User's Manual, V 1.0, January 2003

C868

8 - Bit CMOS Microcontroller

Microcontrollers

Infineon technologies

Never stop thinking.

Edition 2003-01

Published by Infineon Technologies AG,
St.-Martin-Strasse 53,
D-81541 München, Germany

Attention please!

The information herein is given to describe certain components and shall not be considered as warranted characteristics.

Terms of delivery and rights to technical change reserved.

We hereby disclaim any and all warranties, including but not limited to warranties of non-infringement, regarding circuits, descriptions and charts stated herein.

Infineon Technologies is an approved CECC manufacturer.

Information

For further information on technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies Office in Germany or our Infineon Technologies Representatives worldwide (see address list).

Warnings

Due to technical requirements components may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies Office.

Infineon Technologies Components may only be used in life-support devices or systems with the express written approval of Infineon Technologies, if a failure of such components can reasonably be expected to cause the failure of that life-support device or system, or to affect the safety or effectiveness of that device or system. Life support devices or systems are intended to be implanted in the human body, or to support and/or maintain and sustain and/or protect human life. If they fail, it is reasonable to assume that the health of the user or other persons may be endangered.

C868

8 - Bit CMOS Microcontroller

Microcontrollers

Never stop thinking.

Revision History: 2003-01

Previous Version: v 0.2

Page	Subjects (major changes since last revision)
	Added description of I2C in chapter 9

We Listen to Your Comments

Any information within this document that you feel is wrong, unclear or missing at all? Your feedback will help us to continuously improve the quality of this document. Please send your proposal (including a reference to this document) to:

mcdocu.comments@infineon.com

1 Introduction

The C868 is a member of the Infineon Technologies C800 family of 8-bit microcontrollers. It is fully compatible to the standard 8051 microcontroller. Its features include the capture compare unit (CCU6) which is useful in motor control applications, extended power saving provisions, on-chip RAM and RFI related improvements. The C868 has a maximum CPU clock rate of 40MHz. At 40MHz it achieves a 300ns instruction cycle time.

The C868 basically operates with internal program memory only. The C868-1R contains 8K x 8 on-chip ROM of program memory and the C868-1S contains 8K x 8 of on-chip RAM of program memory. Different operating modes are provided to allow flexibility in the access of the different types of memory. An additional RAM, XRAM, is provided for the implementation of in-system programming.

Figure 1-1 shows the different functional units of the C868 and **Figure 1-2** shows the simplified logic symbol of the C868.

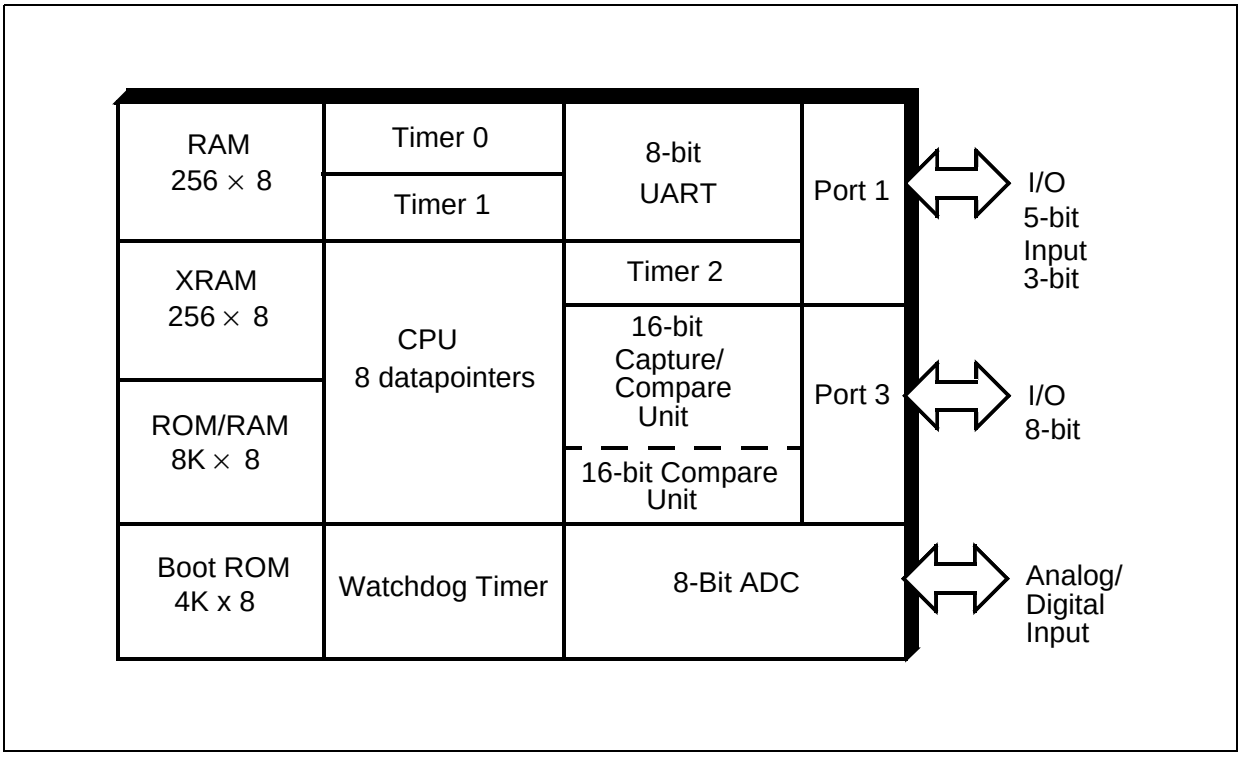


Figure 1-1 C868 Functional Units

1.1 Summary of Basic Features

Listed below is a summary of the main features of the C868:

- C800 core :
  –Fully compatible to standard 8051 microcontroller
  –Superset of the 8051 architecture with 8 datapointers
- 6.25 - 40 MHz internal system clock (built-in PLL with software configurable divider)
  –external clock of 6.67 - 10.67 MHz
  –300ns instruction cycle time (@40 MHz system clock)
- 8 Kbyte on-chip Program ROM for C868-1R and 8 KByte on-chip Program RAM for C868-1S
- In-system programming support for programming the XRAM(C868-1R) or XRAM/ Program RAM(C868-1S)
  –This feature is realized through 4KB Boot ROM
- 256 byte on-chip RAM
- 256 byte on-chip XRAM
- One 8-bit and one 5 bits general purpose push-pull I/O ports
  – Enhanced sink current of 10 mA on Port 1/3 (total sink current of 46 mA @ 100^{o}C)
- Three 16-bit timers/counters
  –Timer 0 / 1
  –Timer/counter 2 (up/down counter feature)
  –Timer 1 or 2 can be used for serial baudrate generator
- Capture/compare unit (CCU6) for PWM signal generation
  –3-channel, 16-bit capture/compare unit
  –1-channel, 16-bit compare unit
- Full duplex serial interface (UART)
- 5 channel 8-bit A/D Converter
- 13 interrupt vectors with 4 priority levels
- Programmable 16-bit Watchdog Timer
- Brown out detection
- Power Saving Modes
  –Slow-down mode
  –Idle mode (can be combined with slow-down mode)
  –Power-down mode with wake up capability through $\overline{\text{INT0}}$ or RxD pins.
- Single power supply of 3.3V, internal voltage regulator for core voltage of 2.5V.
- Individual power-down control for timer/counter 2, capture/compare unit and A/D converter.
- P-DSO-28-1, P-TSSOP-38-1 packages
- Temperature ranges:

  SAF-C868-1RR BA, SAF-C868-1SR BA, SAF-C868-1RG BA, SAF-C868-1SG BA, SAF-C868A-1RR BA, SAF-C868A-1SR BA, SAF-C868A-1RG BA, SAF-C868A-1SG BA, SAF-C868P-1RR BA, SAF-C868-1RG BA T_A= – 40 to 85^{o}C
  SAK-C868-1RR BA, SAK-C868-1SR BA, SAK-C868-1RG BA, SAK-C868-1SG BA,

SAK-C868A-1RR BA, SAK-C868A-1SR BA, SAK-C868A-1RG BA, SAK-C868A-1SG BA, SAK-C868P-1RR BA, SAK-C868-1RG BA T_A = – 40 to 125°C

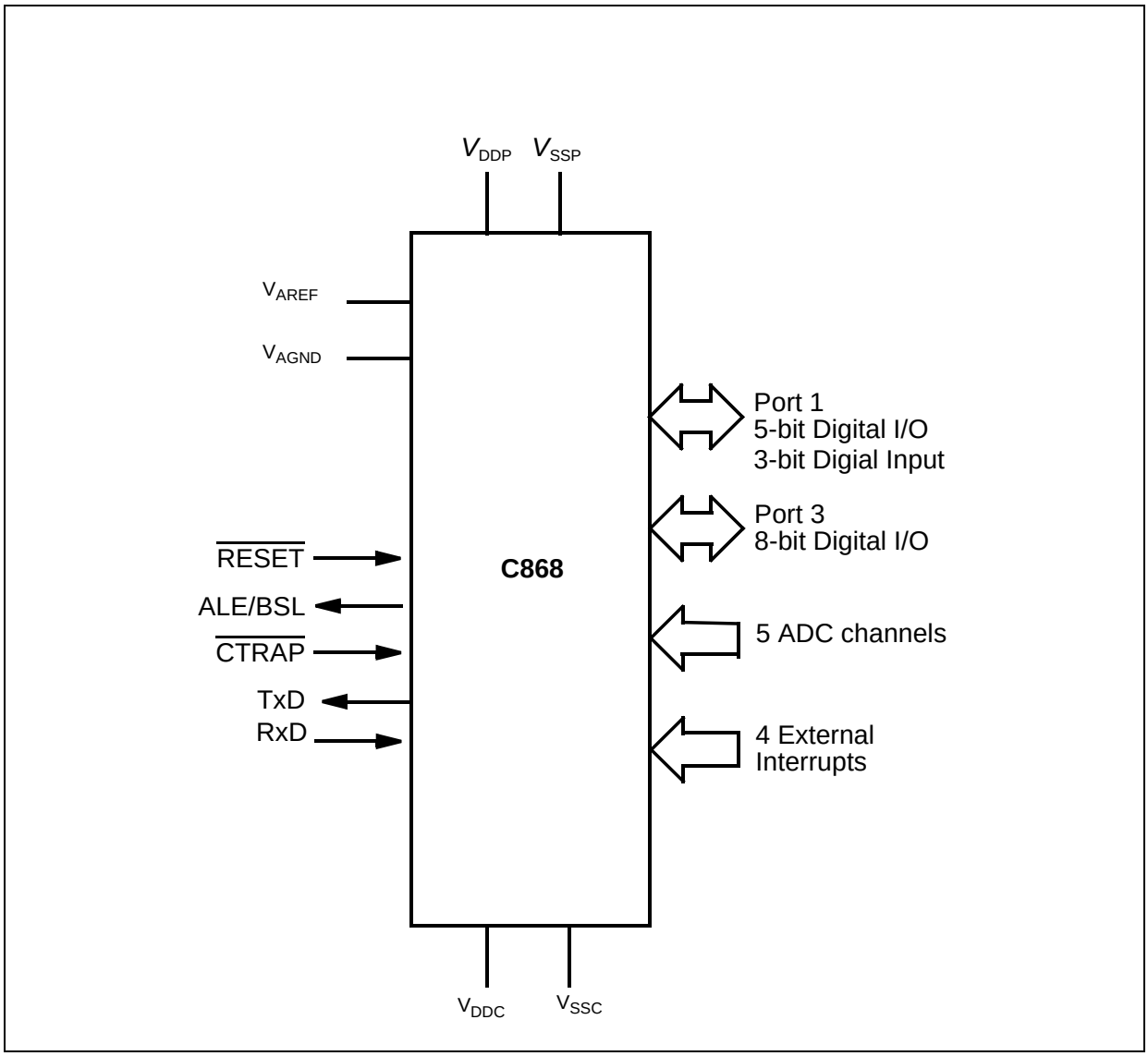


Figure 1-2 Logic Symbol

1.2 Pin Configuration

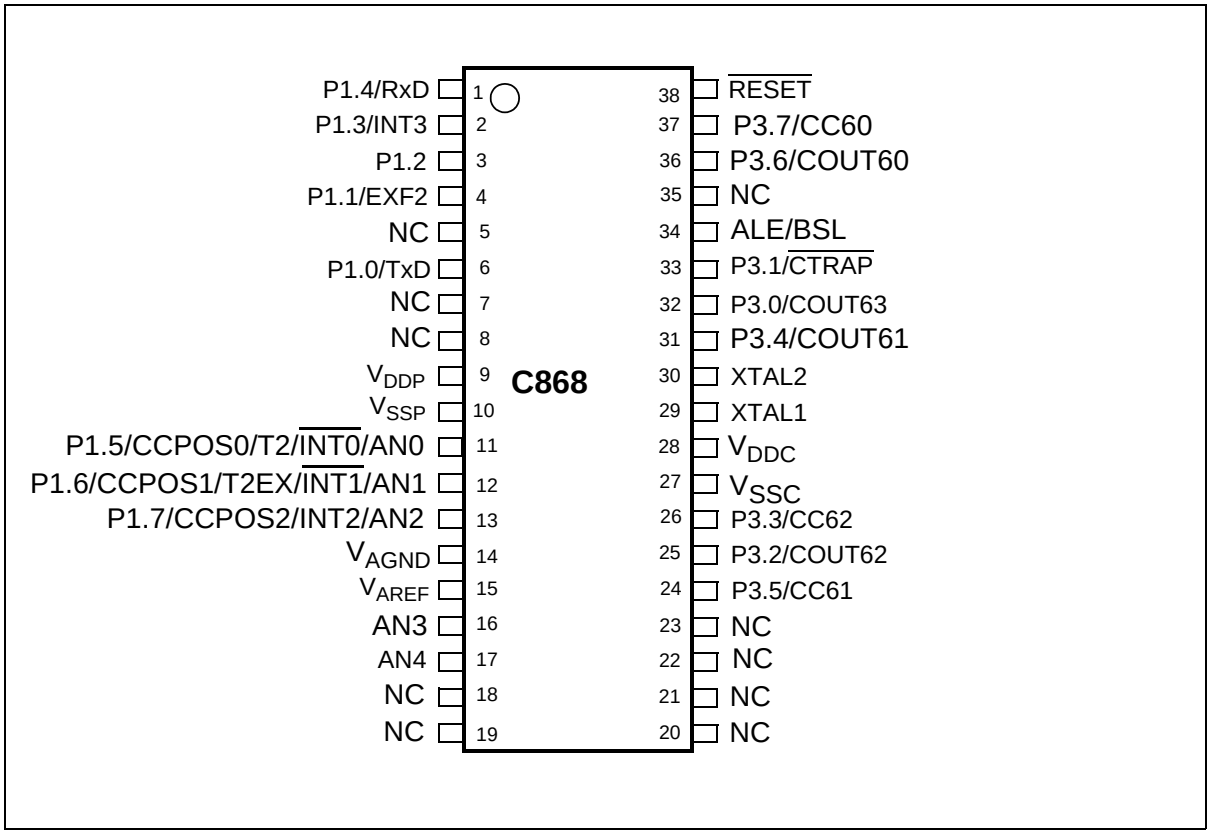


Figure 1-3 C868 Pin Configuration P-TSSOP-38 Package (top view)

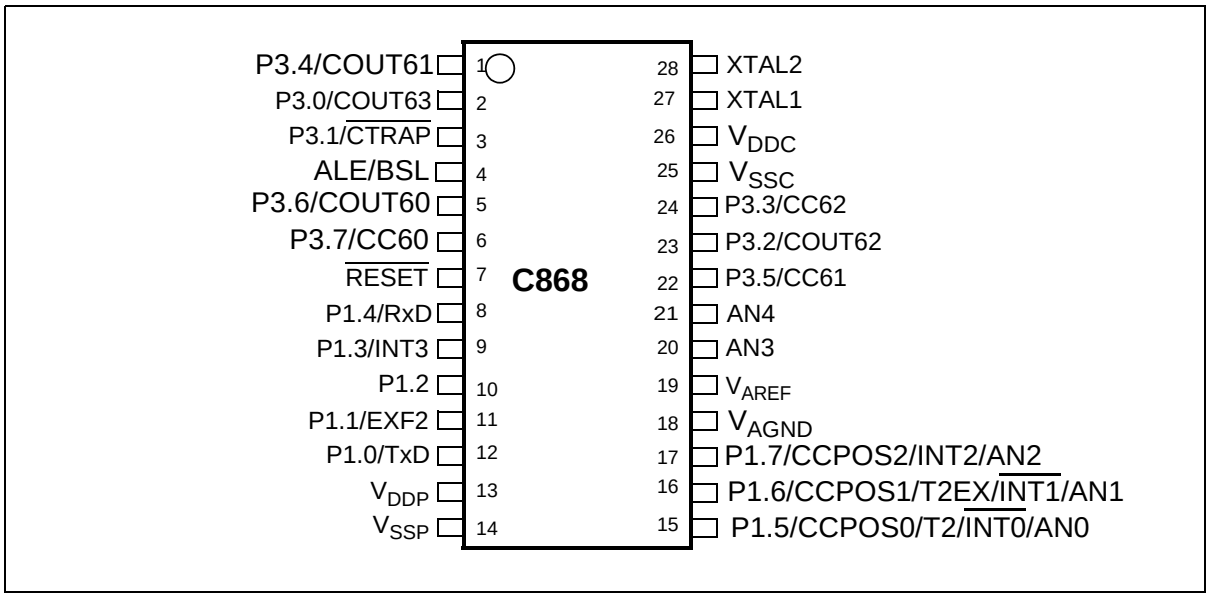


Figure 1-4 C868 Pin Configuration P-DSO-28 Package (top view)

1.3 Pin Definitions and Functions

This section describes all external signals of the C868 with its function.

Table 1 Pin Definitions and Functions

Symbol	Pin Numbers		I/O*)	Function
	P-DSO-28	P-TSSOP-38		
P1.0–P1.4	12-8	6,4-1	I/O	**Port 1** is a combination of 5 bits of push-pull bidirectional I/O ports and 3 bits of input ports. As alternate digital functions, port 1 contains the interrupt 3, timer 2 overflow flag, receive data input and transmit data output of serial interface. The alternate functions are assigned to the pins of port 1 as follows:
P1.5-P1.7	15-17	11-13	I	
	12	6		P1.0/TxD Transmit data of serial interface
	11	4		P1.1/EXF2 Timer 2 overflow flag
	10	3		P1.2
	9	2		P1.3/INT3 Interrupt 3
	8	1		P1.4/RxD Receive data of serial interface, Use as wakeup source from powerdown if bit WS of PMCON0 is set. The input ports are also interrupt ports, input to the timer2, CCU6 modules and ADC:
	15	11	I	P1.5/Input to Counter 2/External Interrupt 0 Input/ Analog Input Channel 0 External interrupt input or Hall input signal, counter 2 input or input channel 0 to the ADC unit. Use as wakeup source from powerdown if bit WS of PMCON0 is cleared.
	16	12	I	P1.6/Timer 2 Trigger/External Interrupt 1 Input/ Analog Input Channel 1 External interrupt input or Hall input signal, input channel 1 to the ADC unit, trigger to Timer 2.
	17	13	I	P1.7/External Interrupt 2 Input/ Analog Input Channel 2 External interrupt input or Hall input signal and input channel 2 to the ADC unit.

Table 1 Pin Definitions and Functions

Symbol	Pin Numbers		I/O*)	Function
	P-DSO-28	P-TSSOP-38		
P3.0–P3.7	2,3,23,24,1,22,5,6	32,33,25,26,31,24,36,37	I/O	**Port 3** is an 8-bit push-pull bidirectional I/O port. This port also serves as alternate functions for the CCU6 functions. The functions are assigned to the pins of port 3 as follows :
	2	32		P3.0/COUT63 16 bit compare channel output
	3	33		P3.1/CTRAP CCU trap input
	23	25		P3.2/COUT62 Output of capture/compare ch 2
	24	26		P3.3/CC62 Input/output of capture/compare ch 2
	1	31		P3.4/COUT61 Output of capture/compare ch 1
	22	24		P3.5/CC61 Input/output of capture/compare ch 1
	5	36		P3.6/COUT60 Output of capture/compare ch 0
	6	37		P3.7/CC60 Input/output of capture/compare ch 0
V_{AREF}	19	15	–	**Reference voltage** for the A/D converter.
V_{AGND}	18	14	–	**Reference ground** for the A/D converter.
AN4	21	17	I	**Analog Input Channel 4** is input channel 4 to the ADC unit.
AN3	20	16	I	**Analog Input Channel 3** is input channel 3 to the ADC unit.
$\overline{\text{RESET}}$	7	38	I	**$\overline{\text{RESET}}$** A low level on this pin for two machine cycle while the oscillator is running resets the device.
ALE/BSL	4	34	I/O	**Address Latch Enable/Bootstrap Mode** A low level on this pin during reset allows the device to go into the bootstrap mode. After reset, this pin will output the address latch enable signal. The ALE can be disabled by bit EALE in SFR SYSCON0.
V_{SSP}	14	10	–	**IO Ground** (0V)
V_{DDP}	13	9	–	**IO Power Supply** (+3.3V)

*)I=Input
O=Output

Table 1 Pin Definitions and Functions

Symbol	Pin Numbers		I/O*)	Function
	P-DSO-28	P-TSSOP-38		
V_{SSC}	25	27	–	**Core Ground** (0V)
V_{DDC}	26	28	O	**Core Internal Reference** (+2.5V) Connect 2*68 - 470nF ceramic capacitor across this pin and core ground.
NC	–	5,7,8,18, 19,20,21, 22,23,35	–	**Not connected**
XTAL1	27	29	I	**XTAL1** Output of the inverting oscillator amplifier.
XTAL2	28	30	O	**XTAL2** Input to the inverting oscillator amplifier and input to the internal clock generation circuits. To drive the device from an external clock source, XTAL2 should be driven, while XTAL1 is left unconnected.

*)I=Input
O=Output

2 Fundamental Structure

The C868 is fully compatible to the architecture of the standard 8051 microcontroller family. While maintaining all architectural and operational characteristics of the 8051, the C868 incorporates a CPU with 8 datapointers, a 8-bit A/D converter, a 16-bit capture/ compare unit, a 16 bit timer 2 that can be used as baudrate generator, an interrupt structure with 2 priority levels, built-in PLL with a fixed factor of 15 and a variable divider, an XRAM data memory as well as some enhancements in the Fail Save Mechanism Unit. **Figure 2-1** shows a block diagram of the C868.

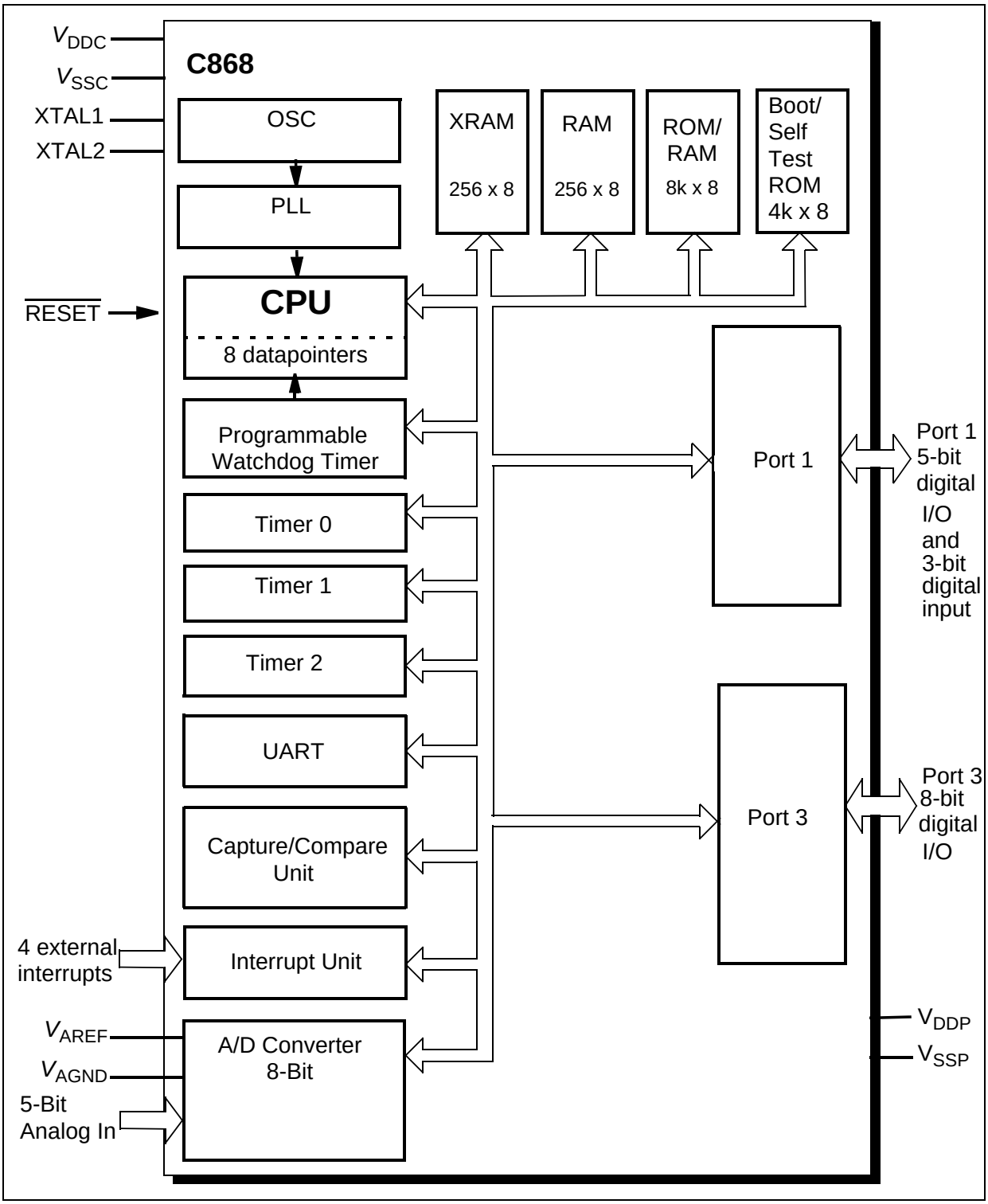


Figure 2-1 Block Diagram of the C868

2.1 CPU

The C868 is efficient both as a controller and as an arithmetic processor. It has extensive facilities for binary and BCD arithmetic and excels in its bit-handling capabilities. Efficient use of program memory results from an instruction set consisting of 44% one-byte, 41% two-byte, and 15% three-byte instructions. With a 10.67 MHz external crystal (giving a 40MHz CPU clock), 58% of the instructions execute in 300 ns.

