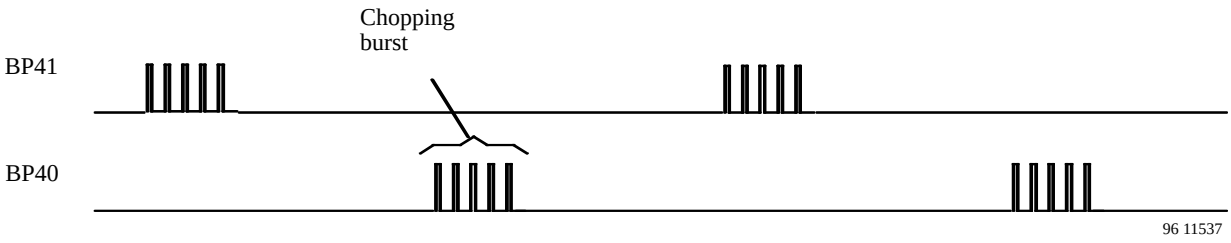
In the counter auto reload mode (50% duty cycle), mask options are available for generating a 1 kHz or 2 kHz frequency with duty cycles of 1/2, 3/8, 5/8 and 3/4. The resultant waveform is used as the chopping frequency for so called “motor chopping”. This technique allows the use of low cost, low voltage clock motors in applications where only higher supply voltages are available. The resultant voltage waveforms are shown in figure 27. To obtain the required motor driver waveforms on BP40 and BP41 as shown in figure 28, the user program must modulate the Timer 0 chopping frequency. This is performed by preloading Port 4 data latches (P4DAT0 and P4DAT1) with ’0’ which sets the normal Port 4 direction register bits to output mode (P4DDR0 = P4DDR1 = ’0’) and switches the TCIO0 and TCIO1 register bits alternately to ’0’ on every chopping burst. The timer chopping signals are thus transferred to the port outputs. In the intermediate periods between bursts both TCIO0 and TCIO1 are set to ’1’ and the preloaded Port 4 data latch outputs appear on the port outputs.



* = Mask Option
Timer 0 configuration reload mode, 50% duty cycle,
Comparator value = ’3’hex (1 kHz) or ’7’hex (2 kHz)
Timer clock = 32 kHz (prescaler bypassed)

96 11536

Figure 27. Motor chopping waveforms



96 11537

Figure 28. Motor driver output waveforms

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 M44C510 generates two output signals, a melody tone and an envelope pulse (see figure 29). 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 30). With the external capacitor connected, the resultant signal on BP41 exhibits a melody chime effect with an exponential decay.

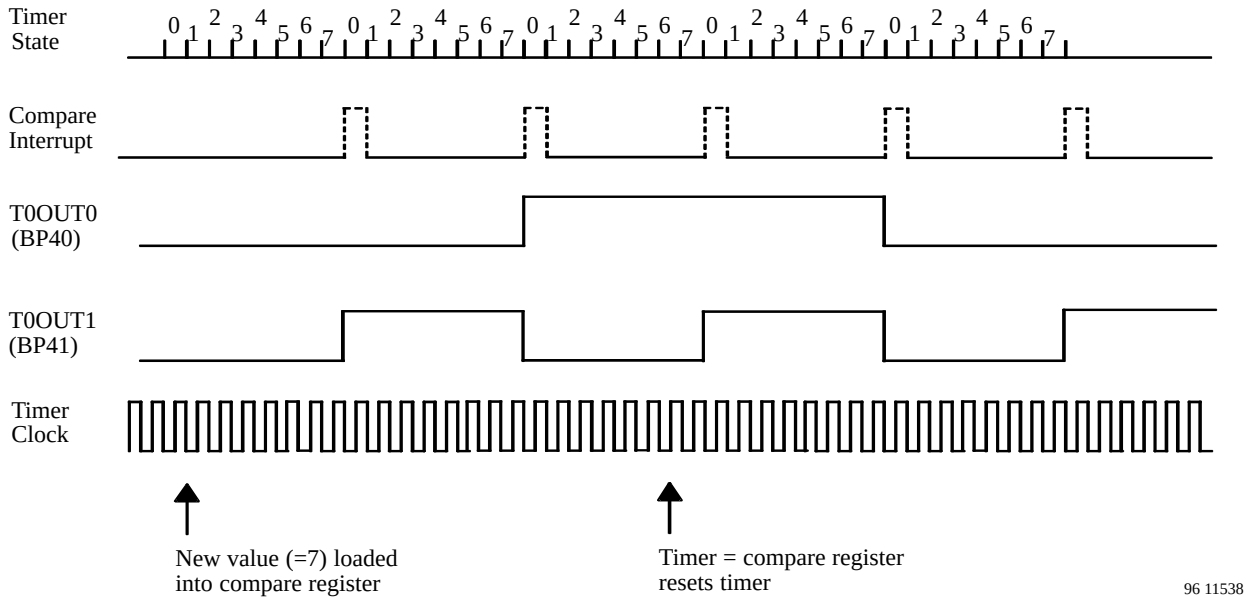


Figure 29. Modulated melody mode

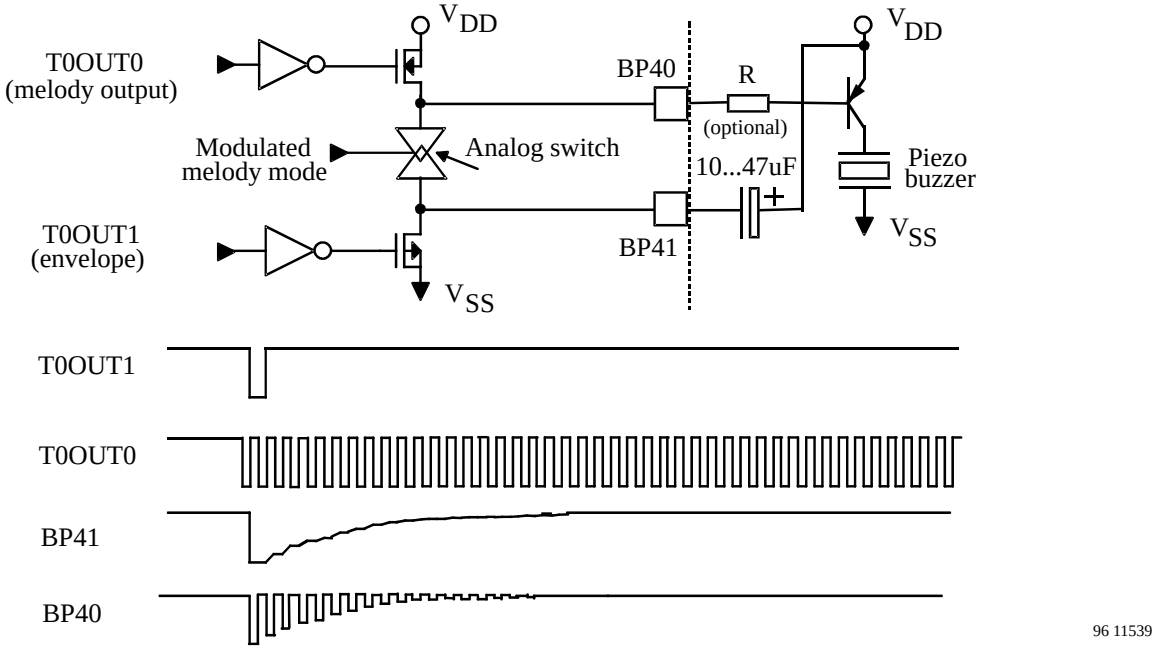


Figure 30. Modulated melody output circuit

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 DC analog value can be gained.

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 31). 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 clock periods, 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 mask options).

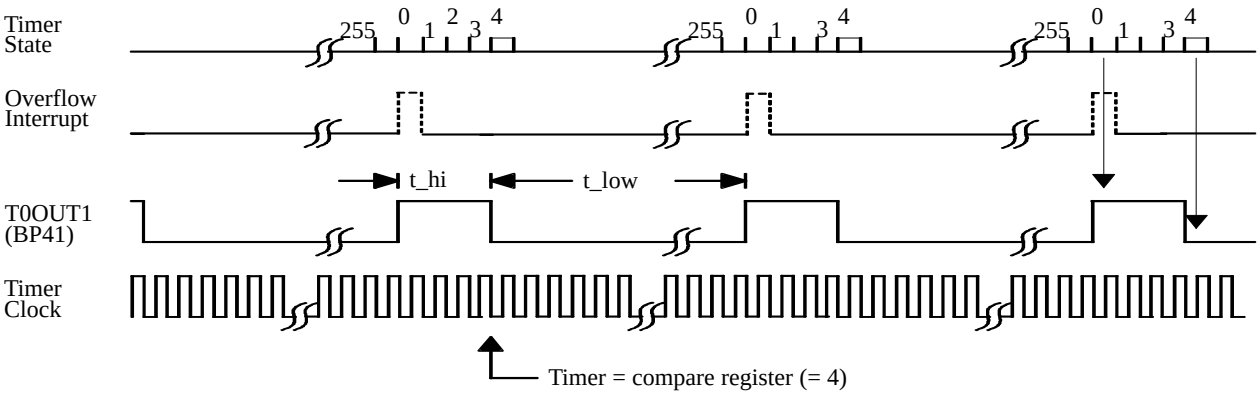


Figure 31. Timer 0 pulse width modulation

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 32). 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.

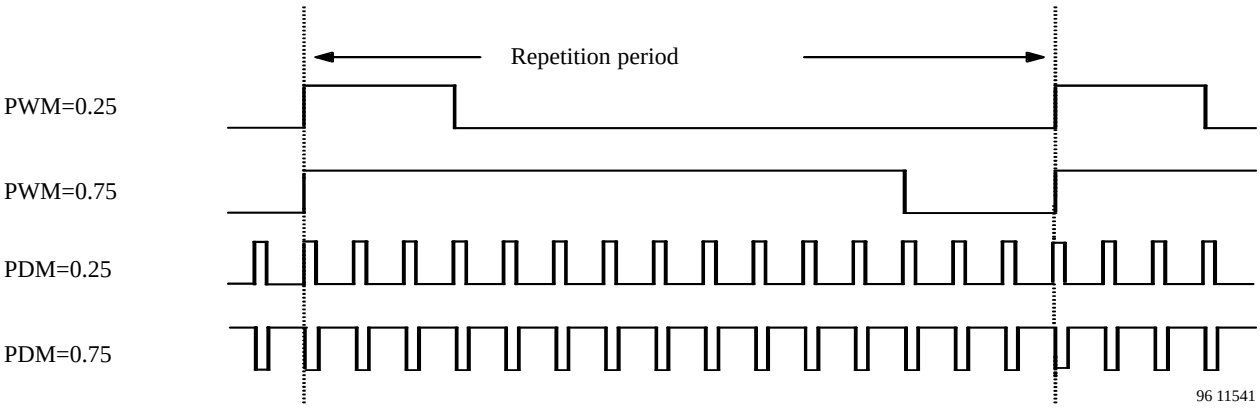


Figure 32. An example 4-bit PWM/PDM comparison

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 33). 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.

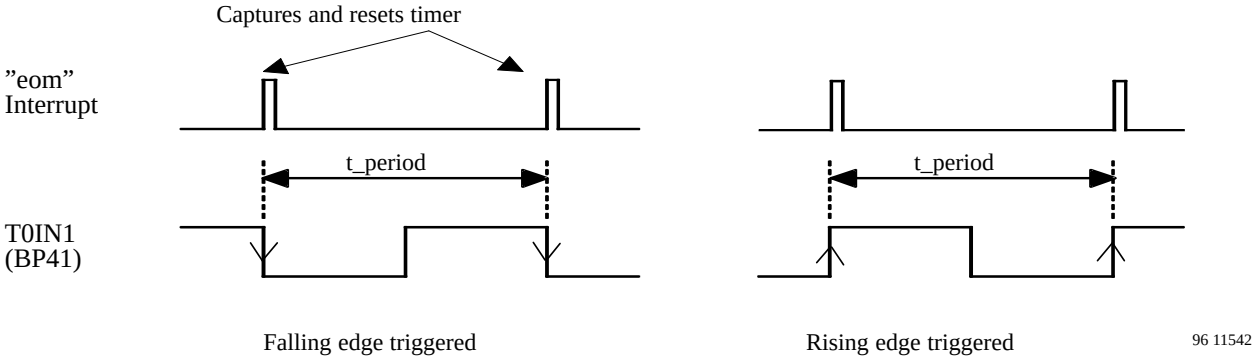


Figure 33. Period measurement

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 34). 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.

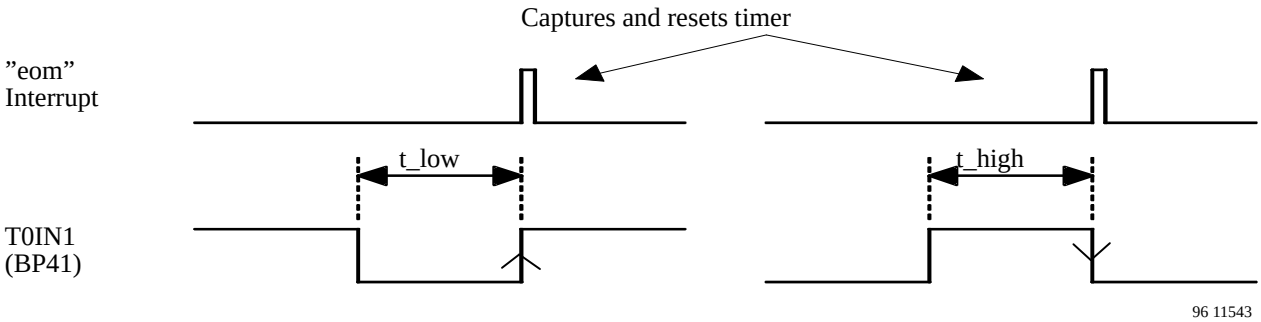


Figure 34. Pulse width measurement

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 35). 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.

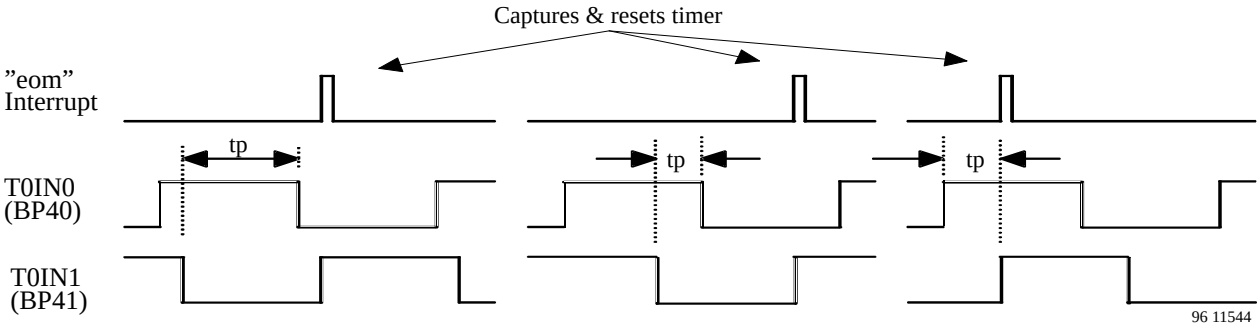


Figure 35. Phase measurement

Position Measurement Mode

This mode is intended for the evaluation of positional sensors with biphase output signals. Figure 36 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.

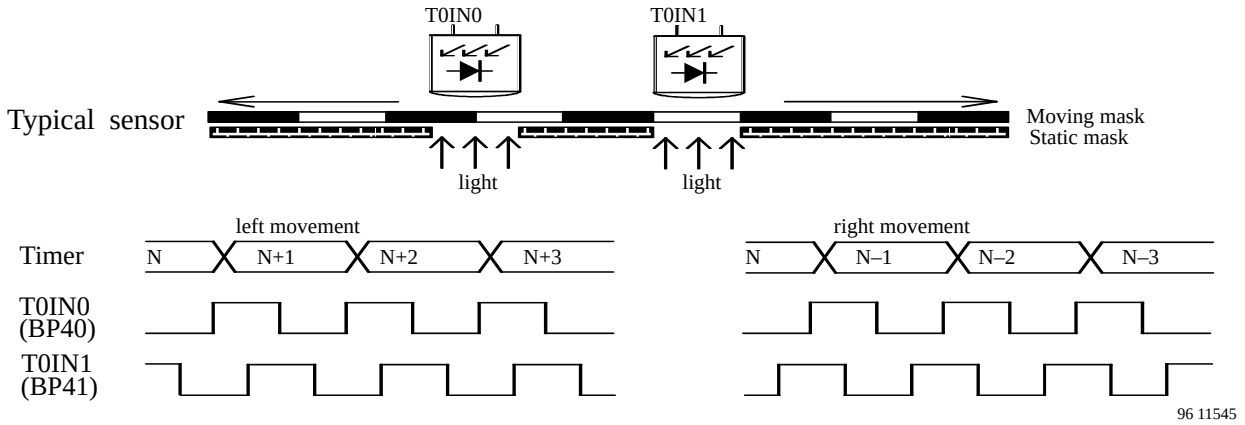


Figure 36. Position measurement mode

2.5.4 Timer 1 Modes

The Timer 1 is aimed at performing event counting and timing functions (see figure 22). 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.