The CPU (Central Processing Unit) of the C868 consists of the instruction decoder, the arithmetic section and the program control section. Each program instruction is decoded by the instruction decoder. This unit generates the internal signals controlling the functions of the individual units within the CPU. These internal signals have an effect on the source and destination of data transfers and control the ALU processing.

The arithmetic section of the processor performs extensive data manipulation and is comprised of the arithmetic/logic unit (ALU), an A register, B register and PSW register.

The ALU accepts 8-bit data words from one or two sources and generates an 8-bit result under the control of the instruction decoder. The ALU performs the arithmetic operations add, substract, multiply, divide, increment, decrement, BCD-decimal-add-adjust and compare, and the logic operations AND, OR, Exclusive OR, complement and rotate (right, left or swap nibble (left four)). Also included is a Boolean processor performing the bit operations as set, clear, complement, jump-if-set, jump-if-not-set, jump-if-set-and-clear and move to/from carry. Between any addressable bit (or its complement) and the carry flag, the ALU can perform the bit operations of logical AND or logical OR with the result returned to the carry flag.

The program control section controls the sequence in which the instructions stored in program memory are executed. The 16-bit program counter (PC) holds the address of the next instruction to be executed. The conditional branch logic enables internal and external events to the processor to cause a change in the program execution sequence.

Additionally to the CPU functionality of the 8051 standard microcontroller, the C868 contains 8 datapointers. For complex applications with peripherals located in the external data memory space or extended data storage capacity, this turned out to be a "bottle neck" for the 8051’s communication to the external world. Especially programming in high-level languages (PLM51, C51, PASCAL51) requires extended RAM capacity and at the same time a fast access to this additional RAM because of the reduced code efficiency of these languages.

Accumulator

ACC is the symbol for the accumulator register. The mnemonics for accumulator-specific instructions, however, refer to the accumulator simply as A.

Program Status Word

The Program Status Word (PSW) contains several status bits that reflect the current state of the CPU.

PSW
Program Status Word Register **[Reset value: 00_H]**

$D7_H$	$D6_H$	$D5_H$	$D4_H$	$D3_H$	$D2_H$	$D1_H$	$D0_H$
CY	**AC**	**F0**	**RS1**	**RS0**	**OV**	**F1**	**P**
rwh	rwh	rw	rw	rw	rwh	rw	rwh

Field	Bits	Typ	Description
P	0	rwh	**Parity Flag** Set/cleared by hardware after each instruction to indicate an odd/even number of "one" bits in the accumulator, i.e. even parity.
F1	1	rw	**General Purpose Flag**
OV	2	rwh	**Overflow Flag** Used by arithmetic instructions.
RS0 **RS1**	3 4	rw	**Register Bank select control bits** These bits are used to select one of the four register banks. **Table 1 :** **RS1 / RS0 / Function** 0 / 0 / Bank 0 selected, data address 00_H-07_H 0 / 1 / Bank 1 selected, data address 08_H-$0F_H$ 1 / 0 / Bank 2 selected, data address 10_H-17_H 1 / 1 / Bank 3 selected, data address 18_H-$1F_H$
F0	5	rw	**General Purpose Flag**
AC	6	rwh	**Auxiliary Carry Flag** Used by instructions which execute BCD operations.
CY	7	rwh	**Carry Flag** Used by arithmetic instructions.

B Register

The B register is used during multiply and divide and serves as both source and destination. For other instructions it can be treated as another scratch pad register.

Stack Pointer

The stack pointer (SP) register is 8 bits wide. It is incremented before data is stored during PUSH and CALL executions and decremented after data is popped during a POP and RET (RETI) execution, i.e. it always points to the last valid stack byte. While the stack may reside anywhere in the on-chip RAM, the stack pointer is initialized to 07_H after a reset. This causes the stack to begin a location = 08_H above register bank zero. The SP can be read or written under software control.

2.2 CPU Timing

A machine cycle of the C868 consists of 6 states (12 system clock periods). Each state is divided into a phase 1 half and a phase 2 half. Thus, a machine cycle consists of 12 internal clock periods, numbered S1P1 (state 1, phase 1) through S6P2 (state 6, phase 2). Each state lasts two internal clock periods. Typically, arithmetic and logic operations take place during phase 1 and internal register-to-register transfers take place during phase 2.

The diagrams in **Figure 2-2** show the fetch/execute timing related to the internal states and phases. Since these internal clock signals are not user-accessible, the ALE (address latch enable) signal are shown for external reference. ALE is normally activated twice during each machine cycle: once during S1P2 and S2P1, and again during S4P2 and S5P1.

Execution of a one-cycle instruction begins at S1P2, when the op-code is latched into the instruction register. If it is a two-byte instruction, the second reading takes place during S4 of the same machine cycle. If it is a one-byte instruction, there is still a fetch at S4, but the byte read (which would be the next op-code) is ignored (discarded fetch), and the program counter is not incremented. In any case, execution is completed at the end of S6P2.

Figure 2-2 (a) and **(b)** show the timing of a 1-byte, 1-cycle instruction and for a 2-byte, 1-cycle instruction.

Most C868 instructions are executed in one cycle. MUL (multiply) and DIV (divide) are the only instructions that take more than two cycles to complete; they take four cycles. Normally two code bytes are fetched from the program memory during every machine cycle. The only exception to this is when a MOVX instruction is executed. MOVX is a one-byte, 2-cycle instruction that accesses external data memory. During a MOVX, the two fetches in the second cycle are skipped while the external data memory is being addressed and strobed. **Figure 2-2 (c)** and **(d)** show the timing for a normal 1-byte, 2-cycle instruction and for a MOVX instruction.

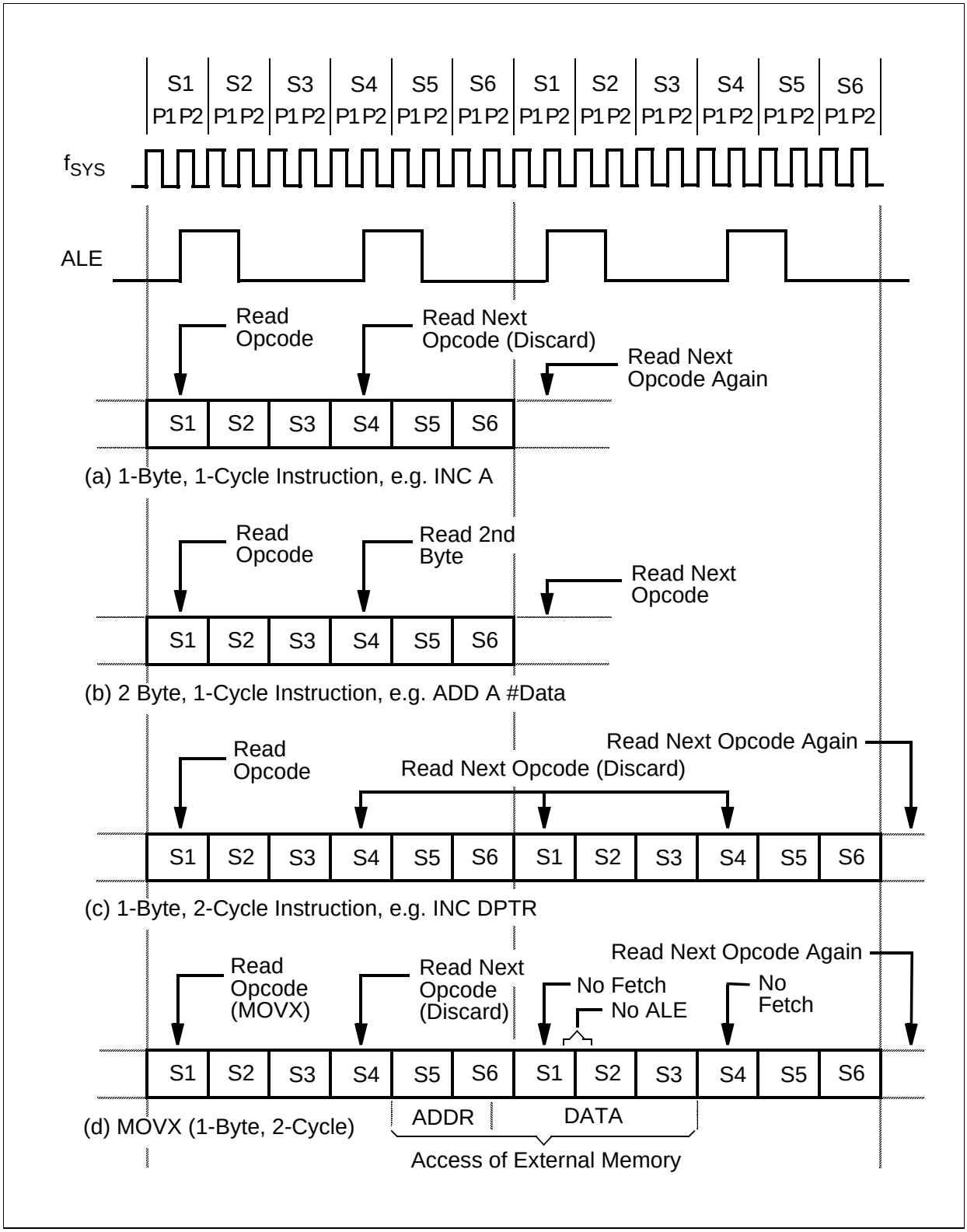


Figure 2-2 Fetch Execute Sequence

3 Memory Organization

The C868 CPU manipulates operands in the following five address spaces:

- up to 8 Kbyte of RAM internal program memory : 8K ROM for C868-1R
  : 8K RAM for C868-1S
- 4 Kbyte of internal Self test and Boot ROM
- 256 bytes of internal data memory
- 256 bytes of internal XRAM data memory
- 128 byte special function register area

Figure 3-1 illustrates the memory address spaces of the C868.

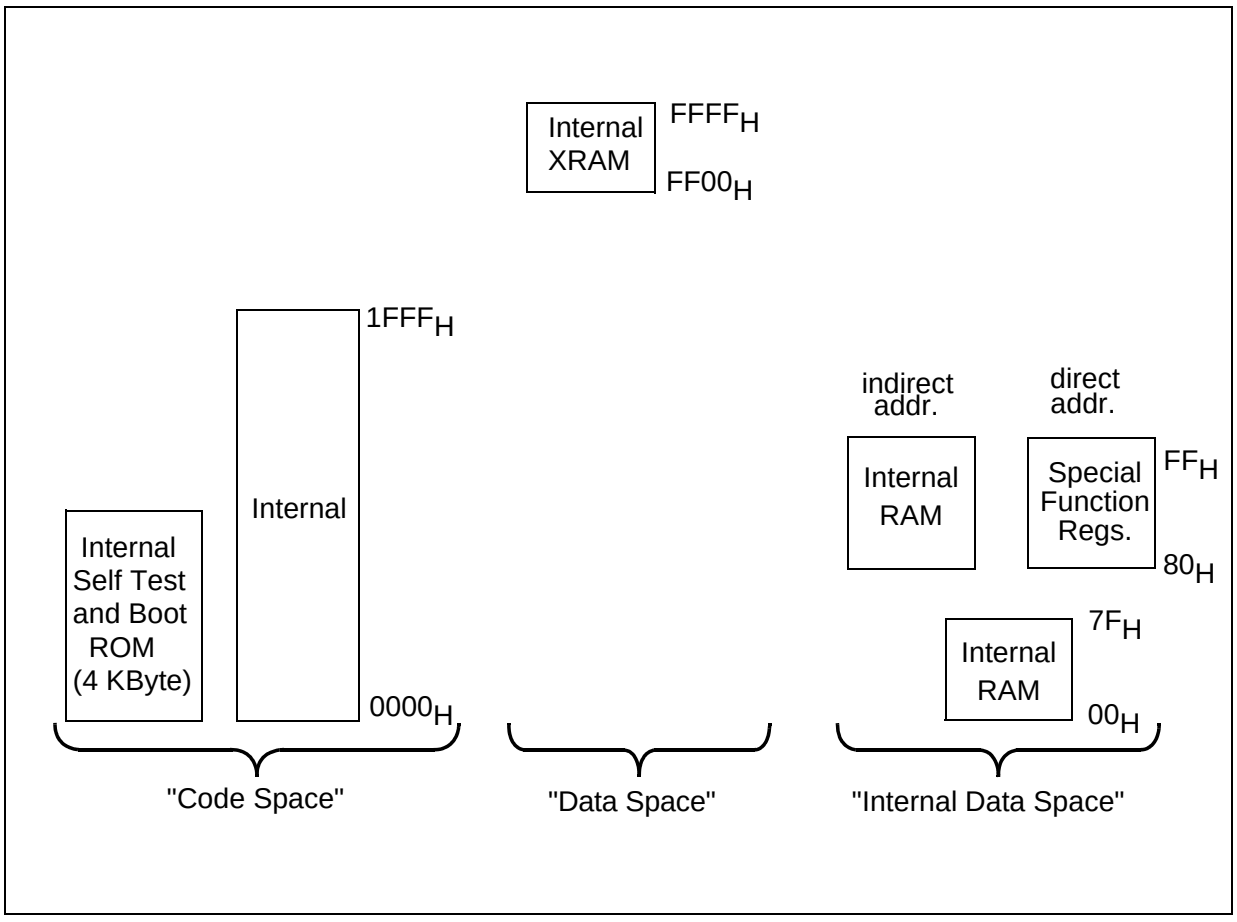


Figure 3-1 C868 Memory Map

The internal Self Test and Boot ROM overlaps the internal program memory in the address range from 0000_H to $0FFF_H$. Depending on the selected operating mode (chipmode), either internal program memory or the internal Self Test and Boot ROM is accessed in this address range.

3.1 Program Memory, “Code Space”

The C868-1S has 8 Kbytes of random access program memory (RAM) and 4 Kbytes of Boot and Self Test ROM. In the normal mode the C868-1S executes program code out of the internal RAM.

The Boot ROM includes a bootstrap loader program for the bootstrap loader of the C868-1S. The software routines of the bootstrap loader program allow the easy and quick programming or loading of the internal program RAM via the serial interface while the MCU is in-circuit.

The C868-1R has 8Kbytes of ROM and 4 Kbytes of Self Test ROM.

The Self Test ROM has a self test program for the self test mode of the C868.

3.2 Data Memory, “Data Space”

The data memory address space consists of an internal and an external(XRAM) memory space. The internal data memory is divided into three physically separate and distinct blocks: the lower 128 bytes of RAM, the upper 128 bytes of RAM, and the 128 byte special function register (SFR) area.

While the upper 128 bytes of data memory and the SFR area share the same address locations, they are accessed through different addressing modes. The lower 128 bytes of data memory can be accessed through direct or register indirect addressing; the upper 128 bytes of RAM can be accessed through register indirect addressing only; the special function registers are accessible through direct addressing. Four 8-register banks, each bank consisting of eight 8-bit multi-purpose registers, occupy locations 00_H through $1F_H$ in the lower RAM area. The next 16 bytes, locations 20_H through $2F_H$, contain 128 directly addressable bit locations. The stack can be located anywhere in the internal data memory address space, and the stack depth can be expanded up to 256 bytes.

The internal XRAM is located in the in the external data memory area and must be accessed by external data memory instructions (MOVX).

3.2.1 General Purpose Registers

The lower 32 locations of the internal RAM are assigned to four banks with eight general purpose registers (GPRs) each. Only one of these banks may be enabled at a time. Two bits in the program status word, RS0 and RS1, select the active register bank. This allows fast context switching, which is useful when entering subroutines or interrupt service routines.

The 8 general purpose registers of the selected register bank may be accessed by register addressing. With register addressing the instruction opcode indicates which register is to be used. For indirect addressing R0 and R1 are used as pointer or index register to address internal or external memory (e.g. MOV @R0).

Reset initializes the stack pointer to location 07_H and increments it once to start from location 08_H which is also the first register (R0) of register bank 1. Thus, if one is going to use more than one register bank, the SP should be initialized to a different location of the RAM which is not used for data storage.

3.3 Program and Data Memory Organisation

The C868 can operate in four different operating modes (chipmodes) with different program and data memory organisations:

- Normal Mode
- Normal XRAM Mode
- Bootstrap Mode
- Bootstrap XRAM Mode

3.3.1 Special function register SYSCON1

There are four control bits located in SFR SYSCON1 which control the code and data memory organisation of the C868. Two of these bits (BSLEN and SWAP) cannot be programmed as normal bits but with a special software unlock sequence. The special software unlock sequence was implemented to prevent unintentional changing of these bits and consists of consecutive followed instructions which have to set set two dedicated enable bits.

SYSCON1
System Control Register 1 **[Reset value: $00XXX0X0_B$]**

7	6	5	4	3	2	1	0
ESWC	**SWC**	**-**	**-**	**-**	**BSLEN**	**-**	**SWAP**
w	w	r	r	r	rw	r	rw

Field	Bits	Typ	Description
SWAP	0	rw	**SWAP Code and Data Memory** SWAP = 0 : Code and data memory are in their standard locations (default) SWAP = 1 : Code and data memory are swapped The modification of this bit is by software only and must be completed by the special software unlock sequence in order to effect the mode change. Otherwise, this bit automatically reverts to its previous value with the third EOI (end of instruction) after this bit is modified. This is to prevent any incorrect status read.

Field	Bits	Typ	Description
BSLEN	2	rw	**Bootstrap Mode Enable** BSLEN = 1 : Bootstrap mode BSLEN = 0 : Normal mode (default) This bit is initialised to the reverse of the value at external pin ALE/BSL latched at the rising edge of RESET. This bit can be set/cleared by software to change between the modes. The modification of this bit by sofware must be completed by the special software unlock sequence in order to effect the mode change. Otherwise, this bit automatically reverts to its previous value with the third EOI (end of instruction) after this bit is modified. This is to prevent any incorrect status read.
SWC	6	w	**Switch Mode** The SWC bit must be set as the second instruction in a special software unlock sequence directly after having set bit ESWC. The new chipmode becomes active after the second EOI (end of instruction) after this event and the SWC bit is also cleared simultaneously. SWC is a write only bit. Reading SWC returns a ’0’.
ESWC	7	w	**Enable Switch Mode** The ESWC bit must be set during the first instruction in the special software unlock sequence. The bit ESWC will be cleared by hardware with the third EOI (end of instruction) after this event. ESWC is a write only bit. Reading ESWC returns a ’0’.
-	[7:2]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

3.3.2 Chip Modes

The various chip modes supported are shown in **Figure 3-2**.

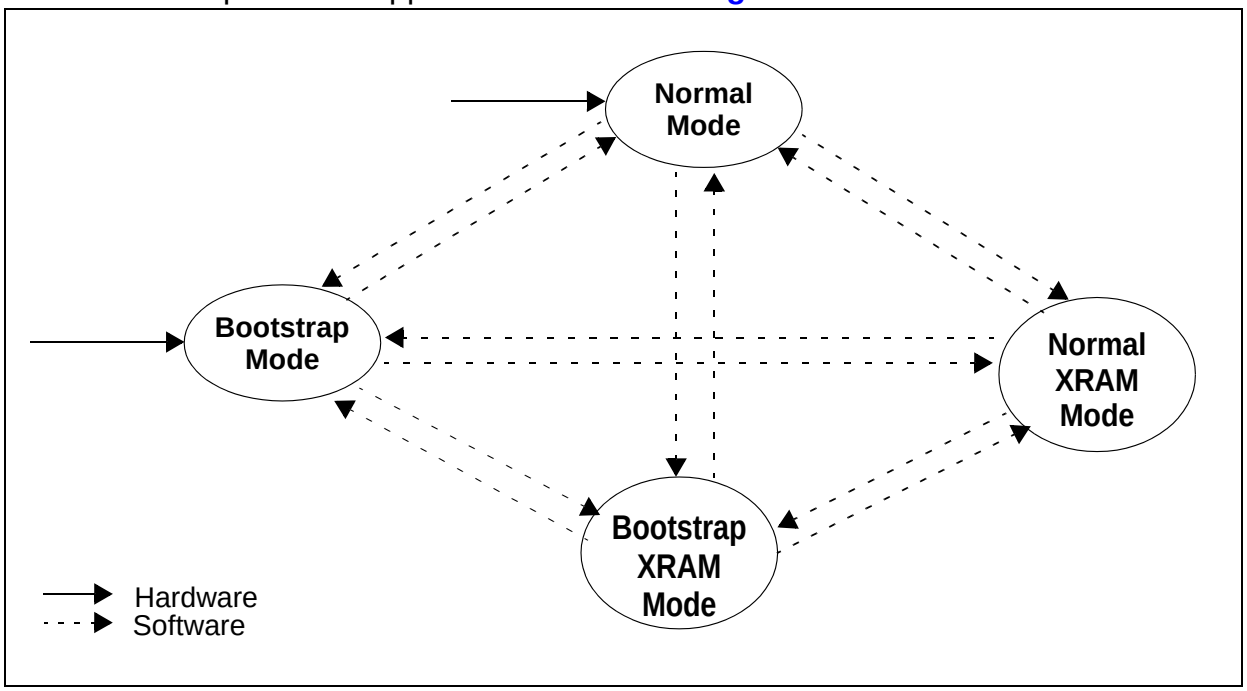


Figure 3-2 Entry and exit of Chip Modes

A valid hardware reset would, of course, override any of the above entry or exit procedures.

Table 3-1 Hardware and Software Selection of Chipmodes

Operating Mode (Chipmode)	Hardware Selection	Software Selection
Normal Mode	ALE/BSL pin = high RESET rising edge	ALE/BSL = don't care; setting bits BSLEN, SWAP = 0,0; execute unlocking sequence
Normal XRAM Mode	Not possible	setting bits BSLEN,SWAP = 0,1; execute unlocking sequence
Bootstrap XRAM Mode	Not possible	setting bits BSLEN,SWAP = 1,1; execute unlocking sequence
Bootstrap Mode	ALE/BSL pin = low RESET rising edge	ALE/BSL = don't care; setting bits BSLEN, SWAP = 1,0; execute unlocking sequence

3.3.2.1 Normal Mode

The normal mode is the standard 8051 compatible operating mode of the C800.

Table 3-2 Normal Memory Configuration for C868

Memory Space	Memory Boundary
Code Space	RAM/ROM: 0000_H to $1FFF_H$
Internal Data Space	XRAM: $FF00_H$ to $FFFF_H$

3.3.2.2 Bootstrap Mode

In the bootstrap mode, code is fetched from the boot-ROM when PC is less than 1000_H. A dedicated 4 Kbyte boot-ROM is implemented to support this function. The actual code inside the boot-ROM could be made up of various components such as programming code for RAM module, download code, initialization routines or diagnostic software.

The bootstrap mode can be entered via one of the possible ways:

- hardware start-up sequence, or
- software entry using special unlock sequence

The exit from the bootstrap mode is possible via one of the possible ways:

- hardware reset, or
- software using special unlock sequence

The memory mapping for this mode is shown in the **Table 3-3**

Table 3-3 Bootstrap Memory Configuration for C868-1R

Memory Space	Memory Boundary
Code Space	Boot ROM: 0000_H to $0FFF_H$
Internal Data Space	XRAM: $FF00_H$ to $FFFF_H$, ROM/RAM: 0000_H to $1FFF_H$

Once in the bootstrap mode, the on-chip XRAM is always enabled irrespective of the XMAP0 bit in SFR SYSCON0. Exiting the bootstrap mode via software, the on-chip XRAM access returns to the state prior to entering this mode, depending on XMAP0.

3.3.2.3 XRAM Modes

In the XRAM modes, code and data memory are swapped and in this case the code can be fetched from the data space. This is useful for running diagnostic software.

The entry and exit into this mode is always through the special software unlock sequence. The XRAM mode could be entered from either the normal mode or the

bootstrap mode. **Table 3-4** and **Table 3-5** show the various memory configurations respectively in an example.

Table 3-4 Normal XRAM Mode

Memory Space	Memory Boundary
Code Space	XRAM: $FF00_H$ to $FFFF_H$
Data Space	ROM/RAM: 0000_H to $1FFF_H$

Table 3-5 Bootstrap XRAM Mode

Memory Space	Memory Boundary
Code Space	Boot ROM: 0000_H to $0FFF_H$ XRAM: $FF00_H$ to $FFFF_H$
Data Space	RAM/ROM: 0000_H to $1FFF_H$

The on-chip XRAM, which is in the upper part of the 64 KB data space, is always enabled in this mode for code access irrespective of the XMAP0 bit. The external data space also becomes code space.

The actual physical sizes of the various memory types as mentioned above are product specific. In the C868, the external accesses are prohibited. For code spaces, appropriate branch instructions must therefore be inserted.

The on-chip data space is accessible, as usual, via MOVX instructions. The on-chip data memory accesses to RAM/ROM are restricted by the physical memory available in the respective product. For the C868-1R, the option to disable the access to the ROM is selectable upon request. This option is reflected in SFR Version bit 7(1 for access disabled).

An exit from the XRAM mode is possible by software only. In this mode the on-chip XRAM is disabled as data space irrespective of XMAP0 bit in SFR SYSCON0. It will remain disabled after exit from XRAM mode unless the XMAP0 had been cleared prior to entering this mode.

3.3.2.4 Software Unlock Sequence

A special software unlock sequence is required to enter or exit the various chip modes supported.

The bits ESWC and SWC in SFR SYSCON1 are implemented in a way to prevent unintentional changing of the bits SWAP and BSLEN. Any change of the bits SWAP or BSLEN not accompanied by the software unlock sequence will have no effect and the above bits will revert back to their previous values two instructions after being changed.

The following programming steps must be executed at the ESWC/SWC unlock sequence:

i) First Instruction:

This instruction should set the ESWC bit and modify of SWAP and/or BSLEN, as necessary.

```
MOV SYSCON1,#10000X0YB ;X is BSLEN, Y is SWAP
```

ii) Second Instruction :

The second instruction must set the SWC bit. If this instruction sequence is followed, then only the mode change in the previous instructions will come into effect. Otherwise the previous mode will be retained and both bits ESWC and SWC are cleared.

The new chip mode becomes effective after the end of the second instruction after the writing of the bit SWC.

```
MOV SYSCON1,#11000X0YB ;X is BSLEN, Y is SWAP
```

iii) Third Instruction:

The instruction following this sequence should be used for initialization of the program counter to the 16 bit start-address of the new code memory resource, e.g. with :

```
LJMP   0XXXXH ;XXXX is the 16-bit hexadecimal address in new code memory
```

If both SWAP and BSLEN bits are set in the first instruction, both modes will still be entered. It is, in any case, the responsibility of the user to provide the appropriate relocation address depending on the mode prior to the execution of this sequence. The special software unlock instruction sequence cannot be interrupted by an interrupt request. Any read or write operation to SFR SYSCON1 will block the interrupt generation for the first cycle of the directly following instruction. Therefore, the response time of an interrupt request may be additionally delayed.

3.4 Special Function Registers

All registers, except the program counter and the four general purpose register banks, reside in the special function register area. The special function register area consists of two portions: the standard special function register area and the mapped special function register area. For accessing the mapped special function area, bit RMAP in special function register SYSCON0 must be set. All other special function registers are located in the standard special function register area which is accessed when RMAP is cleared (“0“).

SYSCON0
System Control Register 0 **[Reset value: XX10XXX1$_B$]**

7	6	5	4	3	2	1	0
-	-	EALE	RMAP	-	-	-	XMAP0
r	r	rw	rw	r	r	r	rw

The functions of the shaded bits are not described here

Field	Bits	Typ	Description
RMAP	4	rw	**Special Function Register Map Control** RMAP = 0 : The access to the non-mapped (standard) special function register area is enabled. RMAP = 1 : The access to the mapped special function register area is enabled.
-	[7:2]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

As long as bit RMAP is set, the mapped special function register area can be accessed. This bit is not cleared automatically by hardware. Thus, when non-mapped/mapped registers are to be accessed, the bit RMAP must be cleared/set respectively by software.

The special function registers (SFR) include pointers and registers that provide an interface between the CPU and the other on-chip peripherals. All available SFRs whose address bits 0-2 are 0 (e.g. 80_H, 88_H, 90_H, ..., $F0_H$, $F8_H$) are bit- addressable. Totally there are directly addressable bits within the SFR area.

All SFRs are listed in **Table 3-6** and **Table 3-7**.

In **Table 3-6** they are organized in groups which refer to the functional blocks of the C868-1R, C868-1S. **Table 3-7** illustrates the contents (bits) of the SFRs

Table 3-6 Special Function Registers - Functional Blocks

Block	Symbol	Name	Add-ress	Contents after Reset
C800 core	ACC	Accumulator	**$E0_H$**[1]	**00_H**
	B	B-Register	**$F0_H$**[1]	**00_H**
	DPH	Data Pointer, High Byte	83_H	00_H
	DPL	Data Pointer, Low Byte	82_H	00_H
	DPSEL	Data Pointer Select Register	84_H	00_H
	PSW	Program Status Word Register	**$D0_H$**[1]	**00_H**
	SP	Stack Pointer	81_H	07_H
	SCON	Serial Channel Control Register	**98_H**[1]	**00_H**
	SBUF	Serial Data Buffer	99_H	00_H
	IEN0	Interrupt Enable Register 0	**$A8_H$**[1]	**$0X000000_B$**[2]
	IEN1	Interrupt Enable Register 1	$A9_H$	$XXXXX000_B$[2]
	IEN2	Interrupt Enable Register 2	AA_H	$XX0000XX_B$[2]
	IP0	Interrupt Priority Register 0	**$B8_H$**[1]	**$XX000000_B$**[2]
	IP1	interrupt Priority Register 1	AC_H	$XX000000_B$[2]
	TCON	Timer 0/1 Control Register	**88_H**[1]	**00_H**
	TMOD	Timer Mode Register	89_H	00_H
	TL0	Timer 0, Low Byte	$8A_H$	00_H
	TL1	Timer 1, Low Byte	$8B_H$	00_H
	TH0	Timer 0, High Byte	$8C_H$	00_H
	TH1	Timer 1, High Byte	$8D_H$	00_H
	PCON	Power Control Register	87_H	$0XXX0000_B$[2]
Sys-tem	PMCON0	Wake-up Control Register	$8E_H$	$XXX00000_B$[2]
	CMCON	Clock Control Register	$8F_H$	10011111_B
	EXICON	External Interrupt Control Register	91_H	$XXXXXX00_B$[2]
	IRCON0	External Interrupt Request Register	92_H	$XXXXXX00_B$[2]
	IRCON1	Peripheral Interrupt Request Register	93_H	$XX0000X0_B$[2]
	PMCON1	Peripheral Management Ctrl Register	**$E8_H$**[1]	**$XXXX000_B$**[2]
	PMCON2	Peripheral Management Status Register	**$F8_H$**[1]	**$XXXX000_B$**[2]
	SCUWDT	SCU/Watchdog Control Register	**$C0_H$**[1]	**$X0X00000_B$**[2]
	VERSION	ROM Version Register	$F9_H$	00_H
	SYSCON0	System Control Register 0	AD_H	$XX10XXX1_B$[2]
	SYSCON1	System Control Register 1	AF_H	$00XXX0X0_B$[2]

1) Bit-addressable special function registers
2) “X“ means that the value is undefined and the location is reserved
3) Register is mapped by bit RMAP in SYSCON0.4=1
4) Register is mapped by bit RMAP in SYSCON0.4=0

Table 3-6 Special Function Registers - Functional Blocks (cont'd)

Block	Symbol	Name	Add-ress	Contents after Reset
A/D-Con-verter	ADCON0 ADCON1 ADDATH	A/D Converter Control Register 0 A/D Converter Control Register 1 A/D Converter Data Register	**$D8_H$** [1] $D9_H$ DB_H	**00000000_B** [2] $XX000000_B$ [2] 00_H
Ports	P1 [4] P1DIR [3] P3 [4] P3DIR [3] P3ALT P1ALT	Port 1 Register Port 1 Direction Register Port 3 Register Port 3 Direction Register Port 3 Alternate Function Register Port 1 Alternate Function Register	**90_H** [1] **90_H** [1] **$B0_H$** [1] **$B0_H$** [1] $B1_H$ $B4_H$	**11111111_B** **11111111_B** **FF_H** **FF_H** 00_H $XXX00X00_B$ [2]
Watch dog	WDTCON WDTREL WDTL WDTH	Watchdog Timer Control Register Watchdog Timer Reload Register Watchdog Timer, Low Byte Watchdog Timer, High Byte	$A2_H$ $A3_H$ $B2_H$ $B3_H$	$XXXXXX00_B$ [2] 00_H 00_H 00_H
Timer 2	T2CON T2MOD RC2H RC2L T2H T2L	Timer 2 Control Register Timer 2 Mode Register Timer 2 Reload/Capture, High Byte Timer 2 Reload/Capture, Low Byte Timer 2, High Byte Timer 2, Low Byte	**$C8_H$** [1] $C9_H$ CB_H CA_H CD_H CC_H	**00_H** $XXXXXXX0_B$ [2] 00_H 00_H 00_H 00_H

1) Bit-addressable special function registers
2) "X" means that the value is undefined and the location is reserved
3) Register is mapped by bit RMAP in SYSCON0.4=1
4) Register is mapped by bit RMAP in SYSCON0.4=0

Table 3-6 Special Function Registers - Functional Blocks (cont'd)

Block	Symbol	Name	Add-ress	Contents after Reset
Cap-ture/ Com-pare Unit	T12L	Timer T12 Counter Register, Low Byte	EC_H	00_H
	T12H	Timer T12 Counter Register, High Byte	ED_H	00_H
	T13L	Timer T13 Counter Register, Low Byte	EE_H	00_H
	T13H	Timer T13 Counter Register, High Byte	EF_H	00_H
	T12PRL	Timer T12 Period Register, Low Byte	DE_H	00_H
	T12PRH	Timer T12 Period Register, High Byte	DF_H	00_H
	T13PRL	Timer T13 Period Register, Low Byte	$D2_H$	00_H
	T13PRH	Timer T13 Period Register, High Byte	$D3_H$	00_H
	CC60RL	Capture/Compare Ch 0 Reg, Low Byte	$C2_H$	00_H
	CC60RH	Capture/Compare Ch 0 Reg, High Byte	$C3_H$	00_H
	CC61RL	Capture/Compare Ch 1 Reg, Low Byte	$C4_H$	00_H
	CC61RH	Capture/Compare Ch 1 Reg, High Byte	$C5_H$	00_H
	CC62RL	Capture/Compare Ch 2 Reg, Low Byte	$C6_H$	00_H
	CC62RH	Capture/Compare Ch 2 Reg, High Byte	$C7_H$	00_H
	CC63RL	T13 Compare Register, Low Byte	$D4_H$	00_H
	CC63RH	T13 Compare Register, High Byte	$D5_H$	00_H
	T12DTCL	Timer T12 Dead Time Ctrl, Low Byte	$E6_H$	00_H
	T12DTCH	Timer T12 Dead Time Ctrl, High Byte	$E7_H$	00_H
	CMPSTATL	Compare Timer Status, Low Byte	$F4_H$	00_H
	CMPSTATH	Compare Timer Status, High Byte	$F5_H$	00_H
	CMPMODIFL	Compare Timer Modification, Low Byte	EA_H	00_H
	CMPMODIFH	Compare Timer Modification, High Byte	EB_H	00_H
	TCTR0L	Timer Control Register 0, Low Byte	$E2_H$	00_H
	TCTR0H	Timer Control Register 0, High Byte	$E3_H$	00_H
	TCTR2L[3]	Timer Control Register 2, Low Byte	$F2_H$	00_H
	TCTR4L[4]	Timer Control Register 4, Low Byte	$F2_H$	0_H
	TCTR4H[4]	Timer Control Register 4, High Byte	$F3_H$	00_H
	ISL	Cap/Com Interrupt Register, Low Byte	$E4_H$	00_H
	ISH	Cap/Com Interrupt Register, High Byte	$E5_H$	00_H
	ISSL[3]	Cap/Com Int Status Set Reg, Low Byte	BC_H	00_H
	ISSH[3]	Cap/Com Int Status Set Reg, High Byte	BD_H	00_H
	ISRL[4]	Cap/Com Int Status Reset Reg, Low Byte	BC_H	00_H
	ISRH[4]	Cap/Com Int Status Reset Reg,High Byte	BD_H	00_H
	PISELH	Port Input Selector Register, High Byte	BB_H	00_H

1) Bit-addressable special function registers
2) "X" means that the value is undefined and the location is reserved
3) Register is mapped by bit RMAP in SYSCON0.4=1
4) Register is mapped by bit RMAP in SYSCON0.4=0

Table 3-6 Special Function Registers - Functional Blocks (cont'd)

Block	Symbol	Name	Add-ress	Contents after Reset
Cap-ture/ Com-pare Unit	INPL[3]	Cap/Com Int Node Ptr Reg, Low Byte	BE_H	40_H
	INPH[3]	Cap/Com Int Node Ptr Reg, High Byte	BF_H	39_H
	IENL[4]	Cap/Com Interrupt Register, Low Byte	BE_H	00_H
	IENH[4]	Cap/Com Interrupt Register, High Byte	BF_H	00_H
	CC60SRL	Cap/Com Channel 0 Shadow, Low Byte	FA_H	00_H
	CC60SRH	Cap/Com Channel 0 Shadow, High Byte	FB_H	00_H
	CC61SRL	Cap/Com Channel 1 Shadow, Low Byte	FC_H	00_H
	CC61SRH	Cap/Com Channel 1 Shadow, High Byte	FD_H	00_H
	CC62SRL	Cap/Com Channel 2 Shadow, Low Byte	FE_H	00_H
	CC62SRH	Cap/Com Channel 2 Shadow, High Byte	FF_H	00_H
	CC63SRL	T13 Compare Shadow Reg, Low Byte	$B6_H$	00_H
	CC63SRH	T13 Compare Shadow Reg, High Byte	$B7_H$	00_H
	MODCTRL[3]	Modulation Control Register, Low Byte	$D6_H$	00_H
	MODCTRH[3]	Modulation Control Register, High Byte	$D7_H$	00_H
	TRPCTRL	Trap Control Register, Low Byte	CE_H	00_H
	TRPCTRH	Trap Control Register, High Byte	CF_H	00_H
	PSLRL	Passive State Level Register, Low Byte	$A6_H$	00_H
	MCMOUTL[3]	MCM Output Register, Low Byte	DC_H	00_H
	MCMOUTH[3]	MCM Output Register, High Byte	DD_H	00_H
	MCMOUTSL[4]	MCM Output Shadow Register, Low Byte	DC_H	00_H
	MCMOUTSH[4]	MCM Output Shadow Register,High Byte	DD_H	00_H
	MCMCTRLL[4]	MCM Control Register, Low Byte	$D6_H$	00_H
	T12MSELL	T12 Cap/Com Mode Sel Reg, Low Byte	$F6_H$	00_H
	T12MSELH	T12 Cap/Com Mode Sel Reg, High Byte	$F7_H$	00_H

1) Bit-addressable special function registers
2) "X" means that the value is undefined and the location is reserved
3) Register is mapped by bit RMAP in SYSCON0.4=1
4) Register is mapped by bit RMAP in SYSCON0.4=0

Table 3-7 Contents of the SFRs, SFRs in numeric order of their addresses

Addr	Reg-ister	Content after Reset[1)]	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
81_H	SP	07_H	.7	.6	.5	.4	.3	.2	.1	.0
82_H	DPL	00_H	.7	.6	.5	.4	.3	.2	.1	.0
83_H	DPH	00_H	.7	.6	.5	.4	.3	.2	.1	.0
84_H	DPSEL	00_H	–	–	–	–	–	D2	D1	D0
87_H	PCON	$0XX0\ 0000_B$	SMOD	–	–	SD	GF1	GF0	PDE	IDLE
88_H	TCON	00_H	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
89_H	TMOD	00_H	GATE 1	C/NT1	M1(1)	M0(1)	GATE 0	C/NT0	M1(0)	M0(0)
$8A_H$	TL0	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$8B_H$	TL1	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$8C_H$	TH0	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$8D_H$	TH1	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$8E_H$	PMCON0	$XXX0\ 0000_B$	–	–	–	EBO	BO	SDSTAT	WS	EPWD
$8F_H$	CMCON	$1001\ 1111_B$	KDIV2	KDIV1	KDIV0	REL4	REL3	REL2	REL1	REL0
90_H[2)]	P1	$1111\ 1111_B$	–	–	–	.4	.3	.2	.1	.0
90_H[3)]	P1DIR	$1111\ 1111_B$	–	–	–	.4	.3	.2	.1	.0
91_H	EXICON	$XXXX\ XX00_B$	–	–	–	–	–	–	ESEL3	ESEL2
92_H	IRCON0	$XXXX\ XX00_B$	–	–	–	–	–	–	EXINT3	EXINT2
93_H	IRCON1	$XX000\ 0X0_B$	–	–	INP3	INP2	INP1	INP0	–	IADC

1) X means that the value is undefined and the location is reserved
2) This register is mapped with RMAP (SYSCON0.4)=0
3) This register is mapped with RMAP (SYSCON0.4)=1
Shaded registers are bit-addressable special function registers

Table 3-7 Contents of the SFRs, SFRs in numeric order of their addresses

Addr	Reg-ister	Content after Reset[1)]	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
98_H	SCON	00_H	SM0	SM1	SM2	REN	TB8	RB8	TI	RI
99_H	SBUF	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$A2_H$	WDTCON	$XXXX XX00_B$	–	–	–	–	–	–	–	WDTIN
$A3_H$	WDTREL	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$A6_H$	PSLRL	00_H	PSL63	–	PSL5	PSL4	PSL3	PSL2	PSL1	PSL0
$A8_H$	IEN0	$0X00 0000_B$	EA	–	ET2	ES	ET1	EX1	ET0	EX0
$A9_H$	IEN1	$XXXX X000_B$	–	–	–	–	–	EX3	EX2	EADC
AA_H	IEN2	$XX00 00XX_B$	–	–	EINP3	EINP2	EINP1	EINP0	–	–
AC_H	IP1	$XX00 0000_B$	–	–	.5	.4	.3	.2	.1	.0
AD_H	SYSCON0	$XX10 XXX1_B$	–	–	EALE	RMAP	–	–	–	XMAP0
AF_H	SYSCON1	$00XX X0X0_B$	ESWC	SWC	_	_	_	BSLEN	_	SWAP
$B0_H$[2)]	P3	FF_H	.7	.6	.5	.4	.3	.2	.1	.0
$B0_H$[3)]	P3DIR	FF_H	.7	.6	.5	.4	.3	.2	.1	.0
$B1_H$	P3ALT	00_H	CC60	COUT60	CC61	COUT61	CC62	COUT62	CTRAP	COUT63
$B2_H$	WDTL	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$B3_H$	WDTH	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$B4_H$	P1ALT	$XXX0 0X00_B$	_	_	_	RxD	INT3	_	EXF2_ALT	TxD
$B6_H$	CC63SRL	00_H	.7	.6	.5	.4	.3	.2	.1	.0

1) X means that the value is undefined and the location is reserved
2) This register is mapped with RMAP (SYSCON0.4)=0
3) This register is mapped with RMAP (SYSCON0.4)=1
Shaded registers are bit-addressable special function registers

Table 3-7 Contents of the SFRs, SFRs in numeric order of their addresses

Addr	Reg-ister	Content after Reset[1)]	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$B7_H$	CC63 SRH	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$B8_H$	IP0	$XX00\ 0000_B$	–	–	.5	.4	.3	.2	.1	.0
BB_H	PISEL H	00_H	0	0	ISPOS 2.1	ISPOS 2.0	ISPOS 1.1	ISPOS 1.0	ISPOS 0.1	ISPOS 0.0
BC_H[3)]	ISSL	00_H	ST12P M	ST12O M	SCC62 F	SCC62 R	SCC61 F	SCC61 R	SCC60 F	SCC60 R
BC_H[2)]	ISRL	00_H	RT12P M	RT12O M	RCC6 2F	RCC6 2R	RCC6 1F	RCC6 1R	RCC6 0F	RCC6 0R
BD_H[3)]	ISSH	00_H	–	SIDLE	SWHE	SCHE	–	STRP F	ST13P M	ST13C M
BD_H[2)]	ISRH	00_H	–	RIDLE	RWHE	RCHE	–	RTRP F	RT13P M	RT13C M
BE_H[2)]	IENL	00_H	ENT12 PM	ENT12 OM	ENCC 62F	ENCC 62R	ENCC 61F	ENCC 61R	ENCC 60F	ENCC 60R
BE_H[3)]	INPL	00_H	INPCH E.1	INPCH E.0	INPCC 62.1	INPCC 62.0	INPCC 61.1	INPCC 61.0	INPCC 60.1	INPCC 60.0
BF_H[2)]	IENH	00_H	–	ENIDL E	ENWH E	ENCH E	–	ENTR PF	ENT13 PM	ENT13 CM
BF_H[3)]	INPH	00_H	–	–	INPT1 3.1	INPT1 3.0	INPT1 2.1	INPT1 2.0	INPER R.1	INPER R.0
$C0_H$	SCUW DT	00_H	–	PLLR	–	WDTR	WDTE OI	WDTD IS	WDTR S	WDTR E
$C2_H$	CC60 RL	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$C3_H$	CC60 RH	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$C4_H$	CC61 RL	00_H	.7	.6	.5	.4	.3	.2	.1	.0

1) X means that the value is undefined and the location is reserved
2) This register is mapped with RMAP (SYSCON0.4)=0
3) This register is mapped with RMAP (SYSCON0.4)=1
Shaded registers are bit-addressable special function registers

Table 3-7 Contents of the SFRs, SFRs in numeric order of their addresses

Addr	Reg-ister	Content after Reset[1)]	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$C5_H$	CC61RH	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$C6_H$	CC62RL	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$C7_H$	CC62RH	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$C8_H$	T2CON	00_H	TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/$\overline{T2}$	CP/$\overline{RL2}$
$C9_H$	T2MOD	XXXXXXX0$_B$	–	–	–	–	–	–	–	DCEN
CA_H	RC2L	00_H	.7	.6	.5	.4	.3	.2	.1	.0
CB_H	RC2H	00_H	.7	.6	.5	.4	.3	.2	.1	.0
CC_H	TL2	00_H	.7	.6	.5	.4	.3	.2	.1	.0
CD_H	TH2	00_H	.7	.6	.5	.4	.3	.2	.1	.0
CE_H	TRPCTRL	00_H	–	–	–	–	–	TRPM2	TRPM1	TRPM0
CF_H	TRPCTRH	00_H	TRPPEN	TRPEN13	TRPEN5	TRPEN4	TRPEN3	TRPEN2	TRPEN1	TRPEN0
$D0_H$	PSW	00_H	CY	AC	F0	RS1	RS0	OV	F1	P
$D2_H$	T13PRL	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$D3_H$	T13PRH	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$D4_H$	CC63RL	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$D5_H$	CC63RH	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$D6_H$[2)]	MCMCTRLL	00_H	–	–	SWSYN1	SWSYN0	–	SWSEL2	SWSEL1	SWSEL0

1) X means that the value is undefined and the location is reserved
2) This register is mapped with RMAP (SYSCON0.4)=0
3) This register is mapped with RMAP (SYSCON0.4)=1
Shaded registers are bit-addressable special function registers

Table 3-7 Contents of the SFRs, SFRs in numeric order of their addresses

Addr	Reg-ister	Content after Reset[1]	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$D6_H$[3]	MODCTRL	00_H	MCMEN	–	T12MODEN5	T12MODEN4	T12MODEN3	T12MODEN2	T12MODEN1	T12MODEN0
$D7_H$[3]	MODCTRH	00_H	ECT13O	–	T13MODEN5	T13MODEN4	T13MODEN3	T13MODEN2	T13MODEN1	T13MODEN0
$D8_H$	ADCON0	$0000\ 0000_B$	ADST	ADBSY	ADM1	ADM0	CCU-ADEX	ADCH2	ADCH1	ADCH0
$D9_H$	ADCON1	$XX00\ 0000_B$	–	–	ADSTC2	ADSTC1	ADSTC0	ADCTC2	ADCTC1	ADCTC0
DB_H	ADDATH	00_H	.7	.6	.5	.4	.3	.2	.1	.0
DC_H[3]	MCMOUTL	00_H	–	R	MCMP5	MCMP4	MCMP3	MCMP2	MCMP1	MCMP0
DC_H[2]	MCMOUTSL	00_H	STRMCM	–	MCMPS5	MCMPS4	MCMPS3	MCMPS2	MCMPS1	MCMPS0
DD_H[3]	MCMOUTH	00_H	–	–	CURH2	CURH1	CURH0	EXPH2	EXPH1	EXPH0
DD_H[2]	MCMOUTSH	00_H	STRHP	–	CURHS2	CURHS1	CURHS0	EXPHS2	EXPHS1	EXPHS0
DE_H	T12PRL	00_H	.7	.6	.5	.4	.3	.2	.1	.0
DF_H	T12PRH	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$E0_H$	ACC	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$E2_H$	TCTR0L	00_H	CTM	CDIR	STE12	T12R	T12PRE	T12CLK2	T12CLK1	T12CLK0
$E3_H$	TCTR0H	10_H	–	–	STE13	T13R	T13PRE	T13CLK2	T13CLK1	T13CLK0
$E4_H$	ISL	00_H	T12PM	T12OM	ICC62F	ICC62R	ICC61F	ICC61R	ICC60F	ICC60R

1) X means that the value is undefined and the location is reserved
2) This register is mapped with RMAP (SYSCON0.4)=0
3) This register is mapped with RMAP (SYSCON0.4)=1
Shaded registers are bit-addressable special function registers