Timer 1 Mode Register (T1MO)

Subport address (indirect write address): ’2’hex

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

T1MO3 ... 0 – Timer 1 Mode Control

Table 19. Timer 1 Mode Register (T1MO)

Code 3 2 1 0	Function	Compare Interrupt
1 x 0 0	Counter free running (50% duty cycle)	yes
1 x 0 1	Counter auto reload (50% duty cycle)	yes
1 x 1 0	Pulse width modulation	yes
1 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	–
0 x x x	reserved	–

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

	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 20. 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

		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

		Bit 7	Bit 6	Bit 5	Bit 4	
T1CA	**First read cycle**	**T1CA7**	**T1CA6**	**T1CA5**	**T1CA4**	**Reset value: 0000b**
		Bit 3	Bit 2	Bit 1	Bit 0	
	Second read cycle	**T1CA3**	**T1CA2**	**T1CA1**	**T1CA0**	**Reset value: 0000b**

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.

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 37).

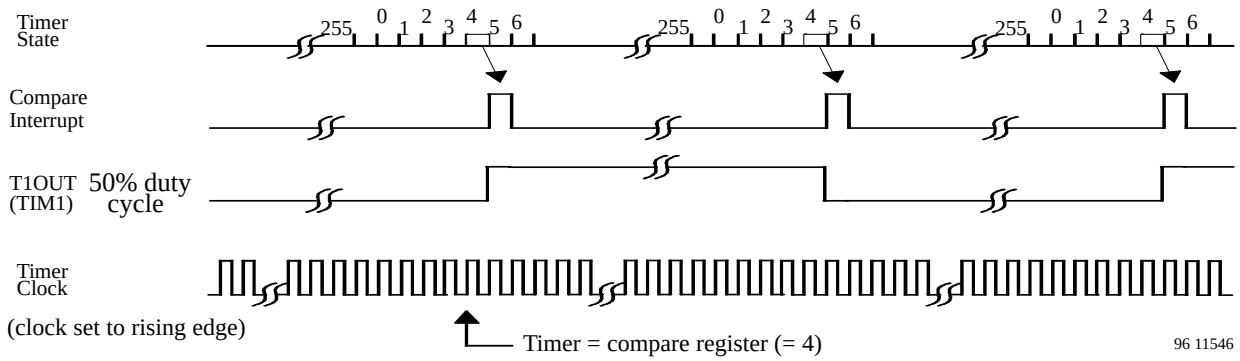


Figure 37. Timer 1 counter free running (50% duty cycle)

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 38). 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.

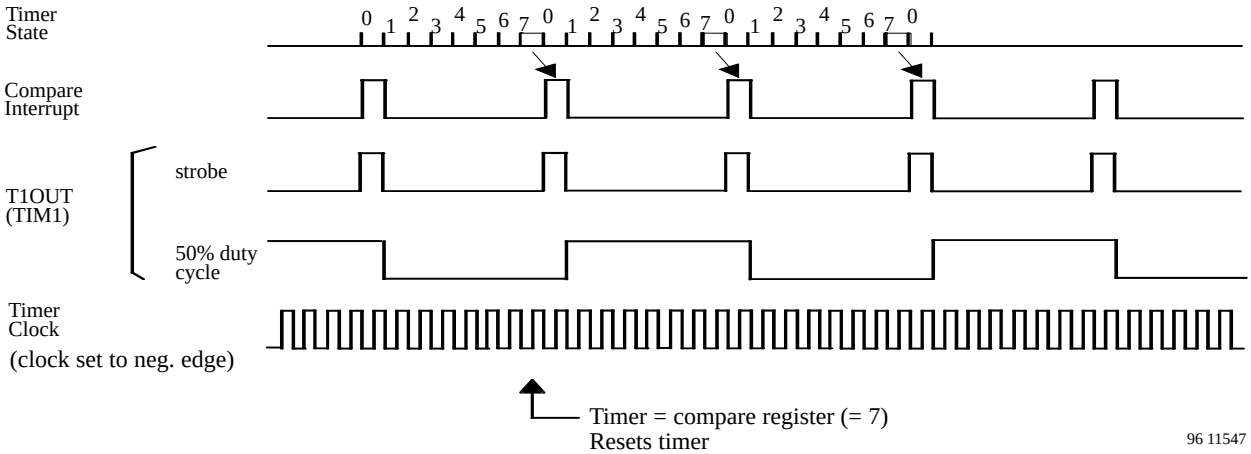


Figure 38. Timer 1 counter auto reload

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 39). 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 to a continuous high 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 the SYSCL clock is used as the PWM clock source because it will stop if the CPU goes into SLEEP.

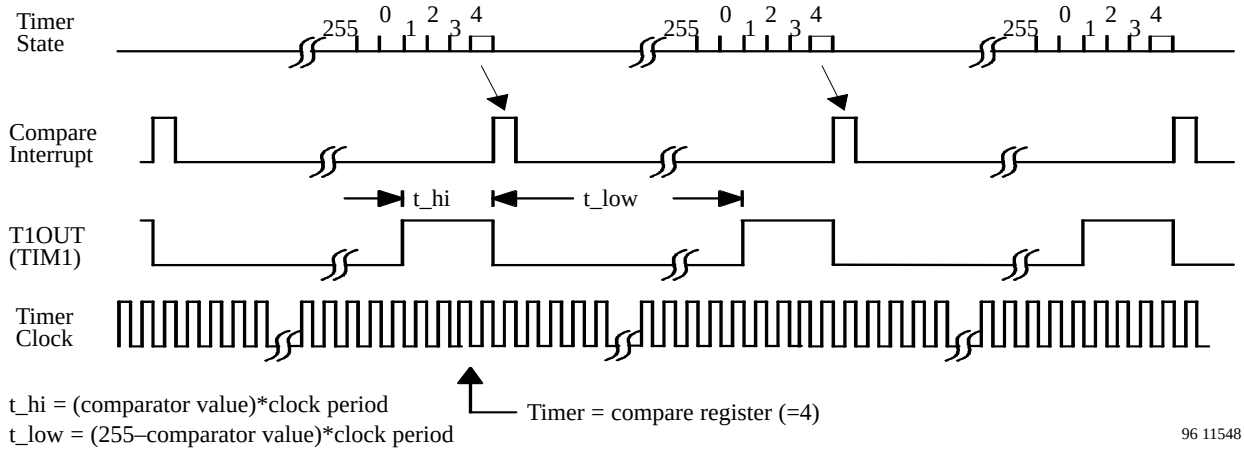


Figure 39. Timer 1 pulse width modulation

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 23. 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.