Table 3-7 Contents of the SFRs, SFRs in numeric order of their addresses

Addr	Reg-ister	Content after Reset[1)]	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$E5_H$	ISH	00_H	–	CCU_IDLE	WHE	CHE	TRPS	TRPF	T13PM	T13CM
$E6_H$	T12DTCL	00_H		–	DTM5	DTM4	DTM3	DTM2	DTM1	DTM0
$E7_H$	T12DTCH	00_H	–	DTR2	DTR1	DTR0	–	DTE2	DTE1	DTE0
$E8_H$	PMCON1	$XXXX\ X000_B$	–	–	–	–	–	CCUDIS	T2DIS	ADCDIS
EA_H	CMPMODIFL	00_H	–	MCC63S	–	–	–	MCC62S	MCC61S	MCC60S
EB_H	CMPMODIFH	00_H	–	MCC63R	–	–	–	MCC62R	MCC61R	MCC60R
EC_H	T12L	00_H	.7	.6	.5	.4	.3	.2	.1	.0
ED_H	T12H	00_H	.7	.6	.5	.4	.3	.2	.1	.0
EE_H	T13L	00_H	.7	.6	.5	.4	.3	.2	.1	.0
EF_H	T13H	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$F0_H$	B	00_H	.7	.6	.5	.4	.3	.2	.1	.0
$F2_H$[2)]	TCTR4L	00_H	T12STD	T12STR	–	–	DTRES	T12RES	T12RS	T12RR
$F2_H$[3)]	TCTR2L	00_H	–	T13TED1	T13TED0	T13TEC2	T13TEC1	T13TEC0	T13SSC	T12SSC
$F3_H$[2)]	TCTR4H	00_H	T13STD	T13STR	–	–	–	T13RES	T13RS	T13RR
$F4_H$	CMPSTATL	00_H	–	CC63ST	–	–	–	CC62ST	CC61ST	CC60ST
$F5_H$	CMPSTATH	00_H	T13IM	COUT63PS	COUT62PS	CC62PS	COUT61PS	CC61PS	COUT60PS	CC60PS
$F6_H$	T12MSELL	00_H	MSEL613	MSEL612	MSEL611	MSEL610	MSEL603	MSEL602	MSEL601	MSEL600

1) X means that the value is undefined and the location is reserved
2) This register is mapped with RMAP (SYSCON0.4)=0
3) This register is mapped with RMAP (SYSCON0.4)=1
Shaded registers are bit-addressable special function registers

Table 3-7 Contents of the SFRs, SFRs in numeric order of their addresses

Addr	Reg-ister	Content after Reset[1)]	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$F7_H$	T12M SELH	00_H	–	–	–	–	MSEL 623	MSEL 622	MSEL 621	MSEL 620
$F8_H$	PMCO N2	$XXXX X000_B$	–	–	–	–	–	CCUS T	T2ST	ADCS T
$F9_H$	VERSI ON	00_H	PROT	VER6	VER5	VER4	VER3	VER2	VER1	VER0
FA_H	CC60 RL	00_H	.7	.6	.5	.4	.3	.2	.1	.0
FB_H	CC60 RH	00_H	.7	.6	.5	.4	.3	.2	.1	.0
FC_H	CC61 RL	00_H	.7	.6	.5	.4	.3	.2	.1	.0
FD_H	CC61 RH	00_H	.7	.6	.5	.4	.3	.2	.1	.0
FE_H	CC62 RL	00_H	.7	.6	.5	.4	.3	.2	.1	.0
FF_H	CC62 RH	00_H	.7	.6	.5	.4	.3	.2	.1	.0

1) X means that the value is undefined and the location is reserved
2) This register is mapped with RMAP (SYSCON0.4)=0
3) This register is mapped with RMAP (SYSCON0.4)=1
Shaded registers are bit-addressable special function registers

4 On-Chip Peripheral Components

This chapter gives detailed information about all on-chip peripherals of the C868 except for the integrated interrupt controller, which is described separately in chapter 7.

4.1 Ports

The C868 BA has two kinds of ports. The first kind is push-pull ports instead of the traditional quasi-bidirectional ports. The ports belonging to this kind is part of port 1 which is a 5-bit I/O port and port 3 which is an eight-bit I/O port. When configured as inputs, these ports will be high impedance with Schmitt trigger feature. Port 3 is alternate for capture/compare functions whereas, port 1 has alternate functions for some of the pins.

The second kind is dedicated ports which are shared by the some port 1 input, interrupts, timer 2 inputs, capture/compare hall inputs and analog inputs.

4.2 I/O Ports

The I/O part of port 1 and 3 are push-pull ports. Port 1 and port 3 can function as normal I/O ports which have associated SFRs at address 90_H and $B0_H$ respectively. These ports also have alternate functions as listed in **Table 4-2.**

There are three SFRs dedicated for each of these ports. The first one is the port latch (P1/P3) and second one is direction control register (P1DIR/P3DIR) which is used to set the direction for each pin. In P1DIR/P3DIR, if the bit is set to 0 the respective port pin is an output, and 1 means an input. For P1.5-7, when set to output, it is internally connected to the CCU module. After reset, by default all the Port 1 and 3 pins are input. The third one is the alternate function register (P1ALT/P3ALT) which is used to set the function of each pin. When used as alternate function, the direction of the pins has to be set accordingly.

When the bit is set an input, any read operation will return the value at the port. When the bit is set as an output, a read operation will return the latched value if it is part of a read-modify-write operation, otherwise a read operation will return the value at the port.

Note: While in the idle mode or the power down mode the I/O ports hold the last values.

4.2.1 Register Overview

The following table lists the port SFR registers. They contain the value in the port latches.

Table 4-1 Memory Organization Register Overview

Register	Description	Address
P1	Port 1 SFR	**90_H**(SYSCON0.RMAP=0)
P1DIR	Port 1 Direction	**90_H**(SYSCON0.RMAP=1)
P3	Port 3 SFR	**$B0_H$**(SYSCON0.RMAP=0)
P3DIR	Port 3 Direction	**$B0_H$**(SYSCON0.RMAP=1)
P3ALT	Port 3 Alternate Function	**$B1_H$**
P1ALT	Port 1 Alternate Function	**$B4_H$**

P1 and P1DIR is mapped on the same address and depend on the RMAP (SYSCON0.4) bit to select between the two registers. By default (bit = 0), P1 occupies the address. If the bit is set to 1 then P1DIR occupy the address.

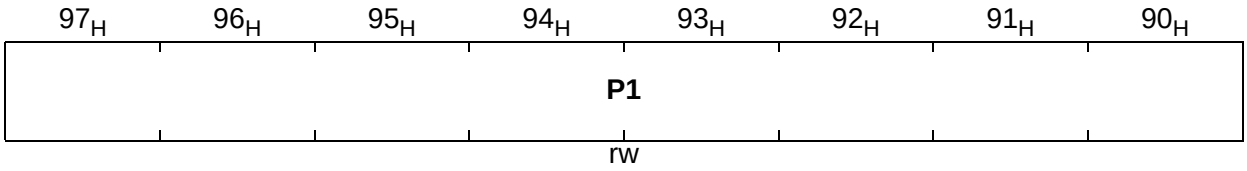
P3 and P3DIR is mapped on the same address and depend on the RMAP (SYSCON0.4) bit to select between the two registers. By default (bit = 0), P3 occupies the address. If the bit is set to 1 then P3DIR occupy the address.

Ports 1 and 3 also serves alternate functions as listed in the **Table 4-2**. To select between the alternate function and normal I/O, registers P1ALT and P3ALT are used. Each can be set to ’1’ for alternate functions, or reset to ’0’ for normal I/O.

Table 4-2 Ports 1 and 3 Alternate Functions

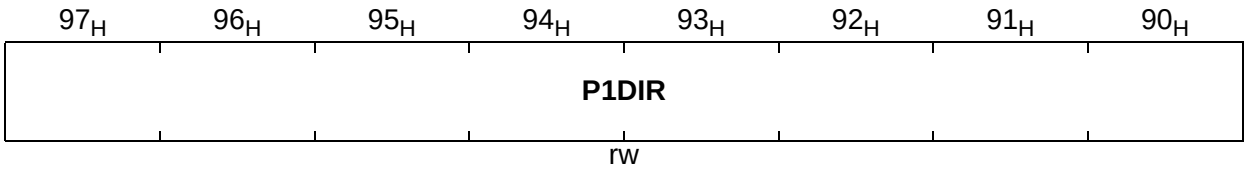
Port	Pin	Alt-Function	Description
1	P1.0	TxD	Transmit data of serial interface
1	P1.1	EXF2	Timer 2 overflow flag
1	P1.3	INT3	Interrupt 3
1	P1.4	RxD	Receive data of serial interface
3	P3.0	COUT63	16 bit compare channel output
3	P3.1	CTRAP	CCU trap input
3	P3.2	COUT62	Output of CCU6 channel 2
3	P3.3	CC62	Input/output of CCU6 channel 2
3	P3.4	COUT61	Output of CCU6 channel 1
3	P3.5	CC61	Input/output of CCU6 channel 1
3	P3.6	COUT60	Output of CCU6 channel 0
3	P3.7	CC60	Input/output of CCU6 channel 0

P1
Port 1 Register **[Reset value: 11111111_B]**



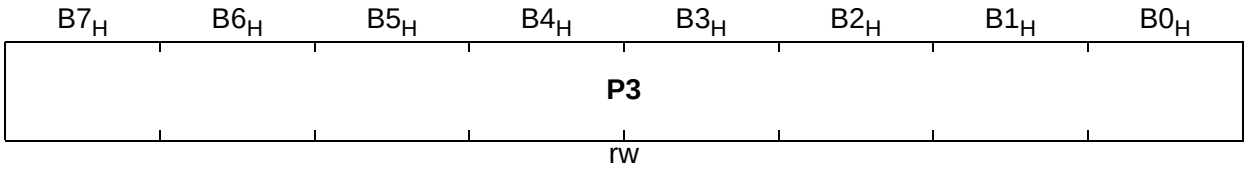
Field	Bits	Typ	Description
P1	[7:0]	rw	**Port 1 Latch**. This SFR appears at address 90_H, only if bit RMAP (SYSCON0.4) is ‘0’.

P1DIR
Port 1 Direction Register **[Reset value: 11111111_B]**



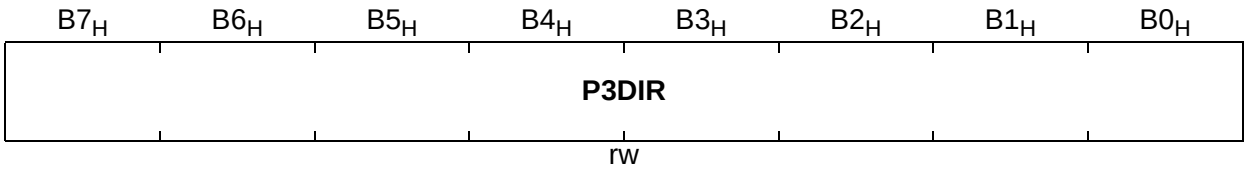
Field	Bits	Typ	Description
P1DIR	[7:0]	rw	**Port 1 Direction Register**. This SFR appears at address 90_H, only if bit RMAP (SYSCON0.4) is ‘1’ 0: The associated pin is an output. 1: The associated pin is an input (default).

P3
Port 3 Register **[Reset value: FF_H]**



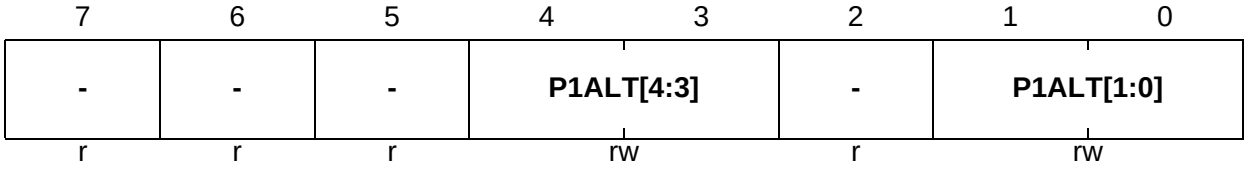
Field	Bits	Typ	Description
P3	[7:0]	rw	**Port 3 Latch**. This SFR appears at address $B0_H$, only if bit RMAP (SYSCON0.4) is ‘0’.

P3DIR
Port 3 Direction Register **[Reset value: FF_H]**



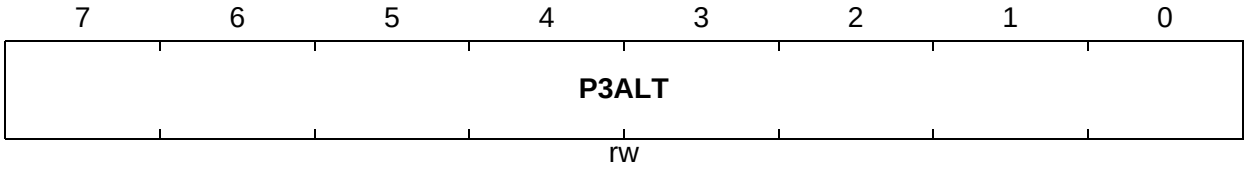
Field	Bits	Typ	Description
P3DIR	[7:0]	rw	**Port 3 Direction Register**. This SFR appears at address $B0_H$, only if bit RMAP (SYSCON0.4) is ‘1’ 0: The associated pin is an output. 1: The associated pin is an input (default).

P1ALT
Port 1 Alternate Function Register **[Reset value: $XXX00X00_B$]**



Field	Bits	Typ	Description
P1ALT.1-0 **P1ALT.4-3**	[1:0] [4:3]	rw	**Port 1 Alternate Function Switch** 0: The associated pin is a normal I/O. (default) 1: The associated pin is an alternate function. Please see **Table 4-2**. All the other bits in this register are reserved for the future use.
-	[7:5],2	r	**reserved;** returns ’0’ if read; should be written with ’0’;

P3ALT
Port 3 Alternate Function Register **[Reset value: 00_H]**



Field	Bits	Typ	Description
P3ALT	[7:0]	rw	**Port 3 Alternate Function Switch** 0: The associated pin is a normal I/O. (default) 1: The associated pin is an alternate function. Please see **Table 4-2**.

4.3 Dedicated Ports

Beside I/O Port, the rest of the ports are dedicated ports. With the exception of pins that are shared with P1.5-7, these dedicated ports do not require bit latches as it uses the 'alternate access'.

ANALOG[4:0]: These are pure analog inputs to the ADC module. The signals from the pin should be rightaway directed to the ADC module. These pins are used as digital input for the input pins of port 1, interrupts, hall inputs to the CCU6 module and inputs to the timer 2 module.

4.4 Port 1, Port 3 Circuitry

The pins of ports 1 and 3 are multifunctional. They are port pins and also serve to implement special features as listed in **Table 4-2**. **Figure 4-1**a) shows a functional diagram of a typical bit latch and I/O buffer, which is the core of the I/O-ports. The bit latch (one bit in the port's SFR) is represented as a type-D flip-flop, which will clock in a value from the internal bus in response to a "write-to-latch" signal from the CPU. The Q output of the flip-flop is placed on the internal bus in response to a "read-latch" signal from the CPU. The level of the port pin itself is placed on the internal bus in response to a "read-pin" signal from the CPU. Some instructions that read from a port (i.e. from the corresponding port SFR P1 and P3) activate the "read-latch" signal, while others activate the "read-pin" signal.

Figure 4-1b) shows a functional diagram for pins P1.5-7. The level of the port pin is placed on the internal bus in response to a "read-pin" signal from the CPU. But the output of the flip-flop is directed internally to the hall inputs of the CCU module. The "write to latch" and "read-latch" behave the same as in a). As output, these pins can be used to stimulate the hall inputs of the CCU module for algorithm testing without the use of external circuitry. As input, they can be used to verify the state of the hall input pins.

Figure 4-1 shows a functional diagram of a port latch with alternate function.

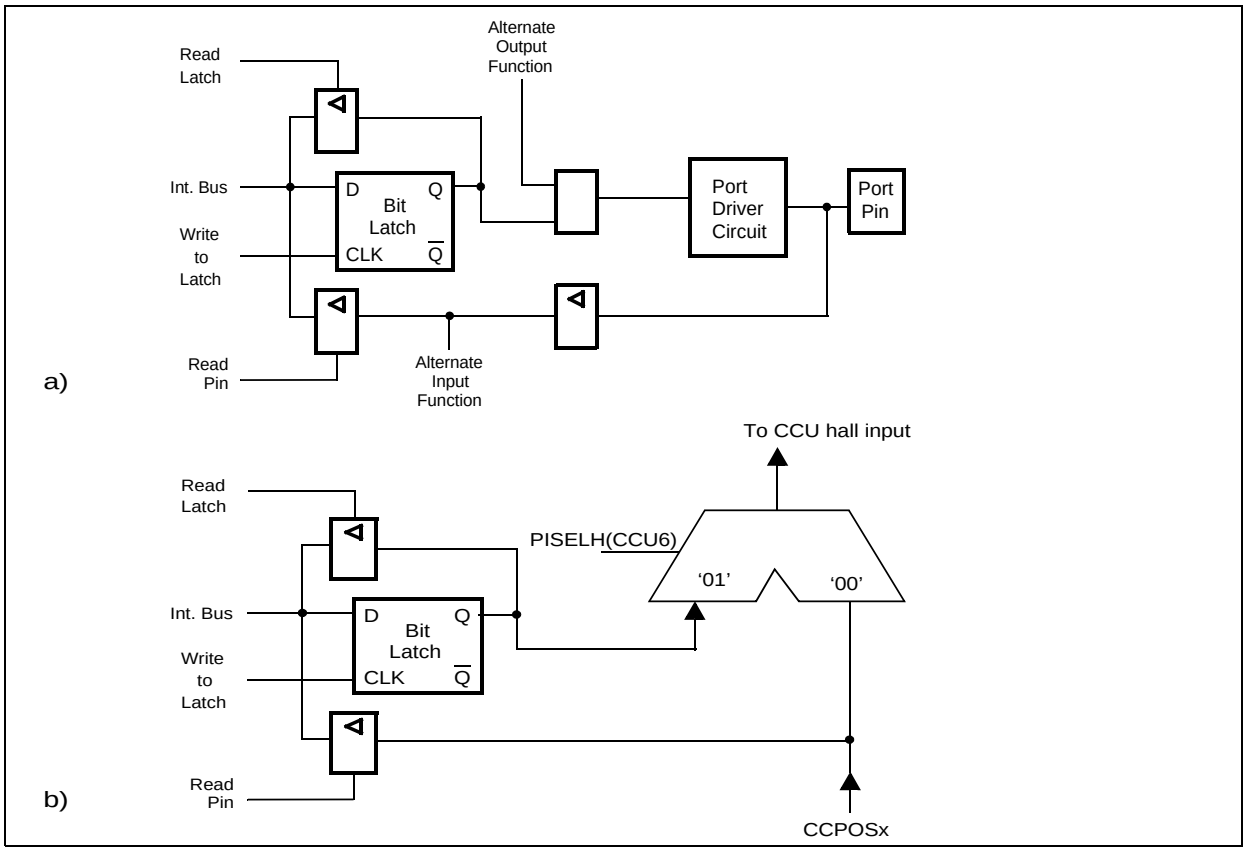


Figure 4-1 Ports 1 and 3

4.4.1 Read-Modify-Write Feature of Ports

Some port-reading instructions read the latch and others read the pin. The instructions reading the latch rather than the pin read a value, possibly change it, and then rewrite it to the latch. These are called "read-modify-write"- instructions, which are listed in 4-3. If the destination is a port or a port pin, these instructions read the latch rather than the pin. Note that all other instructions which can be used to read a port, exclusively read the port pin. In any case, reading from latch or pin, resp., is performed by reading the SFR P3; for example, "MOV A, P3" reads the value from port 3 pins, while "ANL P3, #0AAH" reads from the latch, modifies the value and writes it back to the latch.

It is not obvious that the last three instructions in **Figure 4-3** are read-modify-write instructions, but they are. The reason is that they read the port byte, all 8 bits, modify the addressed bit, then write the complete byte back to the latch.

Table 4-3 "Read-Modify-Write"-Instructions

Instruction	Function
ANL	Logic AND; e.g. ANL P1, A
ORL	Logic OR; e.g. ORL P2, A
XRL	Logic exclusive OR; e.g. XRL P3, A
JBC	Jump if bit is set and clear bit; e.g. JBC P1.1, LABEL
CPL	Complement bit; e.g. CPL P3.0
INC	Increment byte; e.g. INC P4
DEC	Decrement byte; e.g. DEC P5
DJNZ	Decrement and jump if not zero; e.g. DJNZ P3, LABEL
MOV Px.y,C	Move carry bit to bit y of port x
CLR Px.y	Clear bit y of port x
SETB Px.y	Set bit y of port x

The reason why read-modify-write instructions are directed to the latch rather than the pin is to avoid a possible misinterpretation of the voltage level at the pin. For example, a port bit might be used to drive the base of a transistor. When a "1" is written to the bit, the transistor is turned on. If the CPU then reads the same port bit at the pin rather than the latch, it will read the base voltage of the transitor (approx. 0.7 V, i.e. a logic low level!) and interpret it as "0". For example, when modifying a port bit by a SETB or CLR instruction, another bit in this port with the above mentioned configuration might be changed if the value read from the pin were written back to th latch. However, reading the latch rater than the pin will return the correct value of "1".

4.5 Timers/Counters

The C868 contains three 16-bit timers/counters, timer 0, timer 1 and timer/counter 2, which are useful in many applications for timing and counting.

The timer register is incremented every machine cycle. Thus one can think of it as counting machine cycles. Since a machine cycle consists of 12 periods, the counter rate is 1/12 of the system frequency.

4.5.1 Timer 0 and 1

Timer 0 and 1 of the C868 are fully compatible with timer 0 and 1 can be used in the same four operating modes:

Mode 0:8-bit timer with a divide-by-32 prescaler

Mode 1:16-bit timer

Mode 2:8-bit timer with 8-bit auto-reload

Mode 3:Timer 0 is configured as two 8-bit timers. Timer 1 in this mode holds its count. The effect is the same as setting TR1 = 0.

External inputs $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ can be programmed to function as a gate for timer 0 and 1 to facilitate pulse width measurements.

Each timer consists of two 8-bit registers (TH0 and TL0 for timer 0, TH1 and TL1 for timer 1) which may be combined to one timer configuration depending on the mode that is established. The functions of the timers are controlled by two special function registers TCON and TMOD.

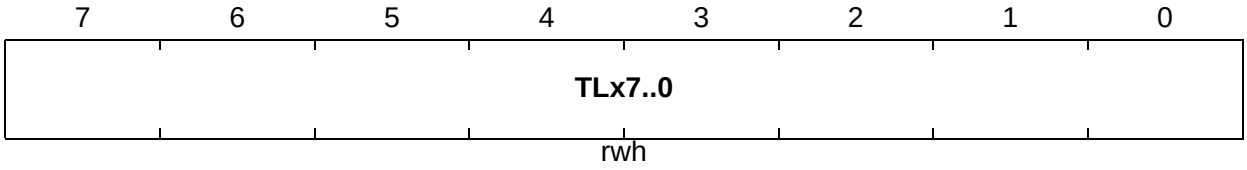
In the following descriptions the symbols TH0 and TL0 are used to specify the high-byte and the low-byte of timer 0 (TH1 and TL1 for timer 1, respectively). The operating modes are described and shown for timer 0. If not explicity noted, this applies also to timer 1.

4.5.1.1 Timer 0 and 1 Registers

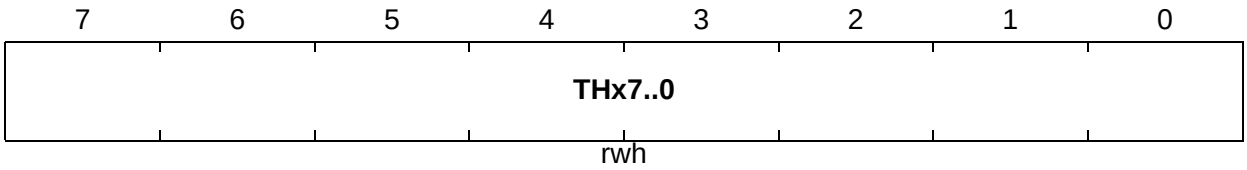
Totally seven special function registers control the timer 0 and 1 operation :

- TL0/TH0 and TL1/TH1 - timer registers, low and high part
- TCON and IEN0 - control and interrupt enable
- TMOD - mode select

TLx(x=0..1)
Timer x Low Register **[Reset value: 00_H]**



THx(x=0..1)
Timer x High Register **[Reset value: 00_H]**



Field	Bits	Typ	Description
TLx.7-0(x=0..1)	[7:0]	rwh	**Timer/counter 0/1 low register**

Operating Mode	Description
0	"TLx" holds the 5-bit prescaler value.
1	"TLx" holds the lower 8-bit part of the 16-bit timer/ counter value.
2	"TLx" holds the 8-bit timer/ counter value.
3	TL0 holds the 8-bit timer/ counter value; TL1 is not used.

Field	Bits	Typ	Description
THx.7-0(x=0..1)	[7:0]	rwh	**Timer 0/1 high register** **Operating Mode** / **Description** 0 – "THx" holds the 8-bit timer value. 1 – "THx" holds the higher 8-bit part of the 16-bit timer value 2 – "THx" holds the 8-bit reload value. 3 – TH0 holds the 8-bit timer value; TH1 is not used.

TCON
Power Control Register

[Reset value: 00_H]

$8F_H$	$8E_H$	$8D_H$	$8C_H$	$8B_H$	$8A_H$	89_H	88_H
TF1	**TR1**	**TF0**	**TR0**	**IE1**	**IT1**	**IE0**	**IT0**
rwh	rw	rwh	rw	rwh	rw	rwh	rw

The functions of the shaded bits are not described here

Field	Bits	Typ	Description
TR0	4	rw	**Timer 0 run control bit** Set/cleared by software to turn timer 0 ON/OFF.
TF0	5	rwh	**Timer 0 overflow flag** Set by hardware on timer overflow. Cleared by hardware when processor vectors to interrupt routine.
TR1	6	rw	**Timer 1 run control bit** Set/cleared by software to turn timer 1 ON/OFF.
TF1	7	rwh	**Timer 1 overflow flag** Set by hardware on timer overflow. Cleared by hardware when processor vectors to interrupt routine.

TMOD
Timer Register **[Reset value: 00_H]**

7	6	5	4	3	2	1	0
GATE1	**C/NT1**	**M1(1)**	**M0(1)**	**GATE0**	**C/NT0**	**M1(0)**	**M0(0)**
rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Typ	Description
M0(0) **M1(0)** **M0(1)** **M1(1)**	0 1 4 5	rw	Mode select bits **M1(x) / M0(x) / Function** 0 / 0 / 8-bit timer: "THx" operates as 8-bit timer "TLx" serves as 5-bit prescaler 0 / 1 / 16-bit timer. "THx" and "TLx" are cascaded; there is no prescaler 1 / 0 / 8-bit auto-reload timer. "THx" holds a value which is to be reloaded into "TLx" each time it overflows 1 / 1 / Timer 0 : TL0 is an 8-bit timer controlled by the standard timer 0 control bits. TH0 is an 8-bit timer only controlled by timer 1 control bits. Timer 1 : Timer/counter 1 stops
C/NT0 **C/NT1**	2 6	rw	**Counter or timer select bit** Cleared for timer operation (input from internal system clock).
GATE0 **GATE1**	3 7	rw	**Gating control** When set, timer "x" is enabled only while "INT x" pin is high and "TRx" control bit is set.

IEN0
Interrupt Enable Register

[Reset value: 00_H]

AF_H	AE_H	AD_H	AC_H	AB_H	AA_H	$A9_H$	$A8_H$
EA	**-**	**ET2**	**ES**	**ET1**	**EX1**	**ET0**	**EX0**
rw	r	rw	rw	rw	rw	rw	rw

The functions of the shaded bits are not described here

Field	Bits	Typ	Description
ET0	1	rw	**Timer 0 overflow interrupt enable.** If ET0 = 0, the timer 0 interrupt is disabled.
ET1	3	rw	**Timer 1 overflow interrupt enable.** If ET1 = 0, the timer 1 interrupt is disabled.

4.5.1.2 Mode 0

Putting either timer 0, 1 into mode 0 configures it as an 8-bit timer with a divide-by-32 prescaler. **Figure 4-2** shows the mode 0 operation.

In this mode, the timer register is configured as a 13-bit register. As the count rolls over from all 1's to all 0's, it sets the timer overflow flag TF0. The overflow flag TF0 then can be used to request an interrupt. The counted input is enabled to the timer when TR0 = 1 and either Gate = 0 or $\overline{INT0}$ = 1 (setting Gate = 1 allows the timer to be controlled by external input $\overline{INT0}$, to facilitate pulse width measurements). TR0 is a control bit in the special function register TCON; Gate is in TMOD.

The 13-bit register consists of all 8 bits of TH0 and the lower 5 bits of TL0. The upper 3 bits of TL0 are indeterminate and should be ignored. Setting the run flag (TR0) does not clear the registers.

Mode 0 operation is the same for timer 0 as for timer 1. Substitute TR0, TF0, TH0, TL0 and $\overline{INT0}$ for the corresponding timer 1 signals in **Figure 4-2**. There are two different gate bits, one for timer 1 (TMOD.7) and one for timer 0 (TMOD.3).

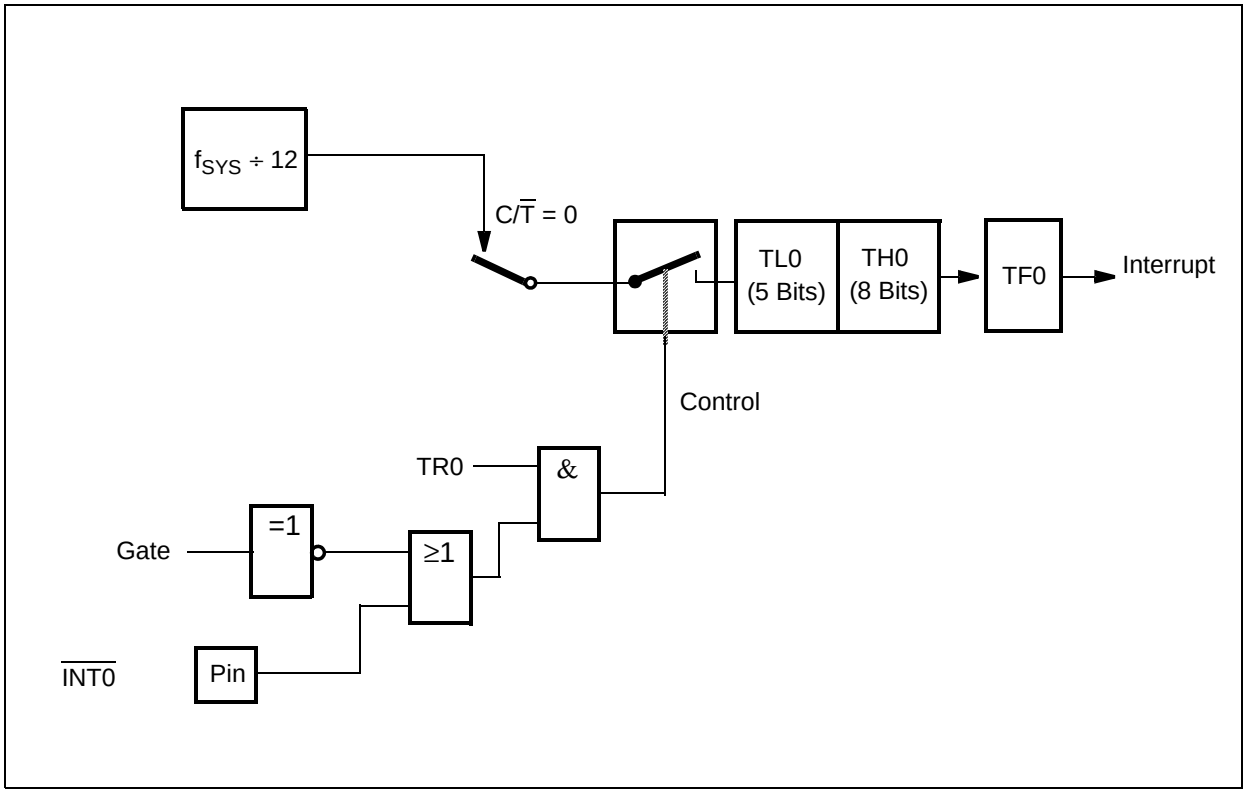


Figure 4-2 Timer 0, Mode 0: 13-Bit Timer

4.5.1.3 Mode 1

Mode 1 is the same as mode 0, except that the timer register is running with all 16 bits. Mode 1 is shown in **Figure 4-3**.

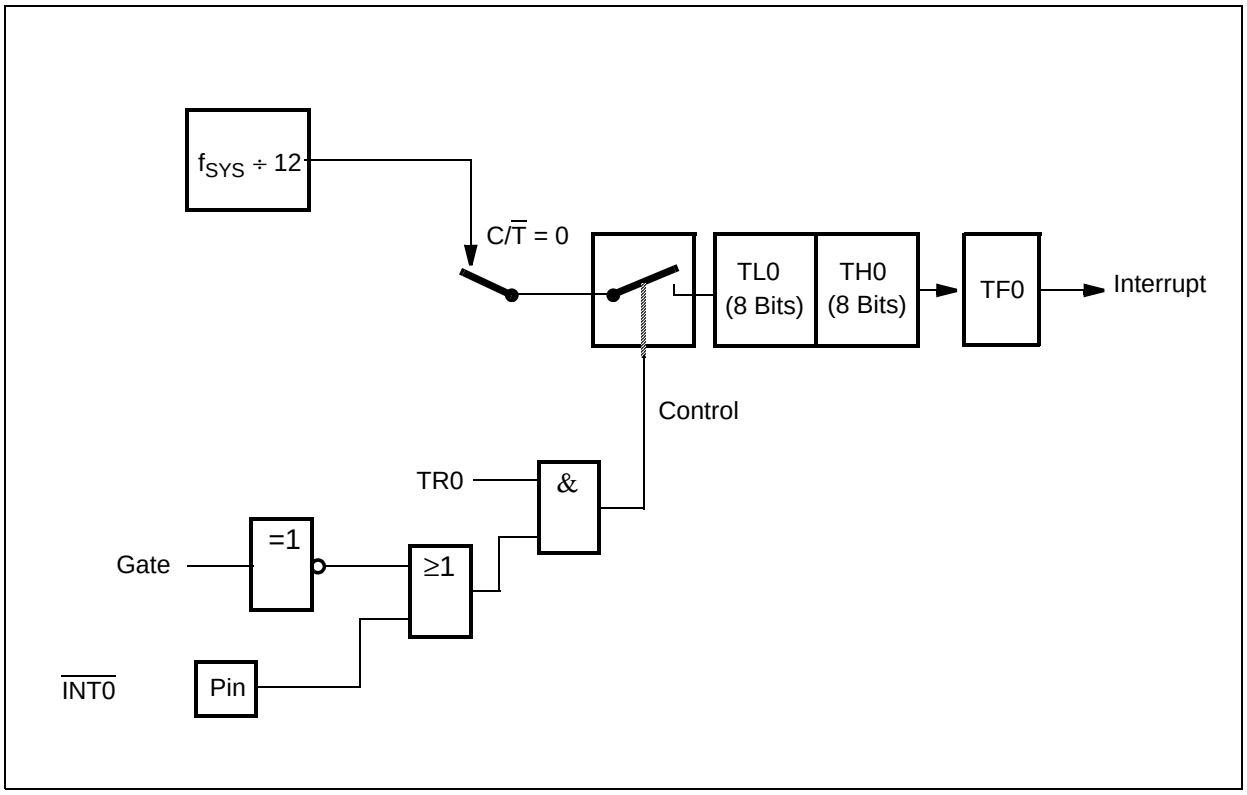


Figure 4-3 Timer 0, Mode 1: 16-Bit Timer

4.5.1.4 Mode 2

Mode 2 configures the timer register as an 8-bit counter (TL0) with automatic reload, as shown in **Figure 4-4**. Overflow from TL0 not only sets TF0, but also reloads TL0 with the contents of TH0, which is preset by software. The reload leaves TH0 unchanged.

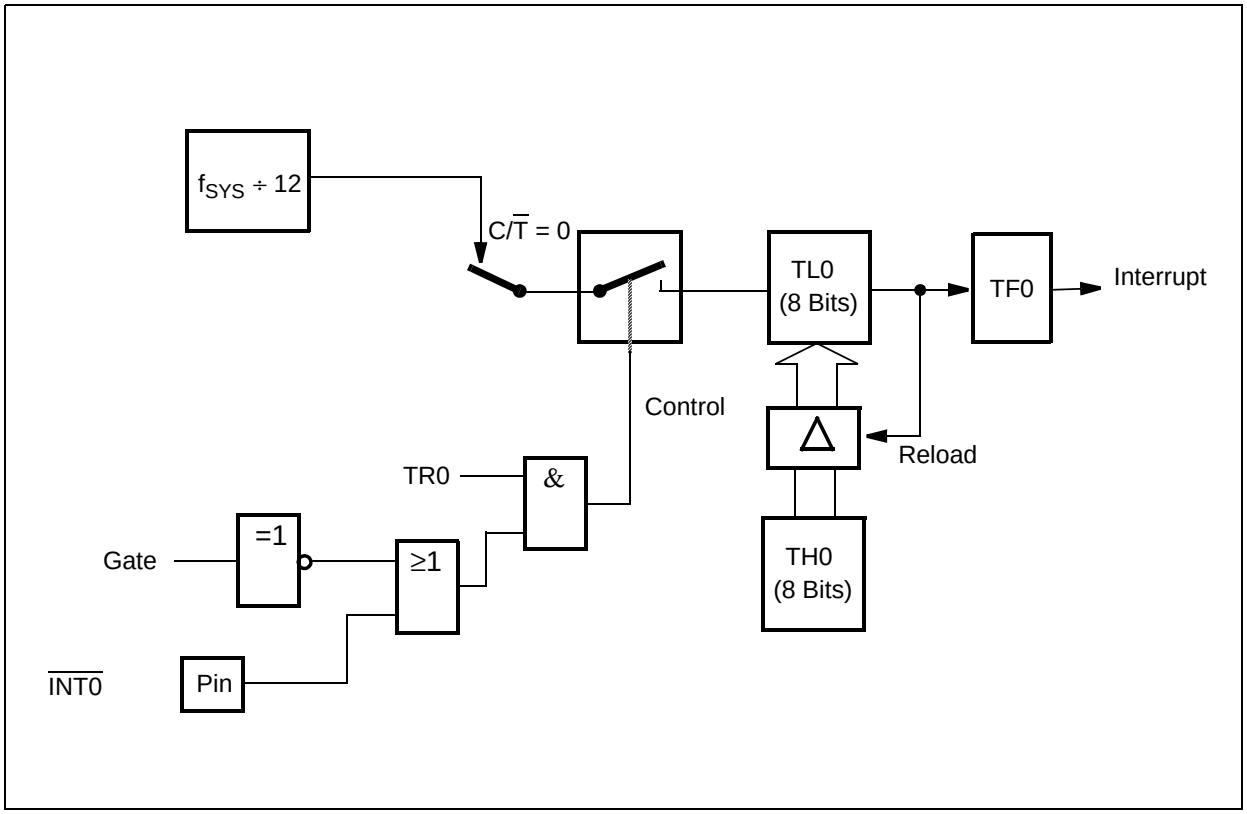


Figure 4-4 Timer 0,1, Mode 2: 8-Bit Timer with Auto-Reload

4.5.1.5 Mode 3

Mode 3 has different effects on timer 0 and timer 1. Timer 1 in mode 3 simply holds its count. The effect is the same as setting TR1=0. Timer 0 in mode 3 establishes TL0 and TH0 as two separate counters. The logic for mode 3 on timer 0 is shown in **Figure 4-5**. TL0 uses the timer 0 control bits: C/$\overline{T}$, Gate, TR0, $\overline{INT0}$ and TF0. TH0 is locked into a timer function (counting machine cycles) and takes over the use of TR1 and TF1 from timer 1. Thus, TH0 now controls the "timer 1" interrupt.

Mode 3 is provided for applications requiring an extra 8-bit timer or counter. When timer 0 is in mode 3 and when TR1 is set, timer 1 can be turned on by switching it to any mode other than 3 and off by switching it into its own mode 3, or can still be used by the serial channel as a baud rate generator, or in fact, in any application not requiring an interrupt from timer 1 itself.

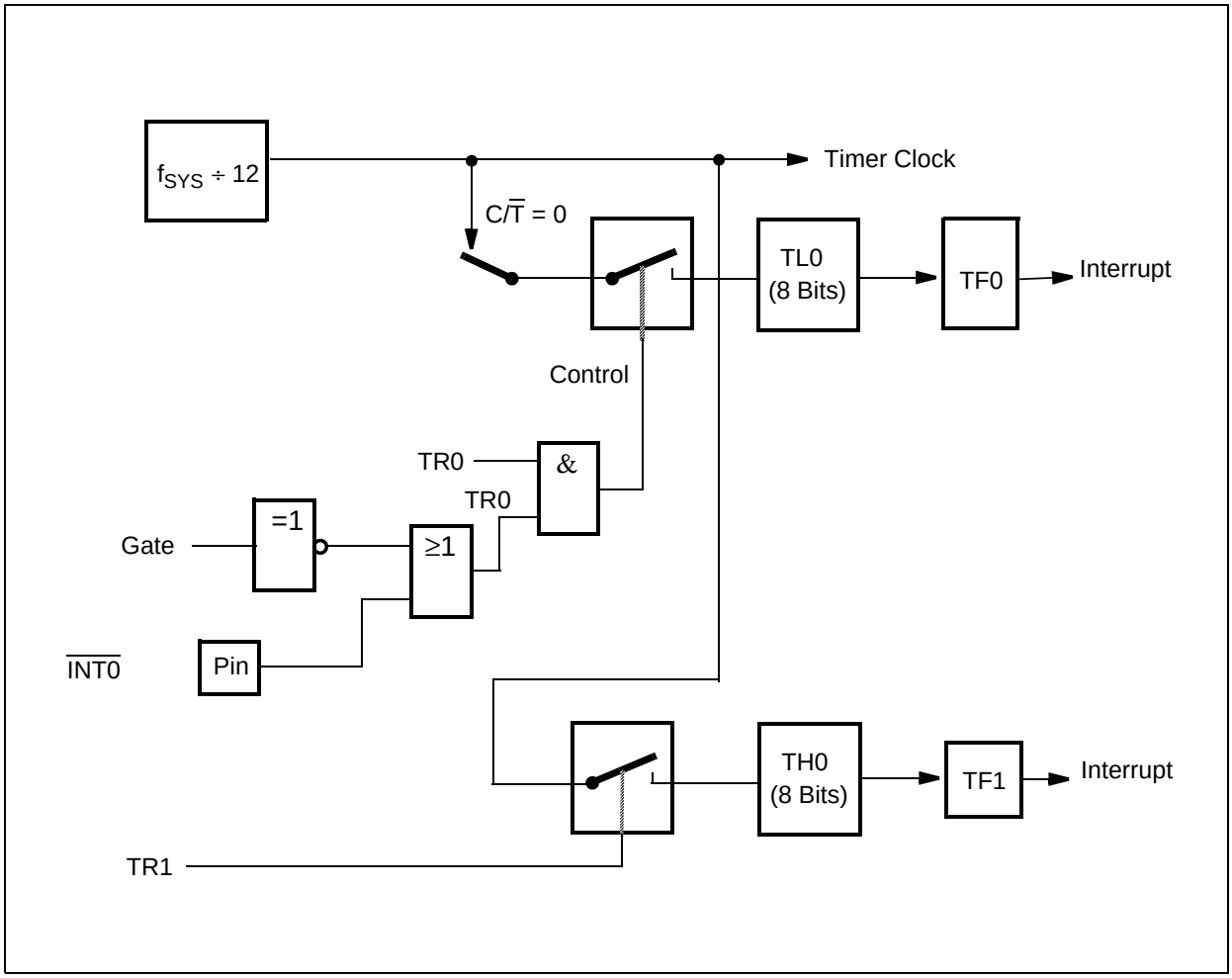


Figure 4-5 Timer 0, Mode 3: Two 8-Bit Timers

4.6 Functional Description of Timer/Counter 2

Timer two serves as a 16-bit timer/counter for which is also capable of being used as a baudrate generator for the UART module.

4.6.1 Features

- 16-bit auto-reload mode
  - selectable up or down counting
- one channel 16-bit capture mode
- Baudrate generator for UART
- Timer/counter powerdown in normal, idle and slow-down modes

4.6.2 Overview

Timer 2 is a 16-bit general purpose timer/counter which can additionally function as a baudrate generator. This module is functionally compatible to the Timer 2 in the C501 product family.

Timer/counter 2 can function as a timer or counter in each of its modes. As a timer, it counts with an input clock of $f_{SYS}/12$. As a counter, it counts 1-to-0 transitions on pin T2. In the counter mode the maximum resolution for the count is $f_{SYS}/24$.

4.6.3 Register Description

The T2CON register is used for controlling the various modes of timer/counter 2 module. Additionally, this register also indicates the status of the timer/counter 2 functions (flags).

T2MOD
Timer 2 Mode Register,Low Byte **[Reset value: 00_H]**

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	**DCEN**
r	r	r	r	r	r	r	rw

Field	Bits	Typ	Description
DCEN	0	rw	**Up/Down Counter Enable** 0 :Up/Down Counter function is disabled 1 :Up/Down Counter function is enabled and controlled by pin T2EX (Up=1,Down=0)
-	[7:6]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

T2CON
Timer 2 Control Register,Low Byte **[Reset value: 00_H]**

CF_H	CE_H	CD_H	CC_H	CB_H	CA_H	$C9_H$	$C8_H$
TF2	**EXF2**	**RCLK**	**TCLK**	**EXEN2**	**TR2**	**C/$\overline{T2}$**	**CP/$\overline{RL2}$**
rwh	rwh	rw	rw	rw	rw	rw	rw

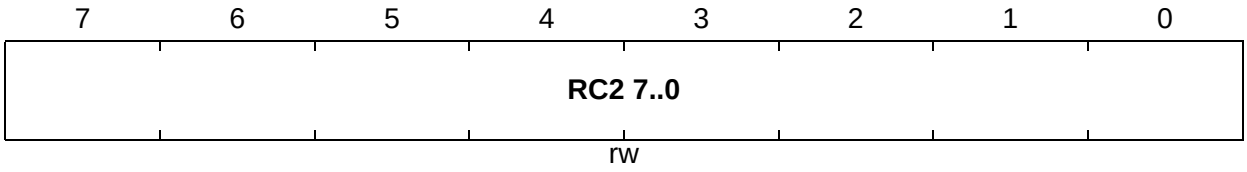
Field	Bits	Typ	Description
CP/$\overline{RL2}$	0	rw	**Capture / Reload Select** 0 :Reload upon overflow or upon negative transition at pin T2EX (when EXEN2=1) 1 :Capture timer/counter 2 data register contents on negative transition at pin T2EX, provided EXEN2 = 1 If TCLK/RCLK=1, this bit is ignored.
C/$\overline{T2}$	1	rw	**Timer or Counter Select** 0 :Timer function selected 1 :Count upon negative edge at pin T2
TR2	2	rw	**Timer/counter 2 Start / Stop Control** 0 :Stop Timer/counter 2 1 :Start Timer/counter 2
EXEN2	3	rw	**Timer/counter 2 External Enable Control** 0 :External events are disabled 1 :External events Capture/Reload enabled
TCLK	4	rw	**Transmit Clock Enable** 0 :Timer/counter 2 overflow is not used for UART transmitter clock 1 :Timer/counter 2 overflow is used for UART transmitter clock
RCLK	5	rw	**Receiver Clock Enable** 0 :Timer/counter 2 overflow is not used for UART receiver clock 1 :Timer/counter 2 overflow is used for UART receiver clock

Field	Bits	Typ	Description
EXF2	6	rwh	**Timer/counter 2 External Flag** This bit is set by hardware when a capture/reload occurred upon a negative transition at pin T2EX, if bit EXEN=1. An interrupt request to the core is generated, unless bit DCEN=1. This bit must be cleared by software.
TF2	7	rw	**Timer/counter 2 Overflow Flag** Set by a timer/counter 2 overflow. Must be cleared by software. TF2 will not be set when either RCLK=1 or TCLK=1.

The RC2L/H registers are used for a 16-bit reload of the timer/counter count upon overflow or a capture of current timer/counter count depending on the mode selected.

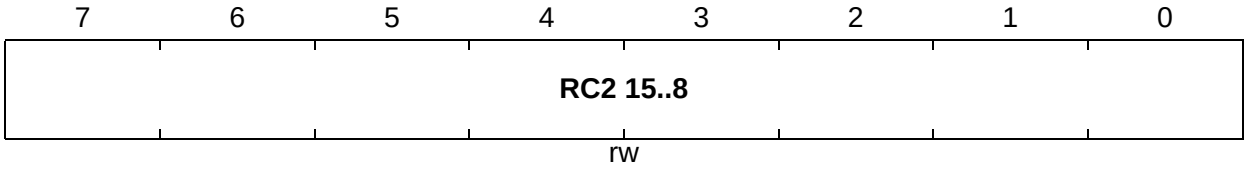
RC2L

Timer 2 Reload/Capture Register, Low Byte **[Reset value: 00_H]**



RC2H

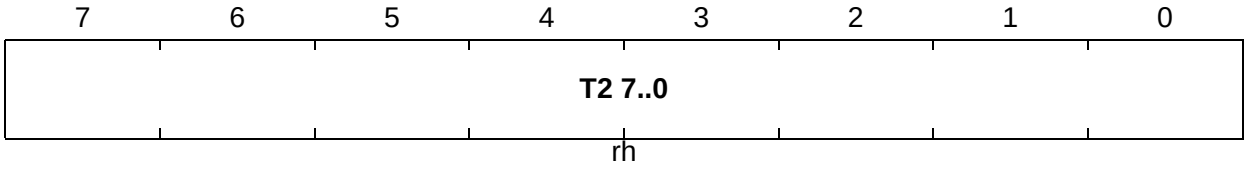
Timer 2 Reload/Capture Register, High Byte **[Reset value: 00_H]**



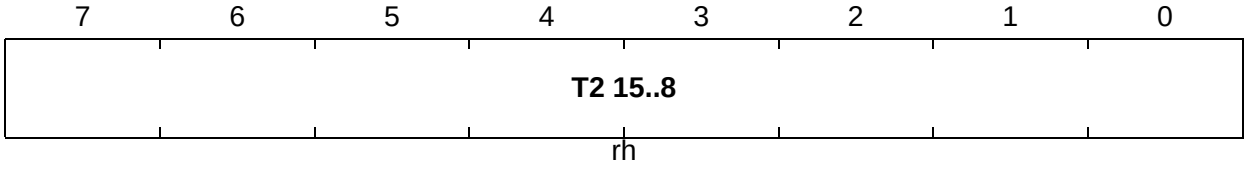
Field	Bits	Typ	Description
RC2	[7:0] of RC2L, [7:0] of RC2H	rw	**Reload/Capture Value** These contents are loaded into the timer/counter registers upon an overflow condition, if CP/RL2=0. If CP/RL2 = 1, this registers are loaded with the current timer count upon a negative transition at pin T2EX when EXEN2=1.