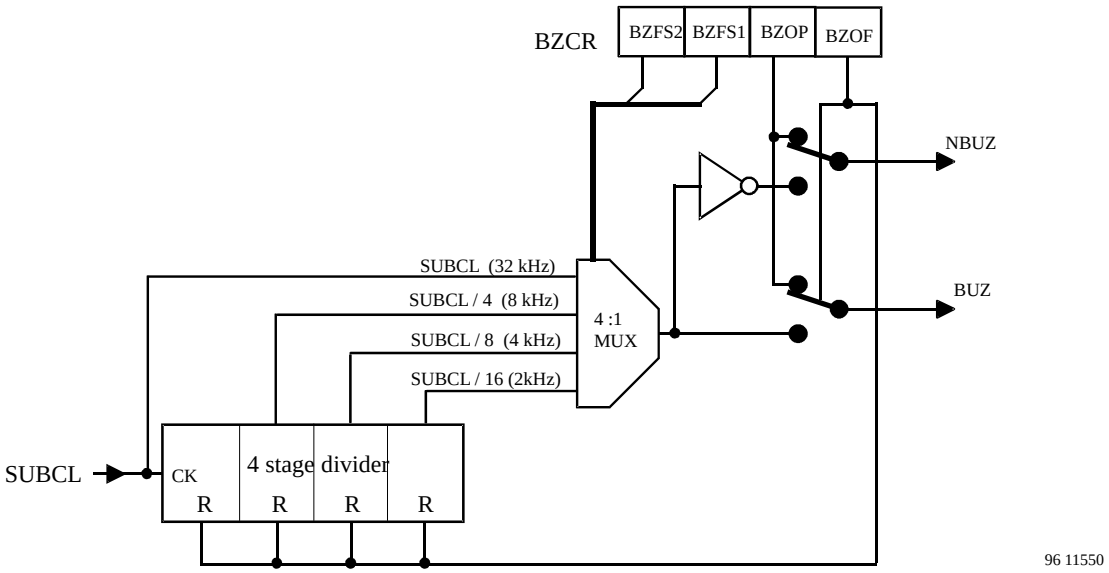


Figure 40. Buzzer module

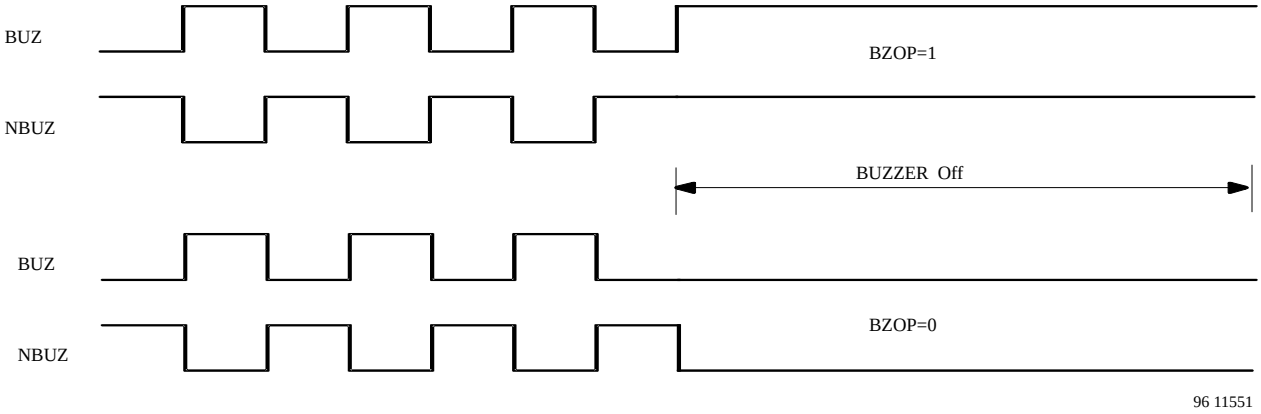


Figure 41. Buzzer waveform

Buzzer Control Register (BZCR)

Subport address (indirect write access): 'A'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 21. 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)

2.7 Emulation

All MARC4 controllers have a special emulation mode. It is activated by setting the TE pin to logic HIGH level after reset. In this mode, the internal CPU core is inactive and the I/O bus is available via port 0 and port 1 to allow the emulator the access to the on-chip peripherals. The emulator contains a special emulation CPU with a MARC4 core and additional breakpoint logic and takes over the core function. The basic function of the emulator is to evaluate the customer's program and hardware in real time.

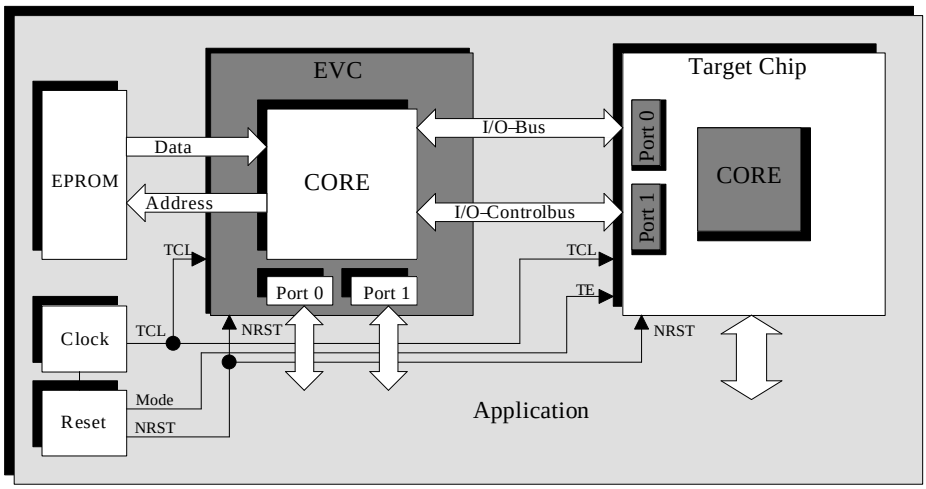


Figure 42. Emulation

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 + 6.5	V
Input voltage (on any pin)	V_{IN}	V_{SS} –0.3 ≤V_{IN} ≤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}	–40 to +130	°C
Thermal resistance (DIP40)	R_{thJA}	110	K/W
Soldering temperature (t ≤ 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 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} = 25°C unless otherwise specified.

Parameters	Test Conditions / Pins	Symbol	Min.	Typ.	Max.	Unit
Power supply						
Active current (CPU active, RC osc. with ext. R 200 kΩ)	f_{SYSCL} = 2 MHz V_{DD} = 2.4 V, Note 1 V_{DD} = 5.0 V V_{DD} = 6.2 V, Note 1	I_{DD}		 0.35 1.0 1.25		 mA mA mA
Power down current (CPU sleep, RC oscillator active, SUBCL = SYSCL/64)	f_{SYSCL} = 2 MHz V_{DD} = 2.4 V, Note 1 V_{DD} = 5.0 V V_{DD} = 6.2 V, Note 1	I_{PD}		 10 20 25		 µA µA µA
Sleep current (CPU sleep, SYSCL stopped, 32 kHz osc. active / inactive)	V_{DD} = 2.4 V, Note 1 V_{DD} = 5.0 V V_{DD} = 6.2 V, Note 1	I_{Sleep}		0.4 1.0 1.2	0.7 1.5 1.8	µA µA µA

Parameters	Test Conditions / Pins	Symbol	Min.	Typ.	Max.	Unit
Power-on reset threshold voltage						
POR threshold voltage		V_{POR}		1.8	2.0	V
POR hysteresis	Note 1	V_{POR}	0.25	0.3		V
Schmitt-trigger input voltage:						
Negative-going threshold voltage	V_{DD} = 2.4 to 6.2 V	V_{T-}	V_{SS}		0.3*V_{DD}	V
Positive-going threshold voltage	V_{DD} = 2.4 to 6.2 V	V_{T+}	0.7*V_{DD}		V_{DD}	V
Hysteresis (VT+ – VT–)	V_{DD} = 2.4 to 6.2 V	V_H		0.1*V_{DD}		

Note 1: Parameter not subject to production test

Input Pins: NRST and TE

Parameters	Test Conditions / Pins	Symbol	Min.	Typ.	Max.	Unit
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}	–100 –250	–125 –320	–150 –400	µA µA
Input TE with pull-down resistor						
Input HIGH current	V_{DD} = 2.4 V, V_{IH} = V_{DD} V_{DD} = 5.0 V	I_{IH}	15 200	30 260	50 300	µA µA

All Bidirectional Ports and TIM1

Parameters	Test Conditions / Pins	Symbol	Min.	Typ.	Max.	Unit
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 LOW current (pull-up)	V_{DD} = 2.4 V, V_{IL}= V_{SS} V_{DD} = 5.0 V	I_{IL}	–1.0 –6.0	–1.5 –8.5	–2.5 –12.0	µA µA
Input HIGH current (pull-down)	V_{DD} = 2.4 V, V_{IH} = V_{DD} V_{DD} = 5.0 V	I_{IH}	1.0 4.0	1.3	3.0 10.0	µA µA
Output LOW current DR = 1	V_{DD} = 2.4 V V_{OL} = 0.2*V_{DD} V_{DD} = 5.0 V	I_{OL}	0.7 2.8	0.9 3.5	1.1 4.2	mA mA
Output LOW current DR = 4	V_{DD} = 2.4 V V_{OL} = 0.2*V_{DD} V_{DD} = 5.0 V	I_{OL}	2.8 10.5	3.5 13.1	4.2 15.7	mA mA
Output LOW current DR = 12	V_{DD} = 2.4 V V_{OL} = 0.2*V_{DD} V_{DD} = 5.0 V	I_{OL}	6.8 28	8.5 34.4	10.2 40.8	mA mA
Output HIGH current DR = 1	V_{DD} = 2.4 V V_{OH} = 0.8*V_{DD} V_{DD} = 5.0 V	I_{OH}	–0.5 –1.8	–0.6 –2.2	–0.7 –2.6	mA mA
Output HIGH current DR = 4	V_{DD} = 2.4 V V_{OH} = 0.8*V_{DD} V_{DD} = 5.0 V	I_{OH}	–1.7 –6.1	–2.1 –7.6	–2.5 –9.1	mA mA
Output HIGH current DR = 12	V_{DD} = 2.4 V V_{OH} = 0.8*V_{DD} V_{DD} = 5.0 V	I_{OH}	–4.4 –16.5	–5.5 –20.5	–6.6 –24.7	mA mA

Bidirectional Port BPA0...BPA3, BPB0...BPB3

Parameters	Test Conditions / Pins	Symbol	Min.	Typ.	Max.	Unit
Input LOW current (30 k pull-up)	V_{DD} = 2.4 V, V_{IL}= V_{SS} V_{DD} = 5.0 V	I_{IL}	–20 –120	–27 –160	–40 –200	µA µA

Bidirectional Port BP60 and BP61 (INTx, INTy)

Parameters	Test Conditions / Pins	Symbol	Min.	Typ.	Max.	Unit
Input LOW current (2 k pull-up)	V_{DD} = 2.4 V, V_{IL}= V_{SS} V_{DD} = 5.0 V	I_{IL}	–0.2 –1.4	–0.35 –1.7	–0.65 –2.5	mA mA

3.3 AC Characteristics

Supply voltage V_{DD} = 2.4 to 6.2 V, V_{SS} = 0 V, T_{amb} = 25°C unless otherwise specified.