The T2L/H registers holds the current 16-bit value of the Timer 2 count.

T2L
Timer 2, Low Byte **[Reset value: 00_H]**



T2H
Timer 2, High Byte **[Reset value: 00_H]**



Field	Bits	Typ	Description
T2	[7:0] of T2L, [7:0] of T2H	rh	**Timer 2 Value** These bits indicate the current timer value.

T2DIS in PMCON1 register controls the powerdown of timer/counter 2, T2ST in PMCON2 shows the power status of timer/counter 2.

PMCON1
Peripheral Management Control Register **[Reset value: XXXXX000$_B$]**

7	6	5	4	3	2	1	0
-	-	-	-	-	CCUDIS	T2DIS	ADCDIS
r	r	r	r	r	rw	rw	rw

The functions of the shaded bits are not described here

Field	Bits	Typ	Description
T2DIS	1	rw	**Timer 2 Disable Request.** 0 : Timer 2 will continue normal operation. (default) 1 : Request to disable the Timer 2 is active.

PMCON2
Peripheral Management Status Register **[Reset value: XXXXX000$_B$]**

7	6	5	4	3	2	1	0
-	-	-	-	-	CCUST	T2ST	ADCST
r	r	r	r	r	rh	rh	rh

The functions of the shaded bits are not described here

Field	Bits	Typ	Description
T2ST	1	rh	**Timer 2 Disable Status** 0 : Timer 2 is not disabled. (default) 1 : Timer 2 is disabled, clock is gated off.

4.6.4 Operating Mode Selection

The operating mode of timer/counter 2 is controlled by register T2CON. This register serves two purposes:

- during initialization it provides access to a set of control bits
- during timer operation it provides access to a set of status flags.

The different modes of operation are:

- Auto-Reload Mode,
- Capture Mode, and
- Baudrate Generator Mode

4.6.5 Auto-Reload Mode

In the auto-reload mode, timer/counter 2 counts to an overflow value and then reloads its registers contents with a 16-bit value start value for a fresh counting sequence. The overflow condition is indicated by setting the bit TF2 in the T2CON register. This will then generate an interrupt request to the core by an active high signal. The overflow flag TF2 must be cleared by software.

The auto-reload mode is further classified into two categories depending upon the DCEN control bit.

4.6.5.1 Up/Down Count Disabled

If DCEN=0, the up-down count selection is disabled. The timer/counter, therefore, functions as a pure up timer/counter only. The operational block diagram is shown in **Figure 4-6**.

In this mode if EXEN2=0, the timer/counter starts to count up to a maximum of $FFFF_H$, once TR2 is set. Upon overflow, bit TF2 is set and the timer register is reloaded with a 16-bit reload of the RC2L/H registers. A fresh count sequence is started and the timer/counter counts up from this reload value as in the previous count sequence. This reload value is chosen by software, prior to the occurrence of an overflow condition.

If EXEN2=1, the timer/counter counts up to a maximum to $FFFF_H$, once TR2 is set. A 16-bit reload of the timer registers from register RC2L/H is triggered either by an overflow condition or by a negative edge at input pin T2EX. If an overflow caused the reload, the overflow flag TF2 is set. If a 1-to-0 transition at pin T2EX caused a reload, bit EXF2 is set. In either case, an interrupt is generated to the core and the timer/counter proceeds to its next count sequence. The EXF2 flag, similar to the TF2, must be cleared by software.

Note: In counter mode, if the reload via T2EX and the count clock T2 are detected simultaneously the reload takes precedence over the count. The counter increments its value with the following T2 count clock.

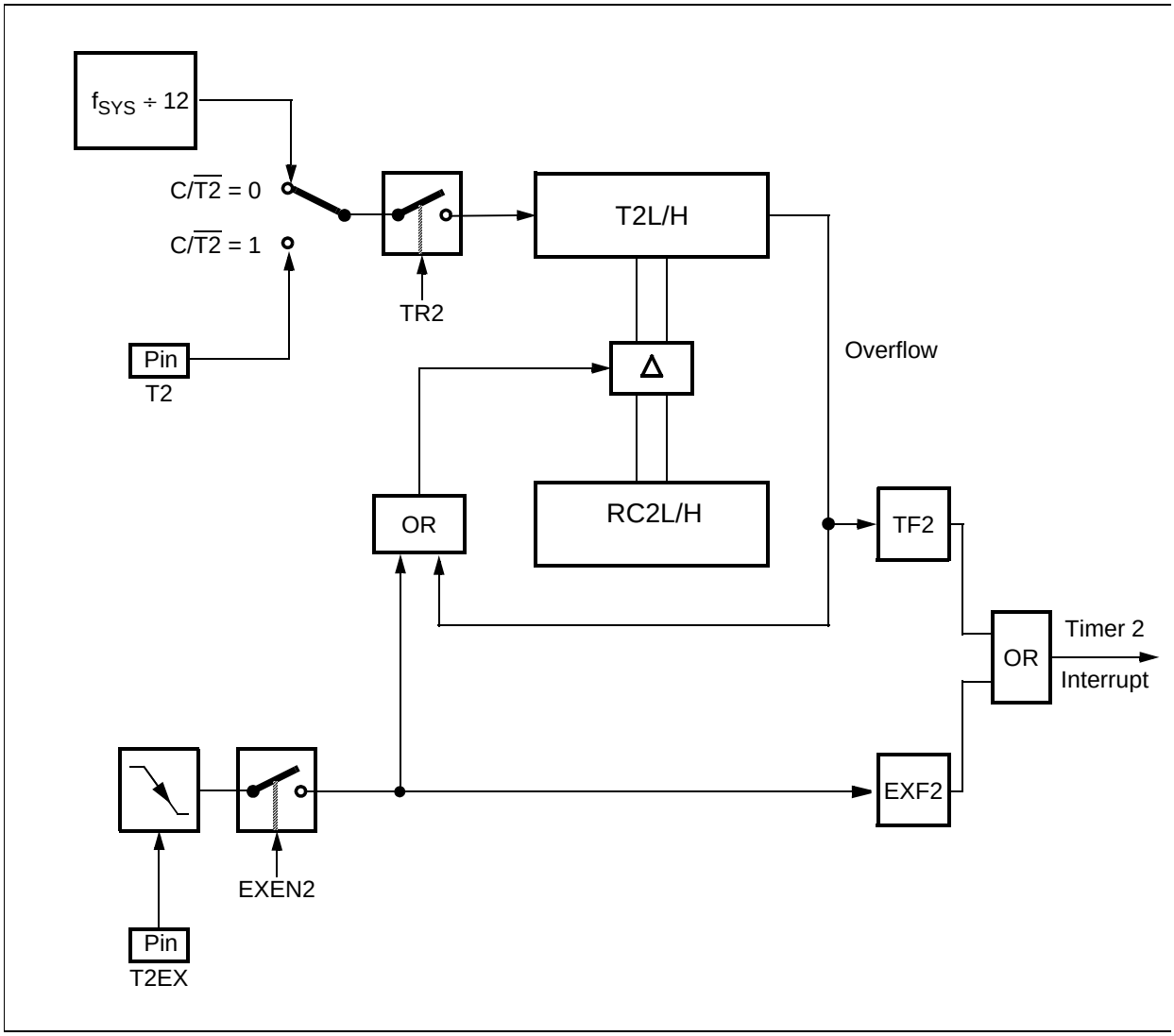


Figure 4-6 Auto-Reload Mode (DCEN = 0)

4.6.5.2 Up/Down Count Enabled

If DCEN=1, the up-down count selection is enabled. The direction of count is determined by the level at input pin T2EX. The operational block diagram is shown in **Figure 4-7**.

A logic 1 at pin T2EX sets the timer/counter 2 to up counting mode. The timer/counter, therefore, counts up to a maximum of $FFFF_H$. Upon overflow, bit TF2 is set and the timer/counter registers are reloaded with a 16-bit reload of the RC2L/H registers. A fresh count sequence is started and the timer counts up from this reload value as in the previous count sequence. This reload value is chosen by software, prior to the occurrence of an overflow condition.

A logic 0 at pin T2EX sets the timer/counter 2 to down counting mode. The timer/counter counts down and underflows when the T2L/H value reaches the value stored at registers

RC2L/H. The underflow condition sets the TF2 flag and causes $FFFF_H$ to be reloaded into the T2L/H registers. A fresh down counting sequence is started and the timer/ counter counts down as in the previous counting sequence.

In this mode, bit EXF2 toggles whenever an overflow or an underflow condition is detected. This flag, however, does not generate an interrupt request.

Note: In counter mode, if the reload via T2EX and the count clock T2 are detected simultaneously the reload takes precedence over the count. The counter increments its value with the following T2 count clock.

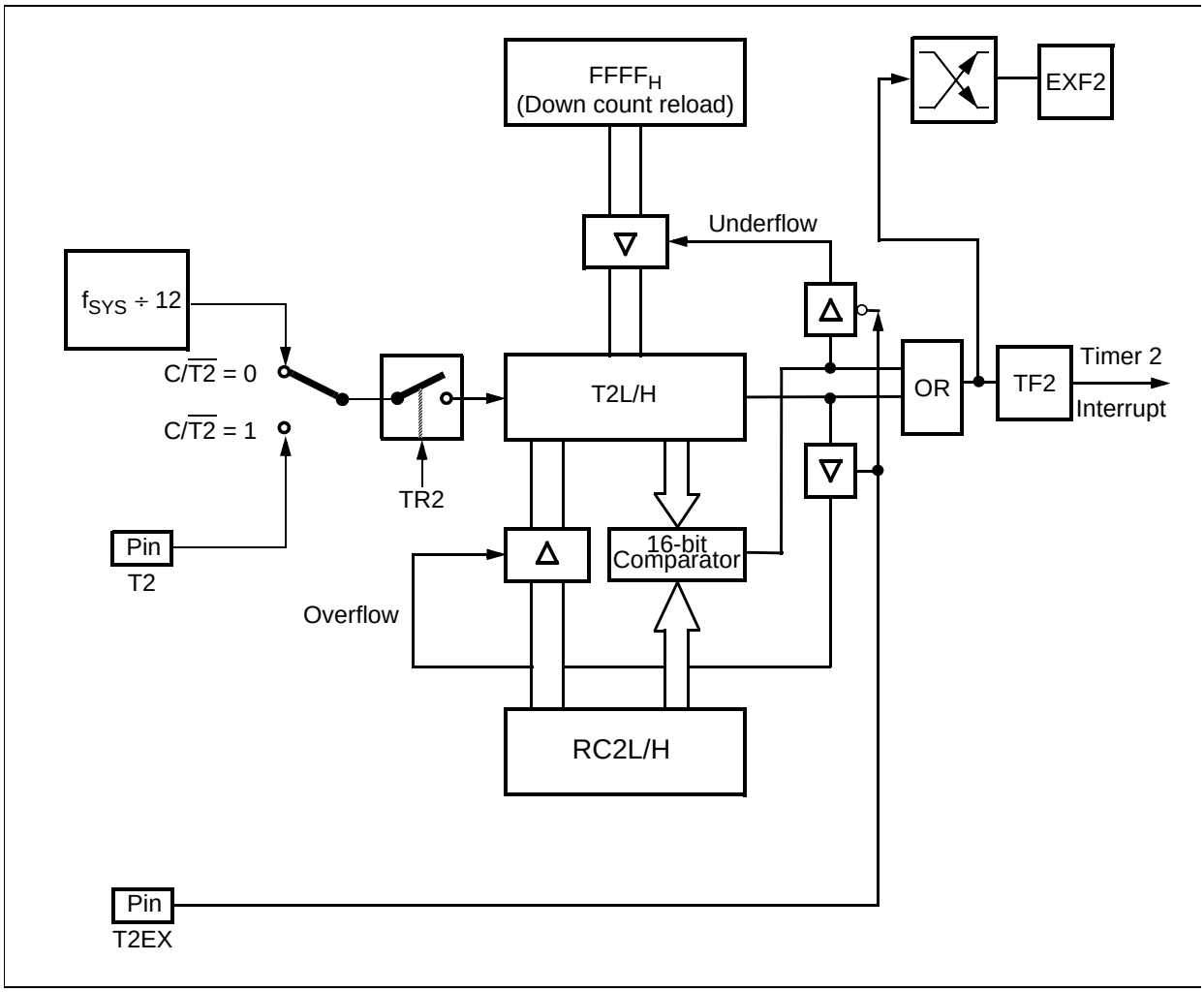


Figure 4-7 Auto-Reload Mode (DCEN = 1)

4.6.6 Capture Mode

In order to enter the 16-bit capture mode, bits CP/$\overline{RL2}$ and EXEN2 in register T2CON must be set. In this mode, the down count function must remain disabled. The timer/counter functions as a 16-bit timer or counter and always counts up to $FFFF_H$ and overflows. Upon an overflow condition, bit TF2 is set and the timer/counter reloads its registers with 0000_H. The setting of TF2 generates an interrupt request to the core.

Additionally, with a falling edge on pin T2EX the contents of the timer/counter registers (T2L/H) are captured into the RC2L/H registers. If the capture signal is detected while the counter is being incremented, the counter is first incremented before the capture operation is performed. This ensures that the latest value of the timer/counter registers are always captured.

When the capture operation is completed, bit EXF2 is set and can be used to generate an interrupt request. **Figure 4-8** describes the capture function of timer/counter 2.

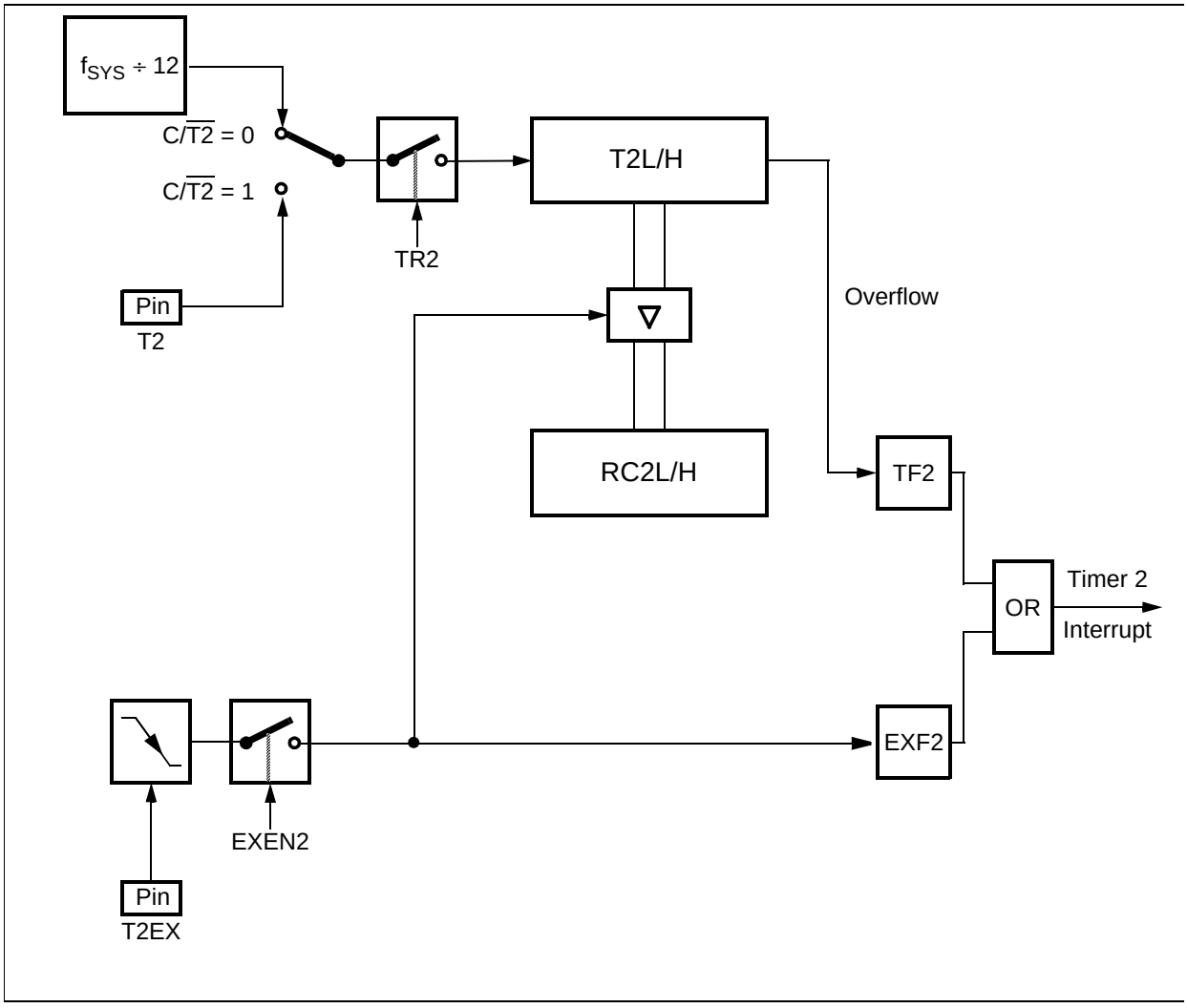


Figure 4-8 Capture Mode

4.6.7 Baudrate Generator Mode

The baudrate generator mode of timer/counter 2 can be selected by setting the bits TCLK and/or RCLK in register T2CON. So the baudrate for the receive and transmit functions can be individually controlled. The timer/counter itself functions similar to the auto-reload mode with up/down counting is disabled. The timer/counter counts up and overflows but the overflow condition does not set the TF2 flag. An interrupt request to the core is not generated. Upon an overflow condition, the timer/counter registers are reloaded with the RC2L/H registers content and continues counting as before.

The overflow signal is provided as an output of the timer/counter 2 block. This is active for one clock cycle only. Additionally, the status of the TCLK and RCLK bits are also provided as outputs of the timer 2 block. The UART, for e.g., could use these signals to control its baudrates.

The main difference between the auto-reload mode and the baudrate generator mode is that timer 2 as a baudrate generator uses $f_{SYS}/2$ as the count clock. In the auto-reload mode the timer/counter 2 uses $f_{SYS}/12$ as the count clock.

If EXEN2=1, in the baudrate generator mode, a falling edge on pin T2EX can be used to generate an interrupt. In this case, flag EXF2 will be set.

Note: When timer/counter 2 is in the baudrate generator mode, an increment of the timer register happens for every other f_{SYS}. Therefore, software should not access the T2L/H registers. Software may however, read the RC2L/H registers. Software write into these registers may coincide with a timer update or reload cycle and should therefore be avoided.

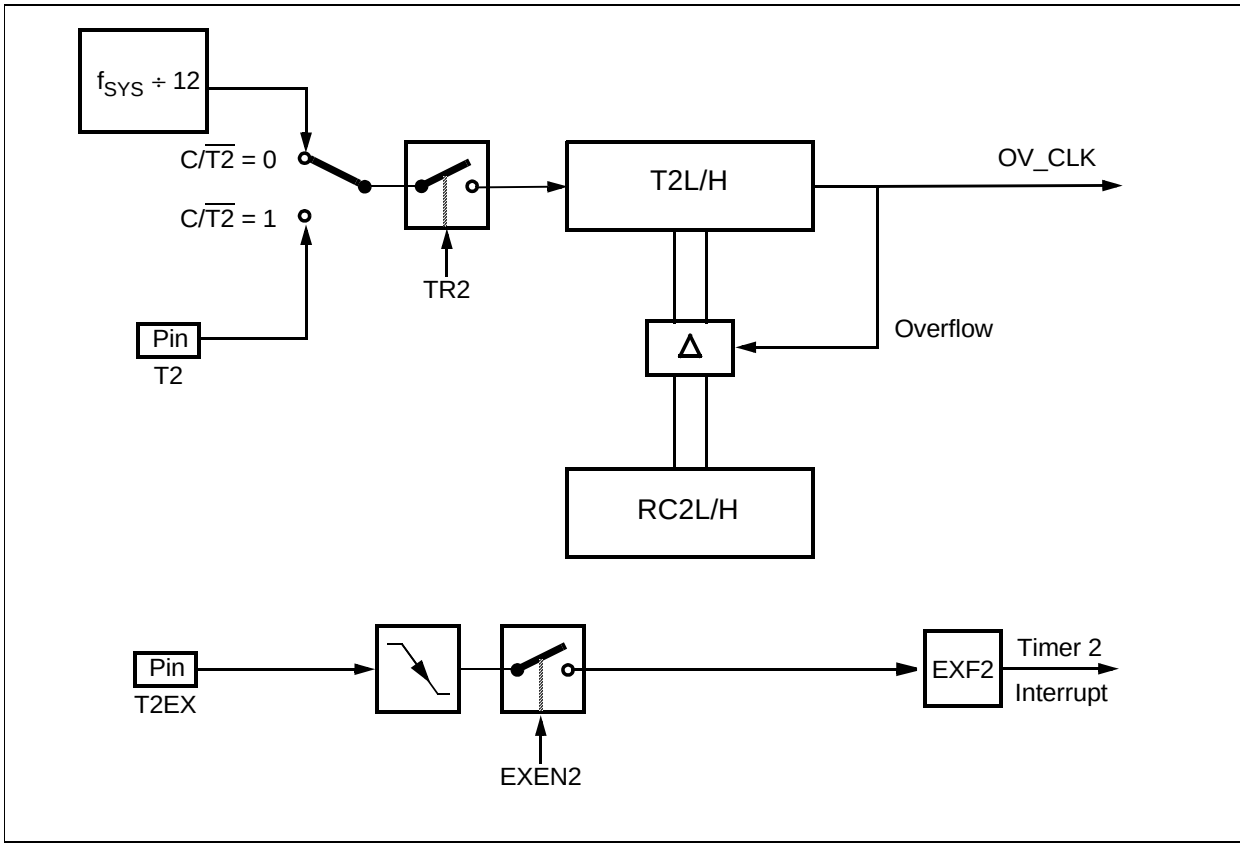


Figure 4-9 Baudrate Generator Mode

4.6.8 Count Clock

The count clock for the auto-reload mode is chosen by the bit C/$\overline{T2}$ in register T2CON. If C/$\overline{T2}$ = 0, a count clock of f_{SYS}/12 is used for the count operation.

If C/$\overline{T2}$ = 1, timer 2 behaves as a counter that counts 1-to-0 transitions of input pin T2. The counter samples pin T2 over 2 f_{SYS} cycles. If a 1 was detected during the first clock and a 0 was detected in the following clock, then the counter increments by one. Therefore, the input levels should be stable for at least 1 clocks.

4.6.9 Module Powerdown

The timer/counter 2 is disabled when the chip goes into the powerdown mode as describe in . Or it can be individually disabled by setting T2DIS in register PMCON1. This helps to reduce current consumption in the normal, slow down and idle modes of operation if the timer/counter 2 is not utilized. Bit T2ST in register PMCON2 reflects the powerdown status of timer/counter 2.

4.7 Capture/Compare Unit (CCU6)

The CCU6 provides two independent timers (T12, T13), which can be used for PWM generation, especially for AC-motor control. Additionally, special control modes for block commutation and multi-phase machines are supported.

Timer 12 Features

- Three capture/compare channels, each channel can be used either as capture or as compare channel.
- Generation of a three-phase PWM supported (six outputs, individual signals for highside and lowside switches)
- 16 bit resolution, maximum count frequency = system clock
- Dead-time control for each channel to avoid short-circuits in the power stage
- Concurrent update of the required T12/13 registers
- Center-aligned and edge-aligned PWM can be generated
- Single-shot mode supported
- Many interrupt request sources
- Hysteresis-like control mode

Timer 13 Features

- One independent compare channel with one output
- 16 bit resolution, maximum count frequency = system clock
- Can be synchronized to T12
- Interrupt generation at period-match and compare-match
- Single-shot mode supported

Additional Features

- Block commutation for Brushless DC-drives implemented
- Position detection via Hall-sensor pattern
- Automatic rotational speed measurement for block commutation
- Integrated error handling
- Fast emergency stop without CPU load via external signal ($\overline{\text{CTRAP}}$)
- Control modes for multi-channel AC-drives
- Output levels can be selected and adapted to the power stage
- Capture/compare unit can be powerdown in normal, idle and slow-down modes

The timer T12 can work in capture and/or compare mode for its three channels. The modes can also be combined. The timer T13 can work in compare mode only. The multi-channel control unit generates output patterns which can be modulated by T12 and/or T13. The modulation sources can be selected and combined (refer to figure *'Modulation Selection, Passive Level and Alternate Output Enable of T12')* for the signal modulation.

4.7.1 Timer T12

4.7.1.1 Overview

The timer T12 is used for capture/compare purposes with three independent channels. The timer T12 is a 16-bit wide counter. Three channel registers (CC60R, CC61R, CC62R), which are built with shadow registers (CC60SR, CC61SR, CC62SR), contain the compare value or the captured timer value. In compare mode, the software writes to the shadow registers and their contents are transferred simultaneously to the actual compare registers during the T12 shadow transfer. In capture mode, the captured value of T12 can be read from the channel registers. The period of the timer T12 is fixed by the period register T12PR, which is also built with a shadow register.

The write access from the CPU targets the corresponding shadow registers, whereas the read access targets the registers actually used (except for the three compare channels, where the actual and the shadow registers can be read).

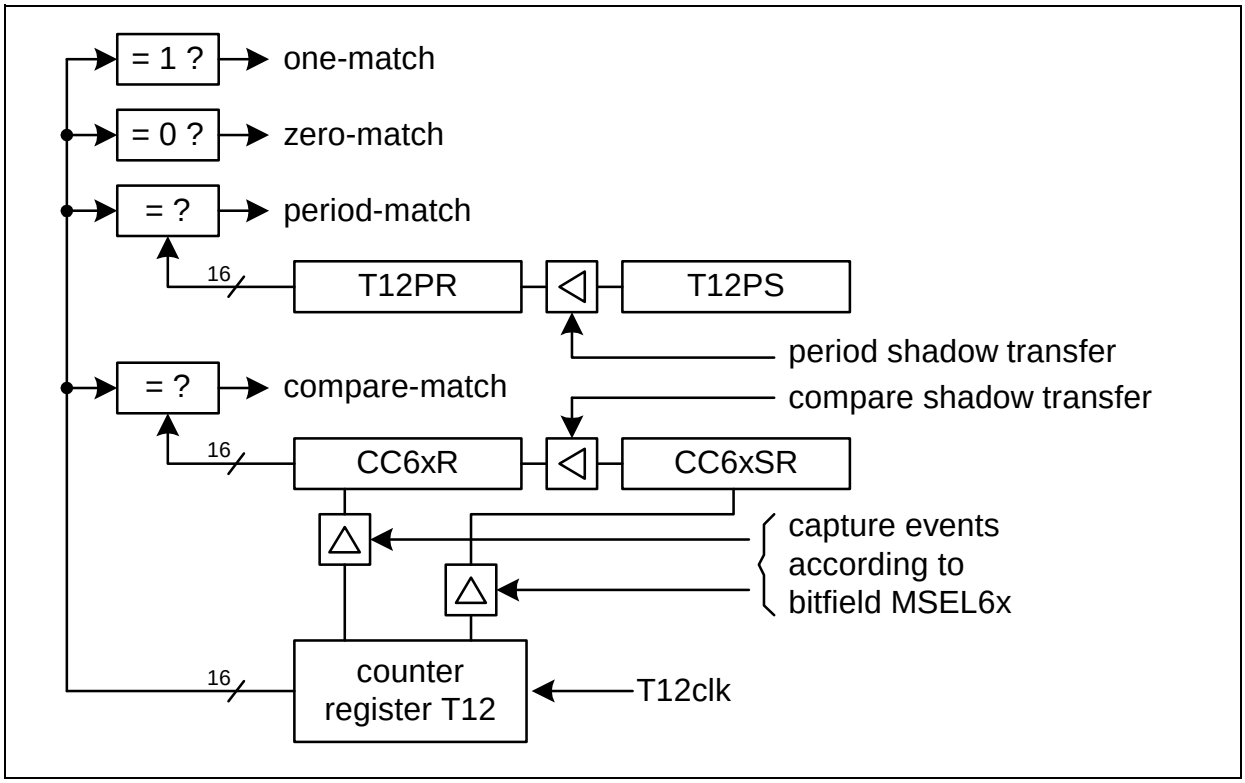


Figure 4-10 T12 Overview

While timer T12 is running, write accesses to register T12 are not taken into account. If the timer T12 is stopped and the dead-time counters are 0, write actions to register T12 are immediately taken into account.

4.7.1.2 Counting Rules

Referring to T12 input clock the counting sequence is defined by the following counting rules:

T12 in edge-aligned mode:

- The counter is reset to zero and (if desired) the T12 shadow transfer takes place if the period-match is detected. The counting direction is always upwards.

T12 in center-aligned mode:

- The count direction is set to counting up (CDIR=’0’) if the one-match is detected while counting down.
- The count direction is set to counting down (CDIR=’1’) if the period-match is detected while counting up.
- The counter counts up while CDIR=’0’ and it counts down while CDIR=’1’.
- If enabled the shadow transfer takes place:
- if the period-match is detected while counting up
- if the one-match is detected while counting down

The timer T12 prescaler is reset while T12 is not running to ensure reproducible timings and delays.

The counting rules lead to the following sequences:

.

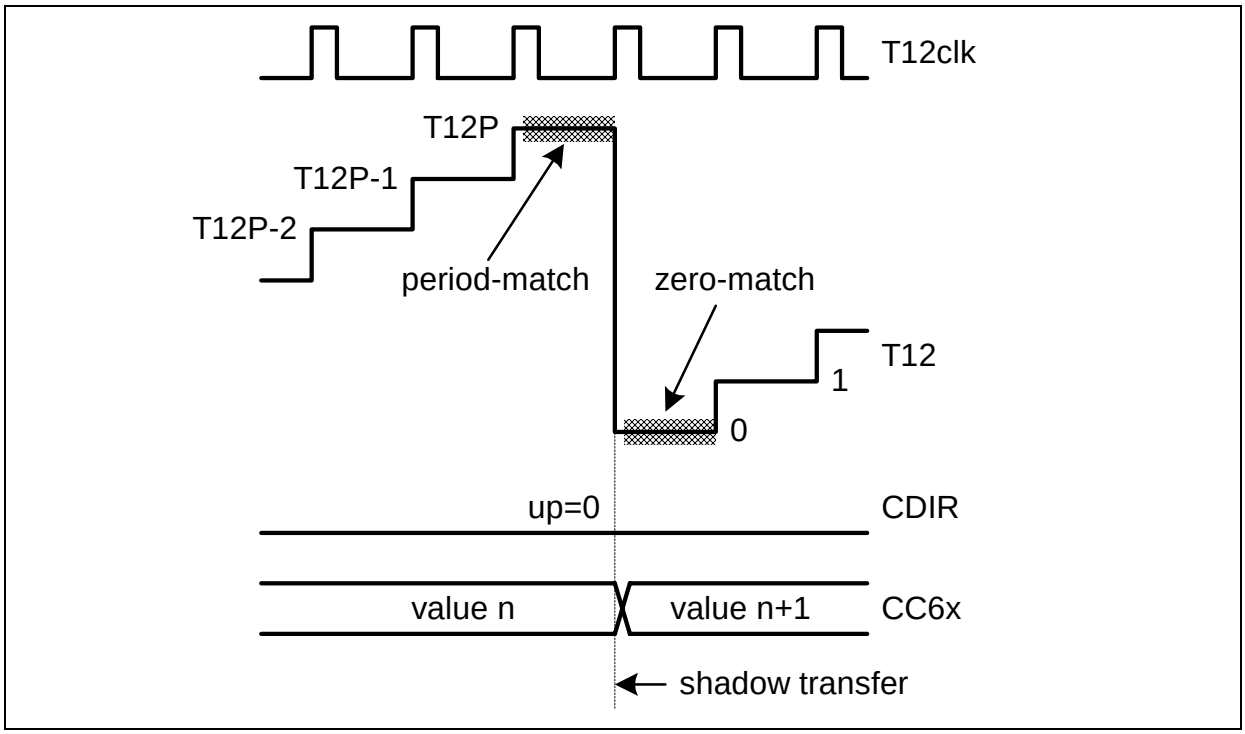


Figure 4-11 T12 in edge-aligned mode

In the center-aligned mode (T12 counts up and down), the counting rules lead to the following behavior:

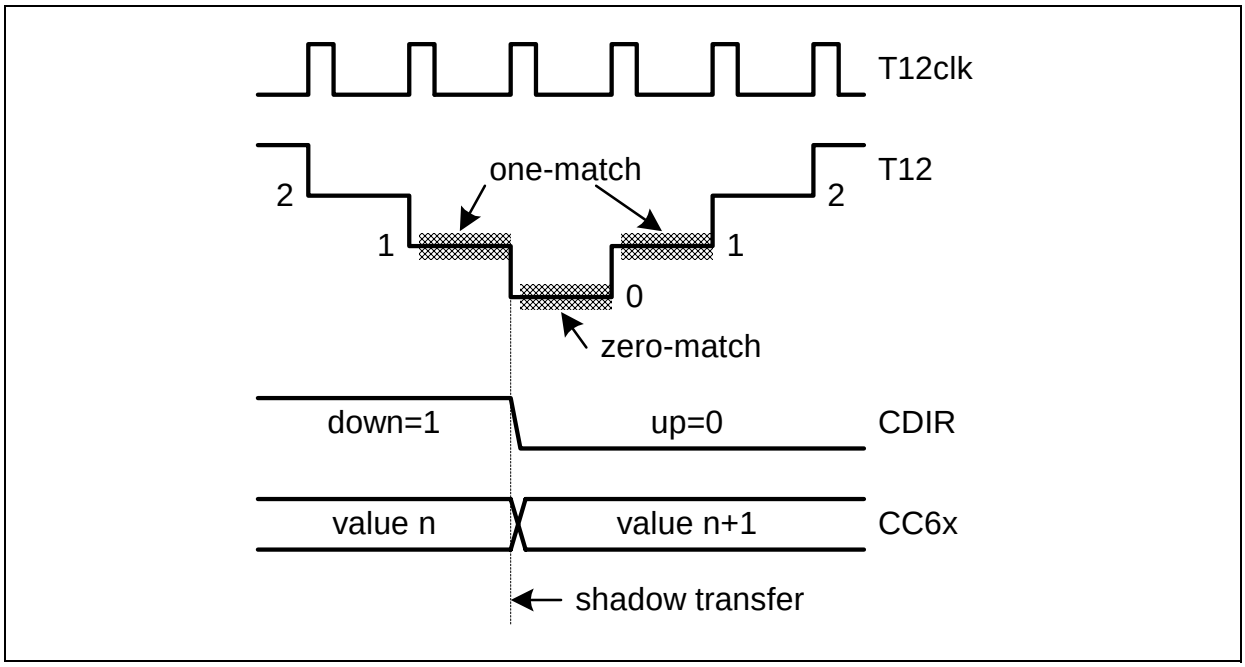


Figure 4-12 T12 in center-aligned mode, one-match detected

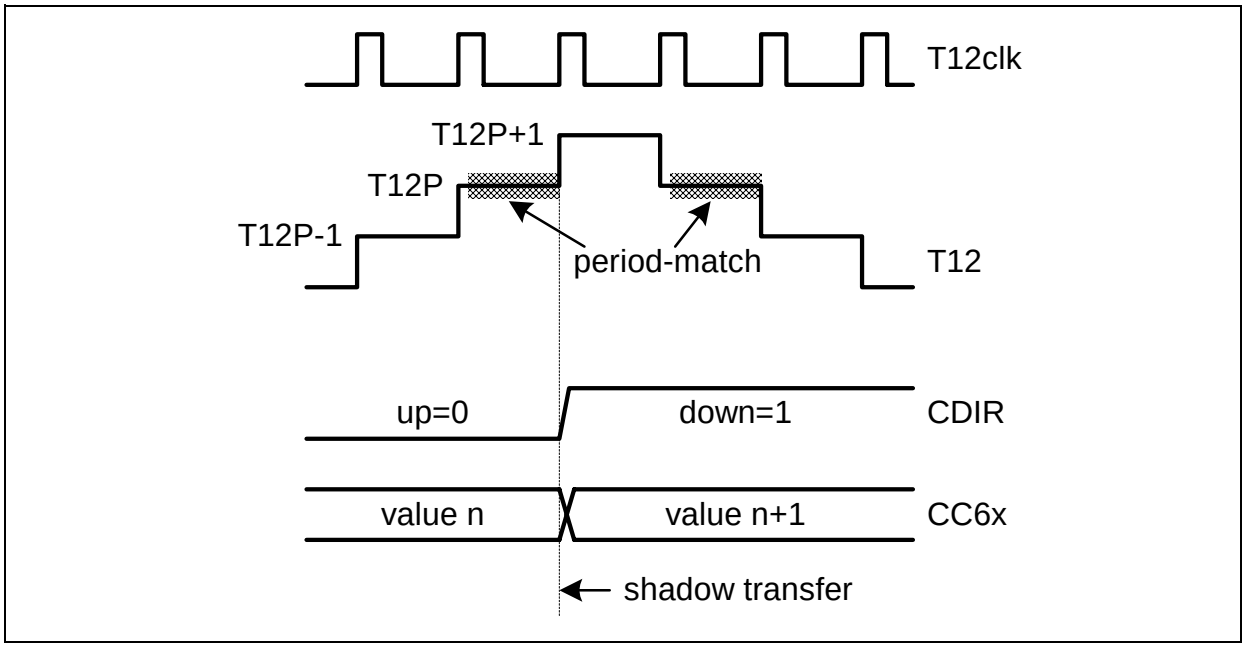


Figure 4-13 T12 in center-aligned mode, period-match detected

4.7.1.3 Switching Rules

The compare actions take place in parallel for the three compare channels. Depending on the count direction, the compare matches have different meanings. In order to get the PWM information independent from the output levels, two different states have been introduced for the compare actions: The active state and the passive state, which are used to generate the desired PWM as a combination of the states delivered by T13, the trap control unit and the multi-channel control unit. If the active state is interpreted as a ’1’ and the passive state as a ’0’, the state information is combined with a logical AND function.

- active AND active = active
- active AND passive = passive
- passive AND passive = passive

The compare states change with the detected compare-matches and are indicated by the CC6xST bits. The compare states of T12 are defined as follows:

- passive if the counter value is below the compare value
- active if the counter value is above the compare value

This leads to the following switching rules for the compare states:

- set to the active state when the counter value reaches the compare value while counting up
- reset to the passive state when the counter value reaches the compare value while counting down
- reset to the passive state in case of a zero-match without compare-match while counting up
- set to the active state in case of a zero-match with a parallel compare-match while counting up

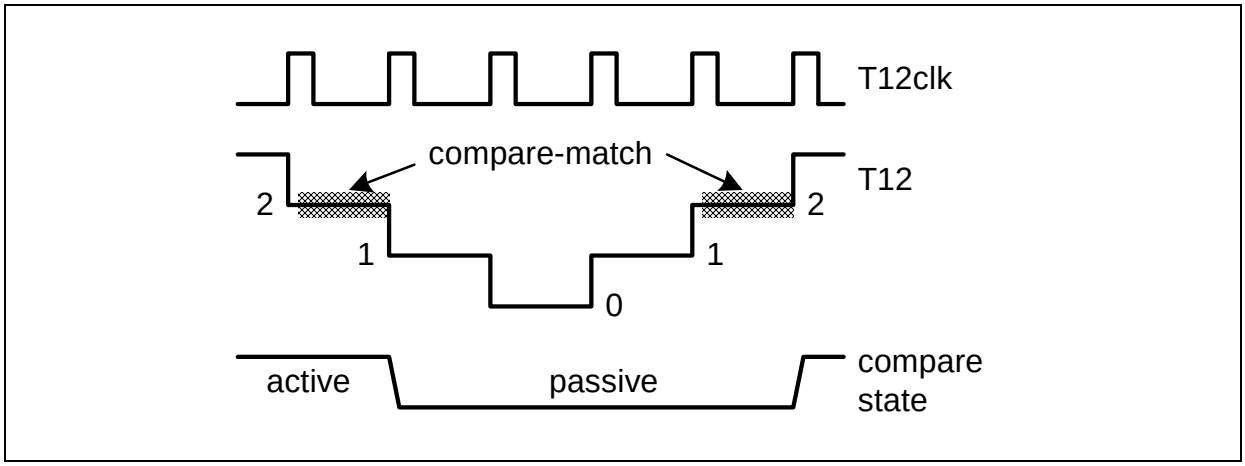


Figure 4-14 Compare States for Compare Value = 2

The switching rules are only taken into account while the timer is running. As a result, write actions to the timer registers while the timer is stopped do not lead to compare actions.

4.7.1.4 Duty Cycle of 0% and 100%

These counting and switching rules ensure a PWM functionality in the full range between 0% and 100% duty cycle (duty cycle = active time / total PWM period). In order to obtain a duty cycle of 0% (compare state never active), a compare value of T12P+1 has to be programmed (for both compare modes). A compare value of 0 will lead to a duty cycle of 100% (compare state always active).

4.7.1.5 Compare Mode of T12

The following figure shows the setting and resetting of the compare state bit CC6xST. In order to simplify the description, only one out of the three parallel channels is described. The letter ’x’ in the simplified bit names and signal names indicates that there are more than one channel. The CC6xST bit is the compare state bit in register CMPSTAT, the bit CC6xPS represents passive state select bit.
The timer T12 generates pulses indicating events like compare-matches, period-matches and zero-matches, which are used to set (signal T12_xST_se) and to reset (signal T12_xST_re) the corresponding compare state bit (CC6xST) according to the counting direction.
The timer T12 modulation output lines T12xO (two for each channel) can be selected to be in the active state while the corresponding compare state is ’0’ (with CC6xPS=’0’) or while the corresponding compare state is ’1’ (with CC6xPS=’1’). The bit COUT6xPS has the same effect for the second output of the channel. The example is shown without dead-time.

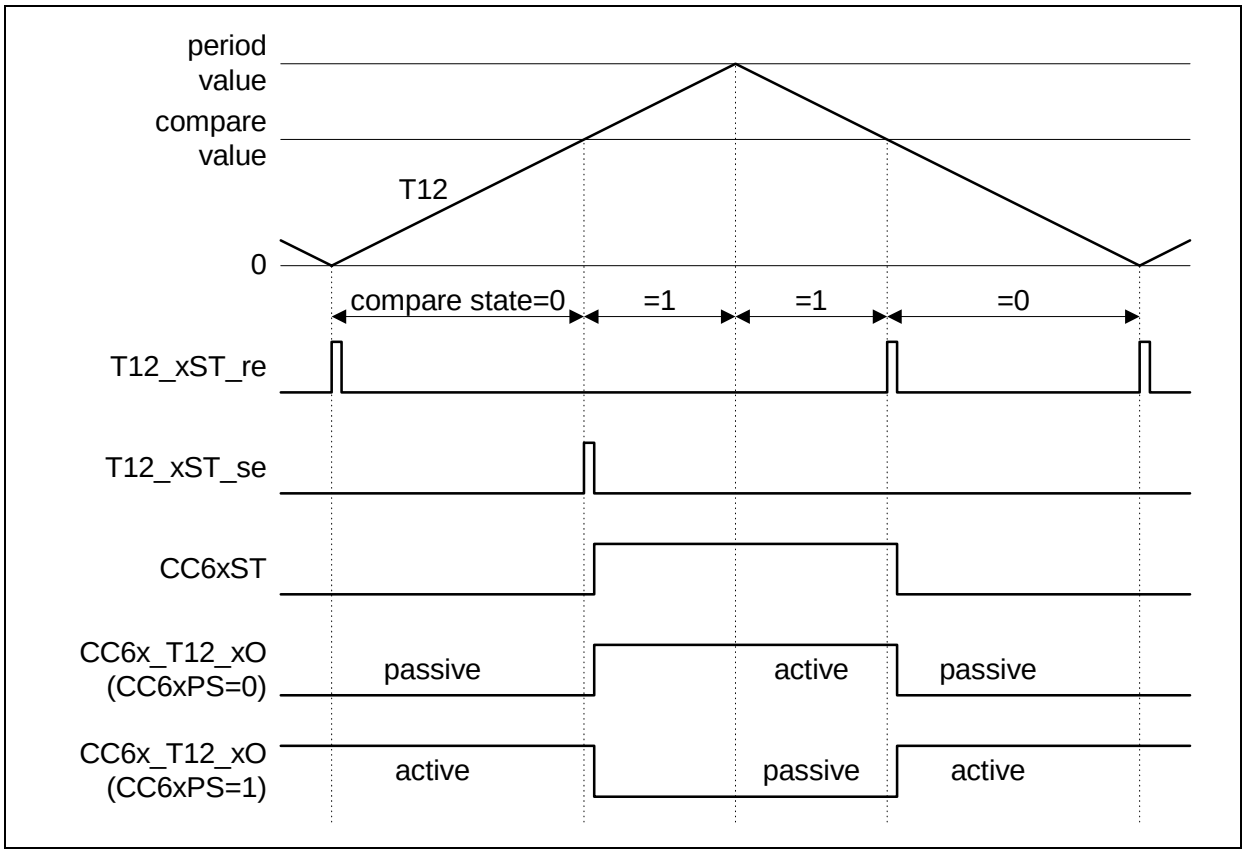


Figure 4-15 Compare States of Timer T12

According to the desired capture/compare mode, the compare state bits have to be switched. Therefore, an additional logic (see **Figure 4-16**) selects how and by which event the compare state bits are modified. The mode selection (by bitfields MSEL6x in register T12MSEL) enables the setting and the resetting of the compare state bits due to compare actions of timer T12.

The HW modifications of the compare state bits is only possible while the timer T12 is running. Therefore, the bit T12R is used to enable/disable the modification by HW.

For the hysteresis-like compare mode (MSEL6x='1001'), the setting of the compare state bit is only possible while the corresponding input CCPOSx='1' (inactive).

If the Hall Sensor mode (MSEL6x='1000') is selected, the compare state bits of the compare channels 1 and 2 are modified by the timer T12 in order to indicate that a programmed time has elapsed.

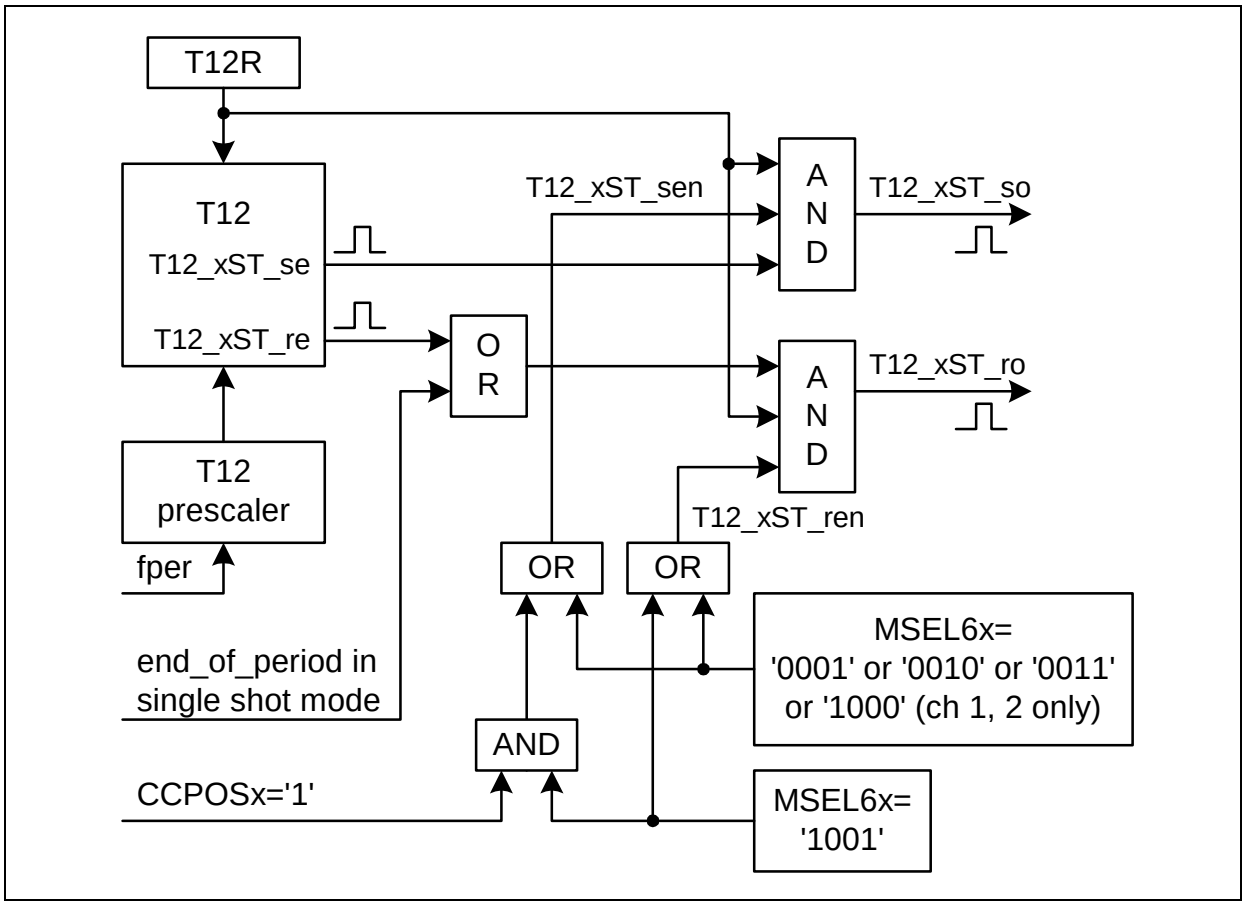


Figure 4-16 T12 Compare Logic

The T12 compare output lines T12_xST_so (to set bit CC6xST) and T12_xST_ro (to reset bit CC6xST) are also used to trigger the corresponding interrupt flags and to generate interrupts. The signal T12_xST_so indicates the interrupt event for the rising edge (ICC6xR), whereas the signal T12_xST_ro indicates the falling edge event (ICC6xF) in compare mode.