Parameters	Test Conditions / Pins	Symbol	Min.	Typ.	Max.	Unit
Timer input timing TIM1, BP40 and BP41						
Timer input clock		f_{TIMx}		4	10	MHz
Timer input LOW time	Rise / fall time < 10 ns	t_{TIL}	50			ns
Timer input HIGH time	Rise / fall time < 10 ns	t_{TIH}	50			ns
Interrupt request input timing						
Int. request LOW time	Rise / fall time < 10 ns	t_{IRL}	50			ns
Int. request HIGH time	Rise / fall time < 10 ns	t_{IRH}	50			ns
System clock						
SCLIN input clock	Rise / fall time < 10 ns	f		4	10	MHz
Start-up time	f_x = 4 MHz, V_{DD} = 3.0 V	t_{SX}		10	20	ms
Reset timing						
Power-on reset time	$V_{DD} > V_{POR}$	T_{POR}		200	500	µs
NRST input LOW time		T_{NRST}	4*SYSCL			µs
RC oscillator – external resistor						
Frequency	Note 1; R_{ext} = 200 kΩ	f_{RCe}	1.8	2.0	2.2	MHz
Stability	Note 1; V_{DD} = 3 to 5.5 V	Δf/f			± 5	%
32-kHz oscillator						
Start-up time	AV_{DD} = 3.0 V	t_{SQ}		0.5	1	s
Stability	Note 2; ΔAV_{DD} = 100 mV	Δf/f		0.1		ppm
Integrated input / output capacitances		C_{IN} C_{OUT}		20 20		pF pF

Crystal Characteristics

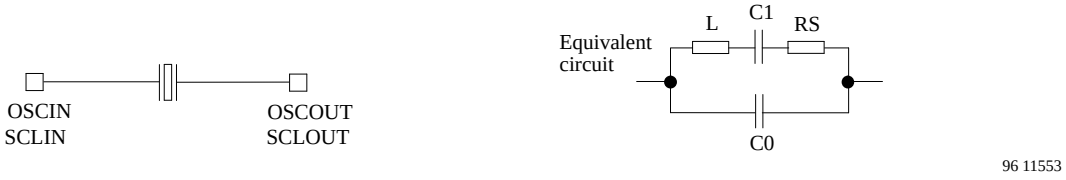
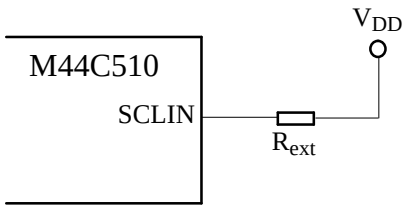


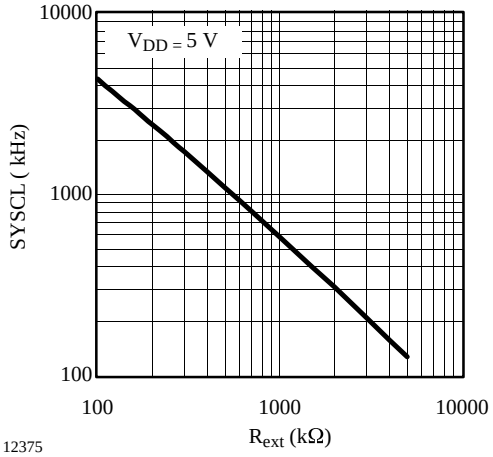
Figure 43. Crystal equivalent circuit

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		10	12.5	pF
4 MHz crystal						
Crystal frequency		f_X		4	4.192	MHz
Series resistance		RS		30	50	Ω
Static capacitance		C0		2	4.5	pF
Dynamic capacitance		C1		3	15	fF



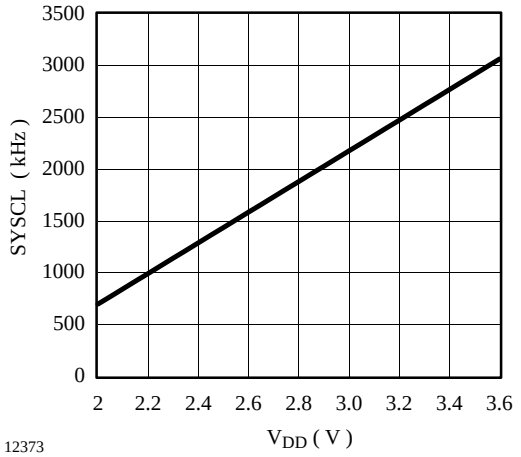
96 12372

Figure 44. Clock generation with external resistor



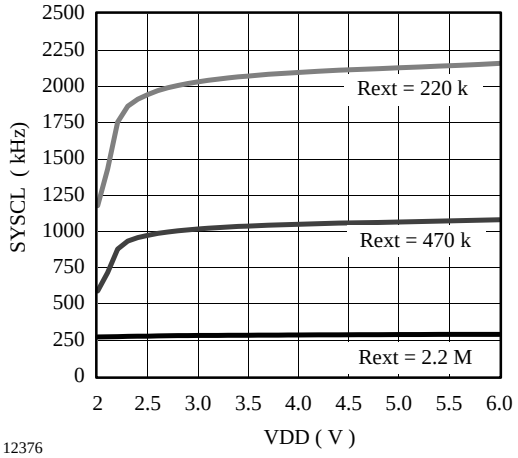
96 12375

Figure 47. SYSCL = f (R_{ext})



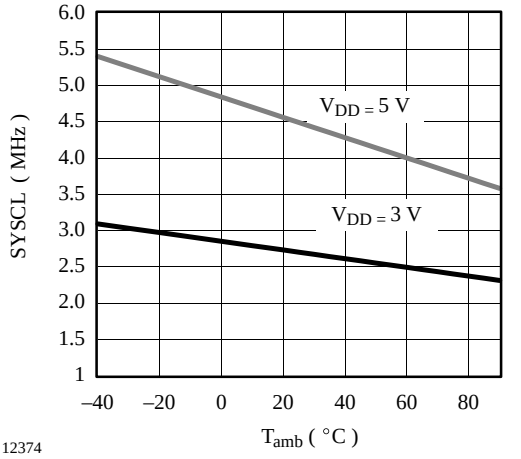
96 12373

Figure 45. Internal RC–oscillator frequency = f (V_{DD})



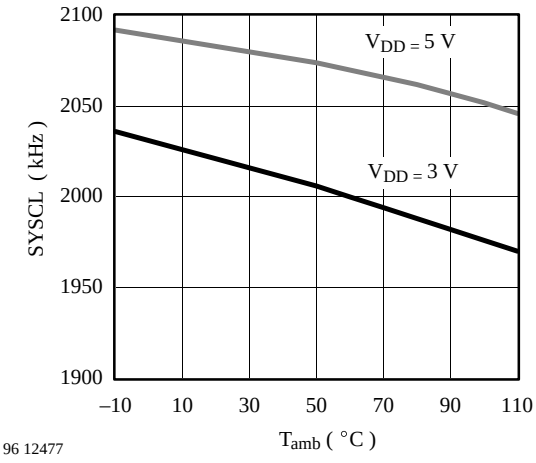
96 12376

Figure 48. SYSCL = f (V_{DD}, R_{ext})



96 12374

Figure 46. Internal RC–oscillator frequency = f (T_{amb})



96 12477

Figure 49. SYSCL = f (T_{amb}); R_{ext} = 220k

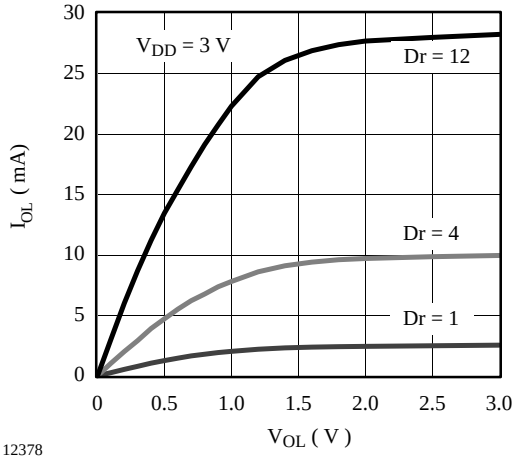


Figure 50. Typical low output driver

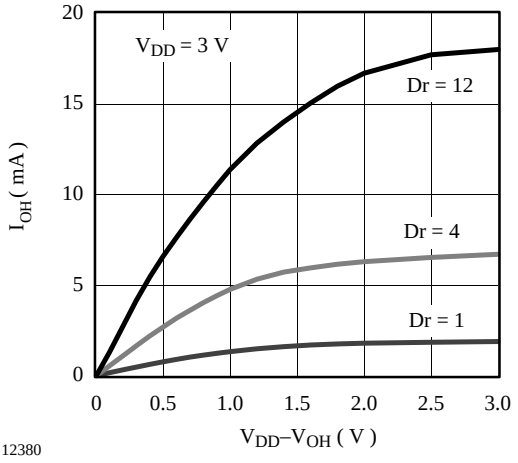


Figure 53. Typical high output driver

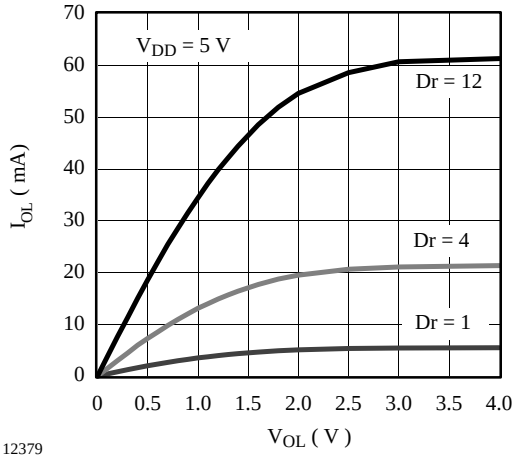


Figure 51. Typical low output driver

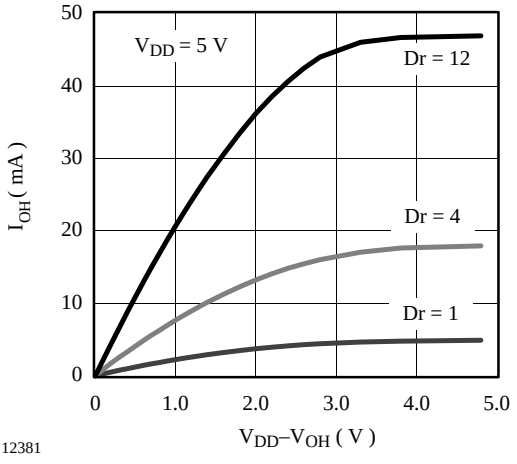


Figure 54. Typical high output driver

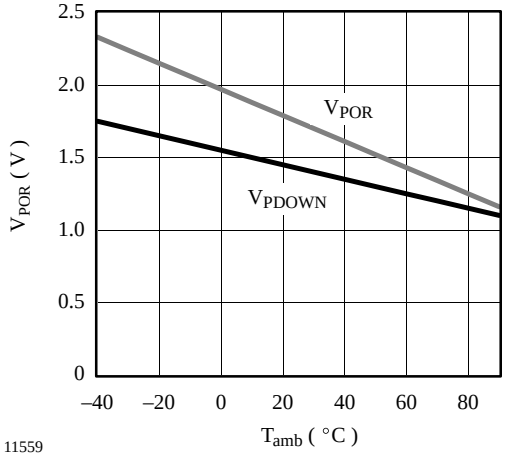


Figure 52. Power-on hysteresis = f (T_{amb})

4 Pad Layout

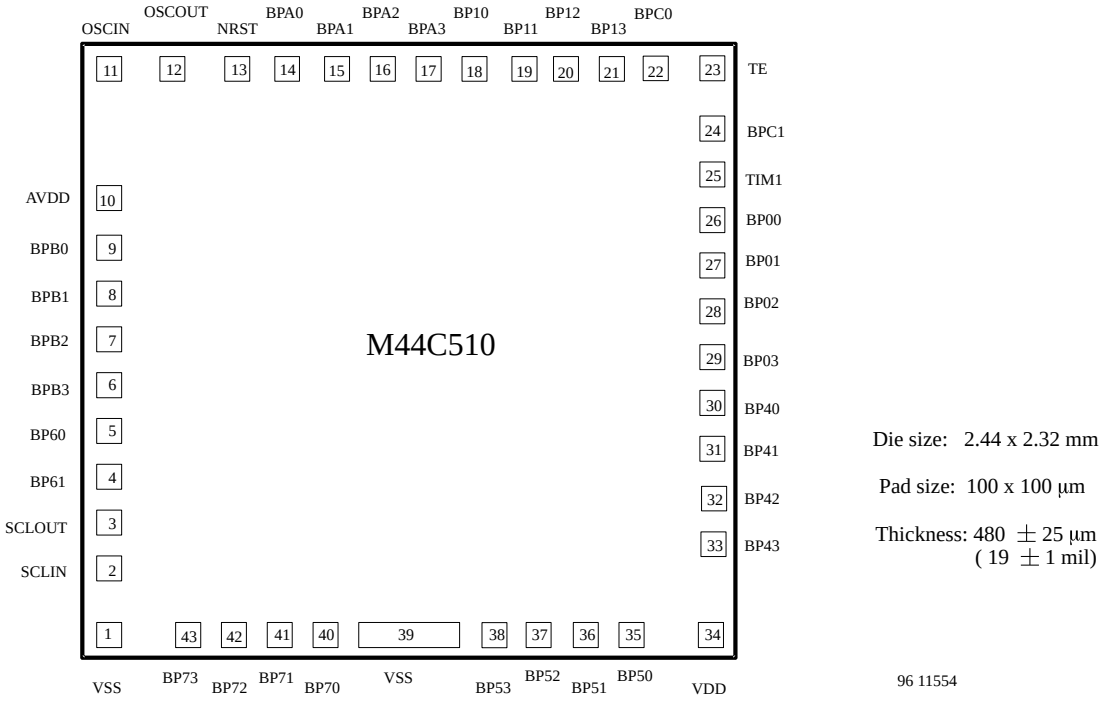


Figure 55. Pad assignments

Table 22. Pad coordinates

Pad No.	Name	X-Coord	Y-Coord		Pad No.	Name	X-Coord	Y-Coord
1	VSS	0.0	0.0		22	BPC0	1863.1	1931.6
2	SCLIN	0.0	202.6		23	TE	2093.0	1931.6
3	SCLOUT	0.0	352.6		24	BPC1	2103.2	1675.8
4	BP61	0.0	502.6		25	TIM1	2103.2	1525.8
5	BP60	0.0	652.6		26	BP00	2103.2	1375.8
6	BPB3	0.0	802.6		27	BP01	2103.2	1225.8
7	BPB2	0.0	952.6		28	BP02	2103.2	1075.8
8	BPB1	0.0	1102.6		29	BP03	2103.2	925.8
9	BPB0	0.0	1252.6		30	BP40	2103.2	757.5
10	AVDD	0.0	1402.6		31	BP41	2103.2	607.5
11	OSCIN	9.8	1931.6		32	BP42	2103.2	457.5
12	OSCOUT	263.0	1931.6		33	BP43	2103.2	307.0
13	NRST	513.1	1931.6		34	VDD	2103.2	0.0
14	BPA0	663.1	1931.6		35	BP50	1705.2	0.0
15	BPA1	813.1	1931.6		36	BP51	1555.2	0.0
16	BPA2	963.1	1931.6		37	BP52	1405.2	0.0
17	BPA3	1113.1	1931.6		38	BP53	1255.2	0.0
18	BP10	1263.1	1931.6		39	VSS	923.5	0.0
19	BP11	1413.1	1931.6		40	BP70	755.5	0.0
20	BP12	1563.1	1931.6		41	BP71	605.5	0.0
21	BP13	1713.1	1931.6		42	BP72	455.5	0.0
					43	BP73	305.5	0.0

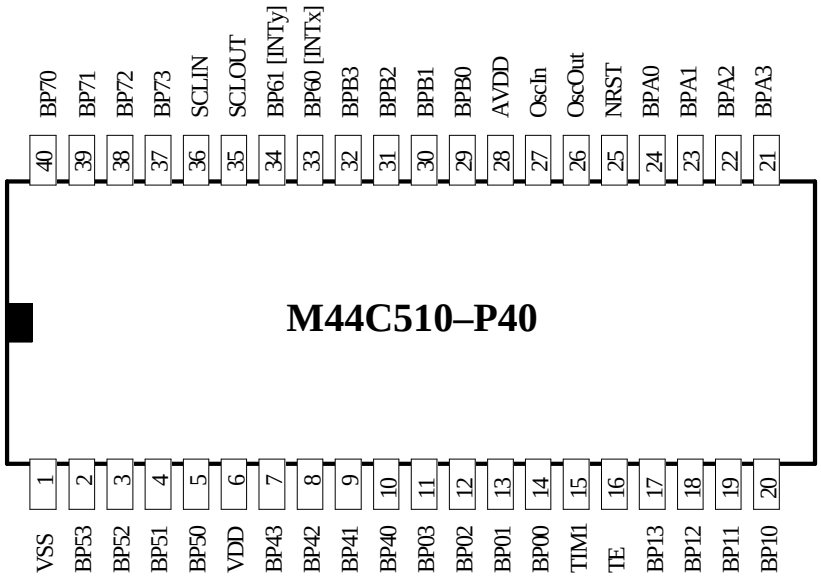
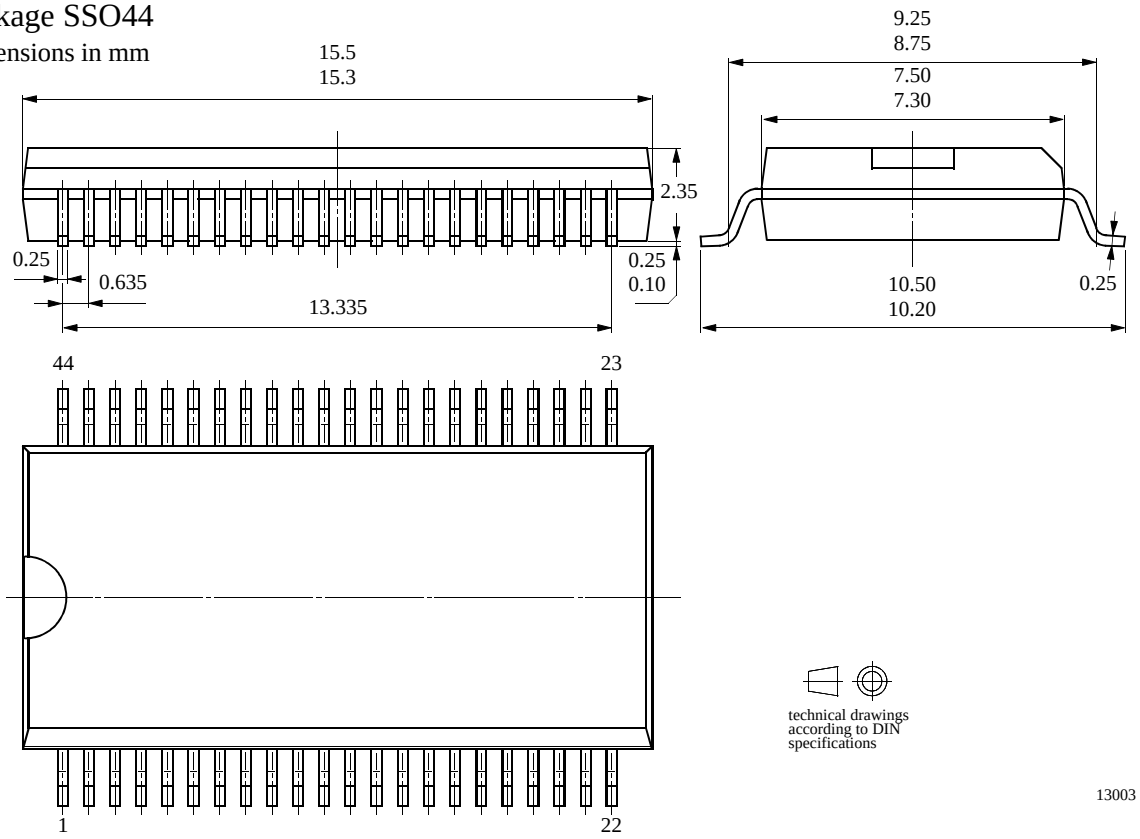


Figure 56. Pin connections for DIP 40

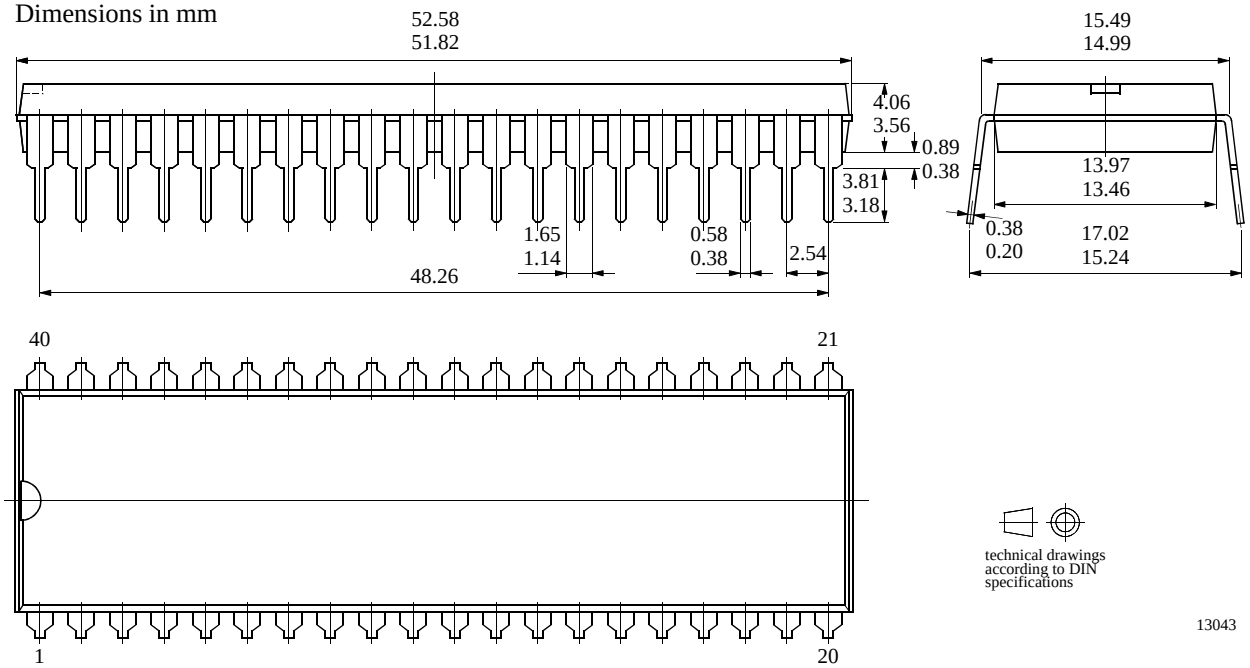
Package SSO44

Dimensions in mm



Package DIP40 (CEI)

Dimensions in mm



5 Application Examples

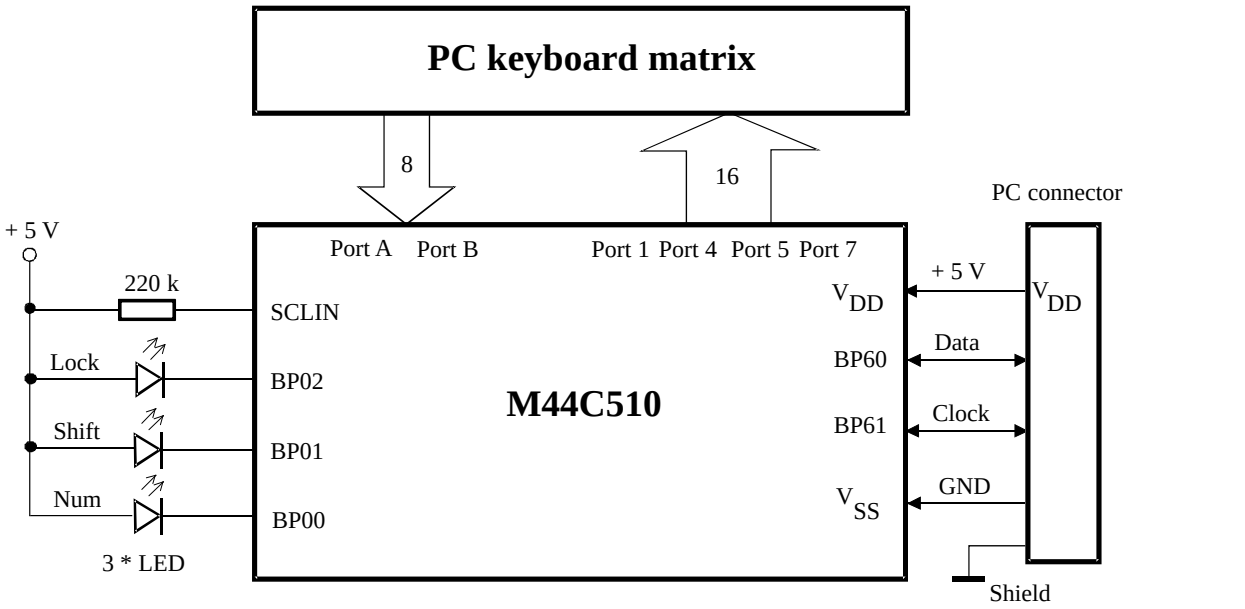


Figure 57. M44C510 as keyboard controller

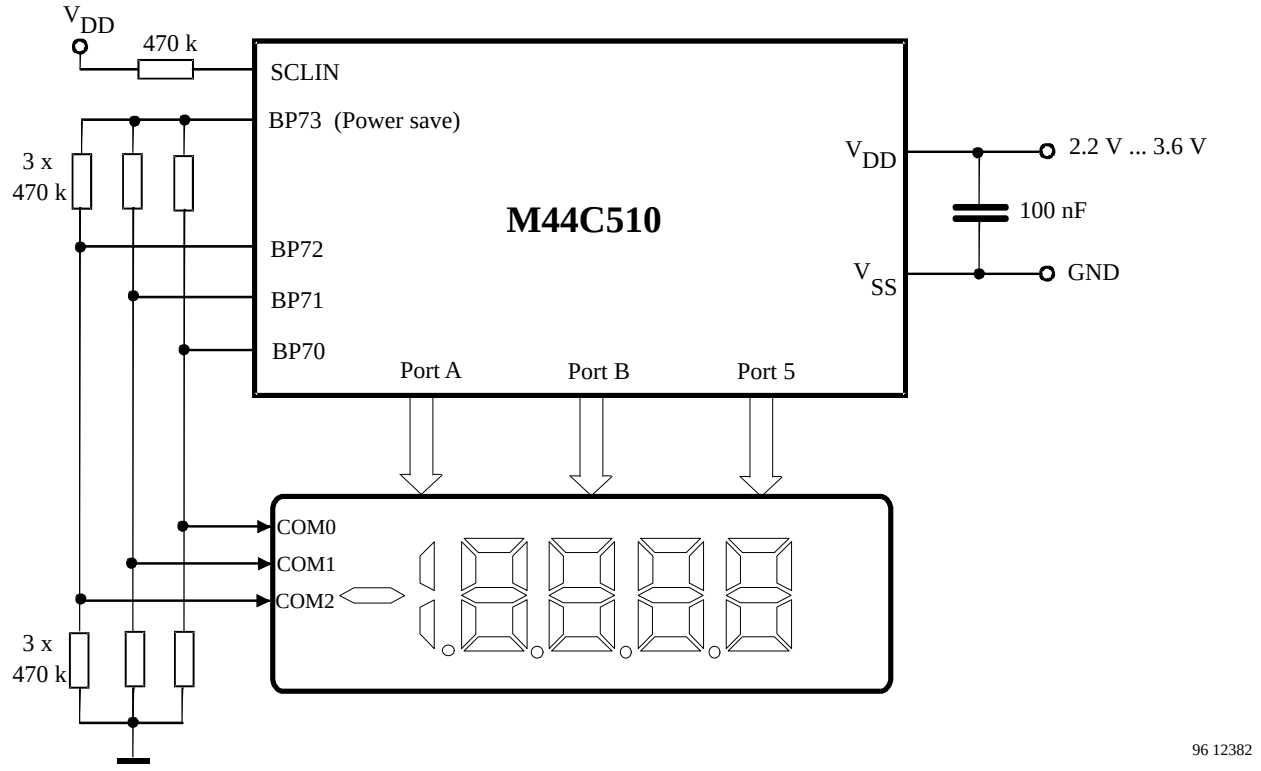


Figure 58. Driving a LCD panel with 1/3 duty

6 Ordering Information

Please select the option setting from the list below and insert ROM CRC. DR means driver ratio (= mA @ 3 V) and can be choosen from 1 to 12, however, the hole port must have the same value.

Port 0 DR =_____

BP00	■ CMOS	❑ Pull-up
		❑ Pull-down
BP01		❑ Pull-up
		❑ Pull-down
BP02		❑ Pull-up
		❑ Pull-down
BP03		❑ Pull-up
		❑ Pull-down

Port 1 DR =_____

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 DR =_____

BP40	❑ CMOS	❑ Pull-up
	❑ Open drain [N]	❑ Pull-down
	❑ Open drain [P]	
BP41	❑ CMOS	❑ Pull-up
	❑ Open drain [N]	❑ Pull-down
	❑ Open drain [P]	
BP42	❑ CMOS	❑ Pull-up
	❑ Open drain [N]	❑ Pull-down
	❑ Open drain [P]	
BP43	❑ CMOS	❑ Pull-up
	❑ Open drain [N]	❑ Pull-down
	❑ Open drain [P]	

Port 5 DR =_____

BP50	❑ CMOS	❑ Pull-up
	❑ Open drain [N]	❑ Pull-down
	❑ Open drain [P]	
BP51	❑ CMOS	❑ Pull-up
	❑ Open drain [N]	❑ Pull-down
	❑ Open drain [P]	
BP52	❑ CMOS	❑ Pull-up
	❑ Open drain [N]	❑ Pull-down
	❑ Open drain [P]	
BP53	❑ CMOS	❑ Pull-up
	❑ Open drain [N]	❑ Pull-down
	❑ Open drain [P]	

Port 6 DR =_____

BP60	❑ CMOS	❑ Pull-up
	❑ Open drain [N]	❑ Pull-down
	❑ Open drain [P]	❑ Pull-up (2 k)
BP61	❑ CMOS	❑ Pull-up
	❑ Open drain [N]	❑ Pull-down
	❑ Open drain [P]	❑ Pull-up (2 k)

Port 7 DR =_____

BP70	❑ CMOS	❑ Pull-up
	❑ Open drain [N]	❑ Pull-down
	❑ Open drain [P]	
BP71	❑ CMOS	❑ Pull-up
	❑ Open drain [N]	❑ Pull-down
	❑ Open drain [P]	
BP72	❑ CMOS	❑ Pull-up
	❑ Open drain [N]	❑ Pull-down
	❑ Open drain [P]	
BP73	❑ CMOS	❑ Pull-up
	❑ Open drain [N]	❑ Pull-down
	❑ Open drain [P]	

Port C DR =_____

BPC0	❑ CMOS	❑ Pull-up
	❑ Open drain [N]	❑ Pull-down
	❑ Open drain [P]	
BPC1	❑ CMOS	❑ Pull-up
	❑ Open drain [N]	❑ Pull-down
	❑ Open drain [P]	

Port A DR =_____

BPA0 ❑ CMOS ❑ Pull-up
❑ Open drain [N] ❑ Pull-down
❑ Open drain [P] ❑ Pull-up (30 k)

BPA1 ❑ CMOS ❑ Pull-up
❑ Open drain [N] ❑ Pull-down
❑ Open drain [P] ❑ Pull-up (30 k)

BPA2 ❑ CMOS ❑ Pull-up
❑ Open drain [N] ❑ Pull-down
❑ Open drain [P] ❑ Pull-up (30 k)

BPA3 ❑ CMOS ❑ Pull-up
❑ Open drain [N] ❑ Pull-down
❑ Open drain [P] ❑ Pull-up (30 k)

Port B DR =_____

BPB0 ❑ CMOS ❑ Pull-up
❑ Open drain [N] ❑ Pull-down
❑ Open drain [P] ❑ Pull-up (30 k)

BPB1 ❑ CMOS ❑ Pull-up
❑ Open drain [N] ❑ Pull-down
❑ Open drain [P] ❑ Pull-up (30 k)

BPB2 ❑ CMOS ❑ Pull-up
❑ Open drain [N] ❑ Pull-down
❑ Open drain [P] ❑ Pull-up (30 k)

BPB3 ❑ CMOS ❑ Pull-up
❑ Open drain [N] ❑ Pull-down
❑ Open drain [P] ❑ Pull-up (30 k)

TIM1 DR =_____

❑ CMOS
❑ Open drain [N] ❑ Pull-up
❑ Open drain [P] ❑ Pull-down

BPA–Reset
❑ No
❑ BPA0 & BPA1
❑ BPA0 & BPA1 & BPA2
❑ BPA0 & BPA1 & BPA2 & BPA3

Watchdog
❑ $^1/_2$ s ❑ Disabled
❑ 1 s
❑ 2 s

SYSCL Type
❑ R extern
❑ RC intern
❑ 4 MHz crystal oscillator
❑ 4 MHz ceramic resonator

SLEEP CLK
❑ SYSCL running
❑ SYSCL stopped

SUBCLK
❑ SYSCL / 64
❑ 32 kHz crystal

OSCIN
❑ No integrated capacitance
❑ Internal CAP (_ pF)

OSCOUT
❑ No intergrated capacitance
❑ Internal CAP (_ pF)

SCLIN
❑ No integrated capacitance
❑ Internal CAP (_ pF)

SCLOUT
❑ No integrated capacitance
❑ Internal CAP (_ pF)

Package
❑ DIT
❑ PDIL40
❑ SSO44
❑ SO28
❑ SO20

File:_____. HEX CRC: _____ HEX

Approval Date: ____–____–____ Signature: _____

We reserve the right to make changes to improve technical design 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 TEMIC products for any unintended or unauthorized application, the buyer shall indemnify TEMIC 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.

TEMIC TELEFUNKEN microelectronic GmbH, P.O.B. 3535, D-74025 Heilbronn, Germany
Telephone: 49 (0)7131 67 2831, Fax Number: 49 (0)7131 67 2423