The compare state bits indicate the occurrence of a capture or compare event of the corresponding channel. It can be set (if it is ’0’) by the following events:

- upon a software set (CC6xS)
- upon a compare set event (see switching rules) if the T12 runs and if the T12 set event is enabled
- upon a capture set event

The bit CC6xST can be reset (if it is ’1’) by the following events:

- upon a software reset (CC6xR)
- upon a compare reset event (see switching rules) if the T12 runs and if the T12 reset event is enabled (including in single shot mode the end of the T12 period)
- upon a reset event in the hysteresis-like control mode

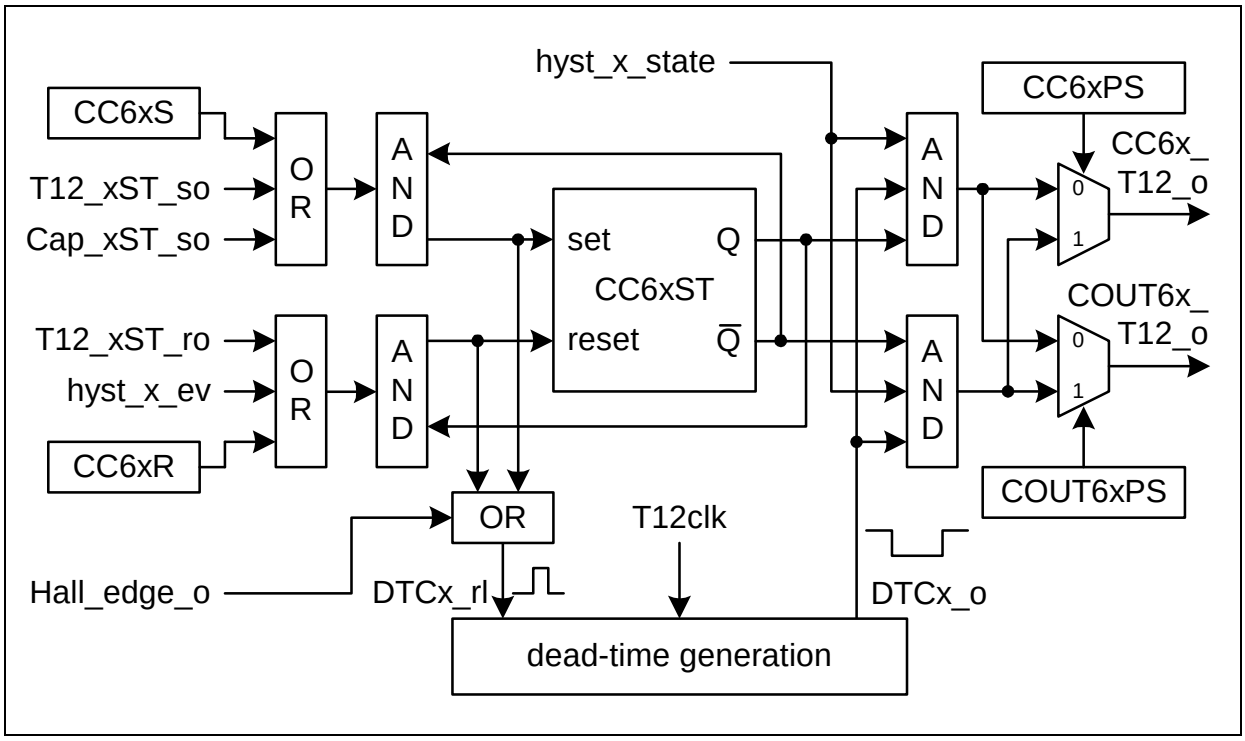


Figure 4-17 T12 Logic for CC6xST Control

The events triggering the set and reset action of the CC6xST bits have to be combined, see **Figure 4-17**. The occurrence of the selected capture event (signal Cap_xST_so) or the setting of CC6xS in register CMPMODIF also leads to a set action of bit CC6xST, whereas the negative edge at pin CCPOSx (in hysteresis-like mode, signal hyst_x_ev) or the setting of bit CC6xR leads to reset action.
The set signal is only generated while bit CC6xST is reset, a reset can only take place while the bit is set. This permits the OR-combination of the resulting set and reset signals to one common signal (DTCx_rl) triggering the reload of the dead-time counter. It is only triggered if bit CC6xST is changed, permitting a correct PWM generation with dead-time and the complete duty cycle range of 0% to 100% in edge-aligned and in center-aligned mode.

In the case that the dead-time generation is enabled, the change of bit CC6xST triggers the dead-time unit and a signal DTCx_o is generated. The length of the ’0’ level of this signal corresponds to the desired dead-time, which is used to delay the rising edge (passive to active edge) of the output signal.
In order to generate independent PWM patterns for the highside and the lowside switches of the power inverter, the interval when a PWM signal should be active can be selected by the bits CC6xPS. They select if the PWM signal is active while the compare state bit is ’0’ (T12 counter value below the compare value) or while it is ’1’ (T12 counter value above the compare value).

In **Figure 4-17**, the signals CC6x_T12_o and COUT6x_T12_o are inputs to the modulation control block, where they can be combined with other PWM signals.

4.7.1.6 Switching Examples in edge-aligned Mode

The following figure shows two switching examples in edge-aligned mode with duty cycles near to 0% and near to 100%. The compare-, period- or zero-matches lead to modifications of the compare state and the shadow transfer (if requested by STE12=’1’) in the next clock cycle.

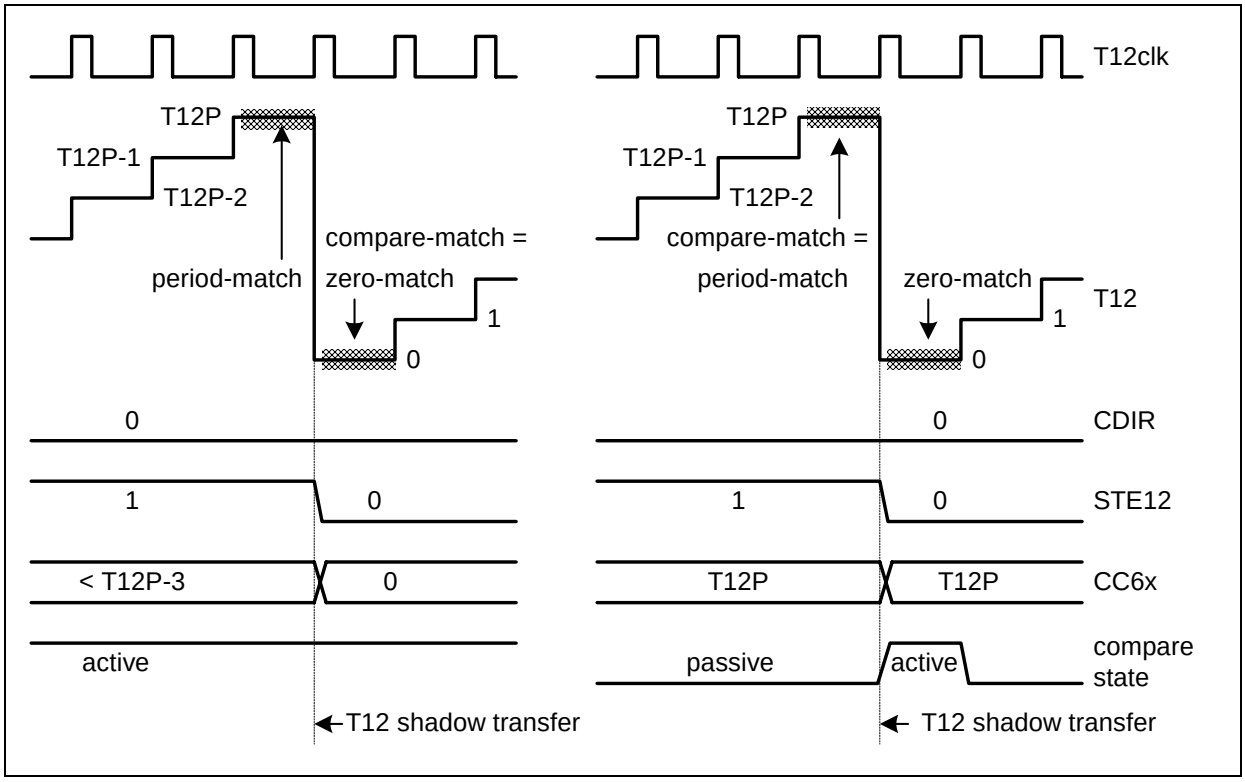


Figure 4-18 Switching Examples in edge-aligned Mode

4.7.1.7 Switching Examples in center-aligned Mode

The following figures show examples of the switching of the compare state and the T12 shadow transfer according to the programmed compare values.

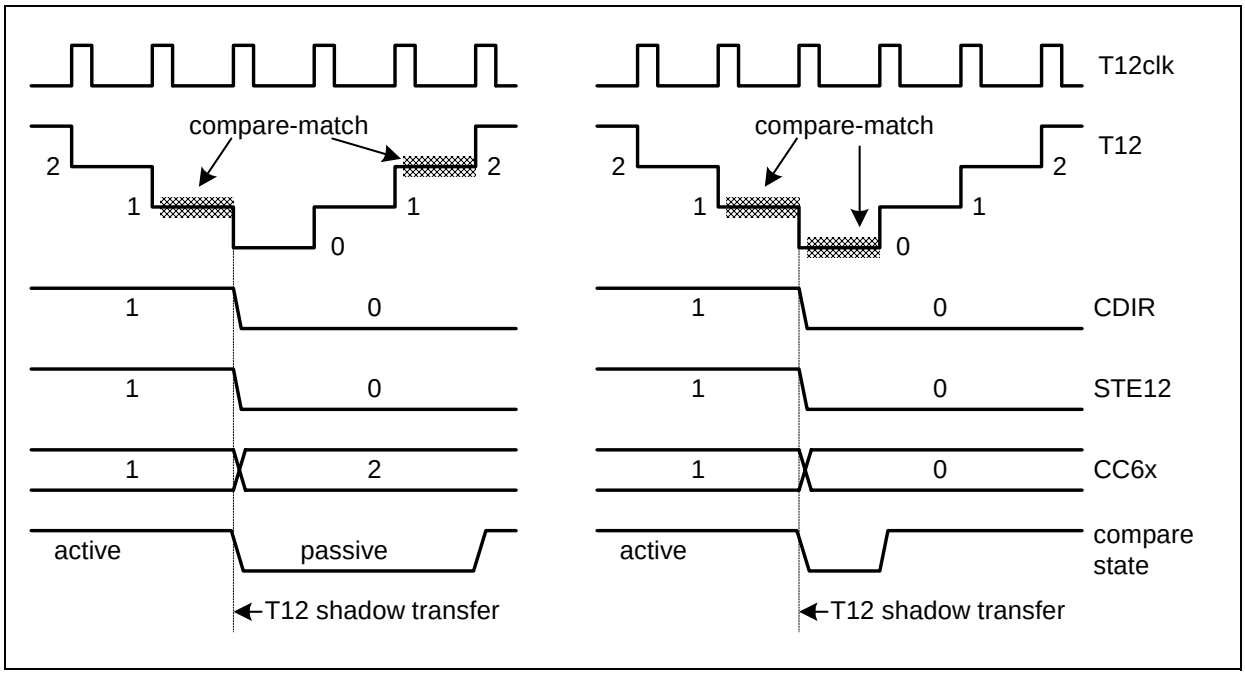


Figure 4-19 Switching Example for Duty Cycles near to 100%

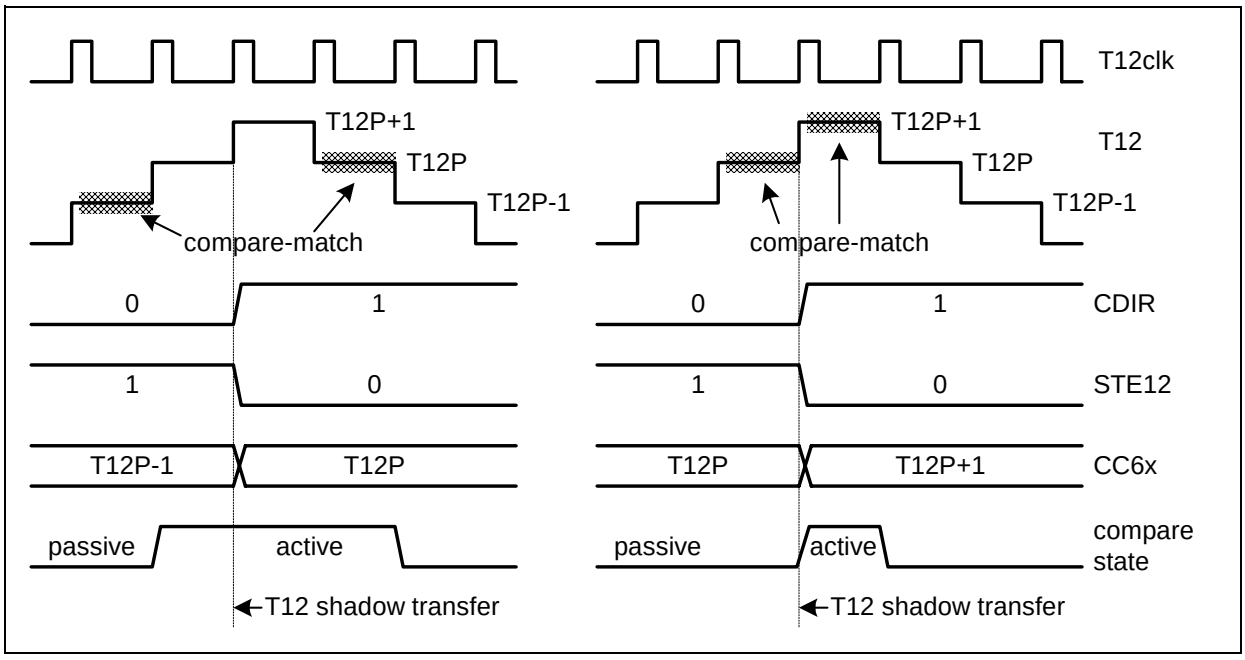


Figure 4-20 Switching Example for Duty Cycles near to 0%

4.7.1.8 Dead-time Generation

The generation of (complementary) signals for the highside and the lowside switches of one power inverter phase is based on the same compare channel. For example, if the

highside switch should be active while the T12 counter value is above the compare value (compare state =’1’), then the lowside switch should be active while the counter value is below (compare state =’0’). The compare state, which may lead to an active output (respecting other modulation sources and the trap functionality) can be selected by the CC6xPS bits.

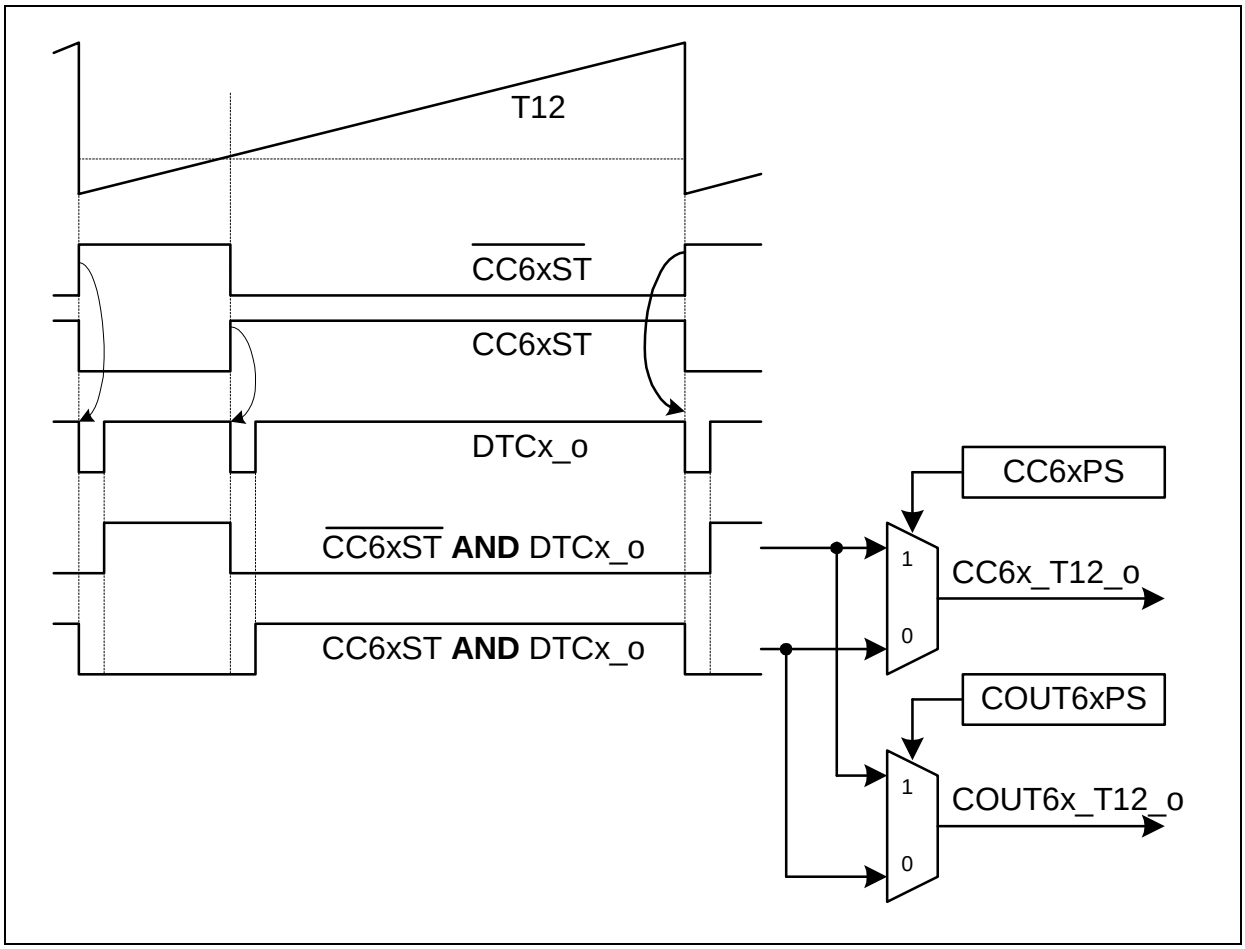


Figure 4-21 PWM-signals with Dead-time Generation

In most cases, the switching behavior of the connected power switches is not symmetrical concerning the times needed to switch on and to switch off. A general problem arises if the time to switch on is smaller than the time to switch off the power device. In this case, a short-circuit in the inverter bridge leg occurs, which may damage the complete system. In order to solve this problem by HW, this capture/compare unit contains a programmable dead-time counter, which delays the passive to active edge of the switching signals (the active to passive edge is not delayed), see **Figure 4-21**.

The dead-time generation logic (see **Figure 4-22**) is built in a similar way for all three channels of T12. Each change of the CC6xST bits triggers the corresponding dead-time counter (6 bit down counter, clocked with T12clk). The trigger pulse (DTCx_rl) leads to a reload of the dead-time counter with the value, which has been programmed in bit field

DTM. This reload can only take place if the dead-time feature is enabled by bit DTEx and while the counter is zero.
While counting down (zero is not yet reached), the output line DTCx_o becomes ’0’. This output line is combined with the T12 modulation signals, leading to a delay of the passive to active edge of the resulting signal, which is shown in **Figure 4-21**. When reaching the counter value zero, the dead-time counter stops counting and the signal DTCx_o becomes ’1’. The dead-time counter can not be reloaded while it is counting.

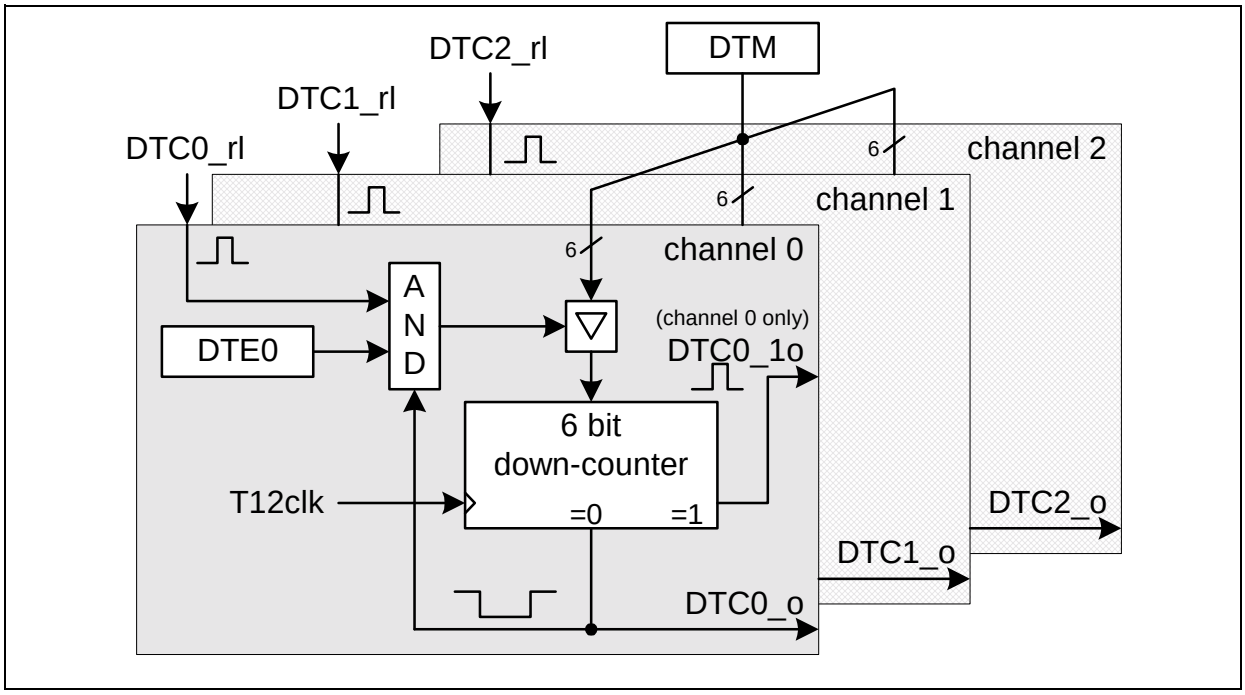


Figure 4-22 Dead-time Counter

Each of the three channels works independently with its own dead-time counter and the trigger and enable signals. The value of bit field DTM is valid for all of the three channels.

In the Hall sensor mode, timer T12 is used to measure the rotational speed of the motor (channel 0 in capture mode) and to control the phase delay before switching to the next state (channel 1 in compare mode). Furthermore, channel 2 can be used to generate a time-out signal (in compare mode). As a result, T12 can not be used for modulation and, due to the block commutation patterns, a dead-time generation is not required. In order to built an efficient noise filter for the Hall signals, channel 0 of the dead-time unit is triggered (reloaded) with each detected edge of the Hall signals, see signal Hall_edge_o in **Figure 4-17**. For this feature, channel 0 also generates a pulse if its counter value is one.

4.7.1.9 Capture Mode

In capture mode the bits CC6xST indicate the occurrence of the selected capture event according to the bit fields MSEL6x. A rising and/or a falling edge on the pins CC6x can

be selected as capture event, that is used to transfer the contents of timer T12 to the CC6xR and CC6xSR registers. In order to work in capture mode, the capture pins have to be configured as inputs.

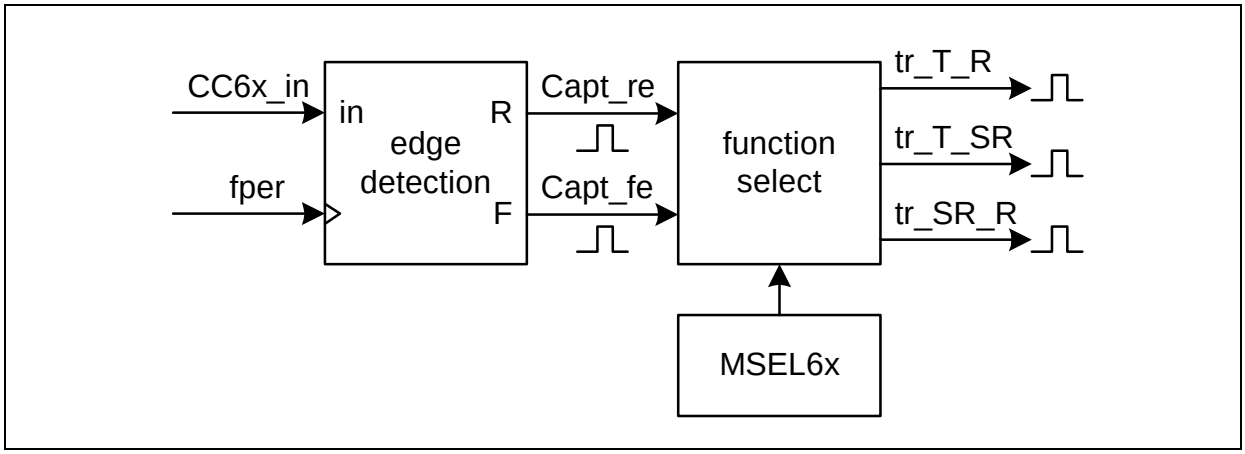


Figure 4-23 Capture Logic

The block diagram of the capture logic for one channel is shown in **Figure 4-23**. This logic is identical for all three independent channels of timer T12. The input signal (CC6x_in) from the input pin CC6x is connected to an edge detection logic delivering two output signals, one for the rising edge (Capt_re) and one for the falling edge (Capt_fe). These signals are also used as trigger sources for the channel interrupts if capture mode is selected.

There are several possibilities to store the captured values in the registers. In double register capture mode the timer value is stored in the channel shadow register CC6xSR. The value formerly stored in this register is simultaneously copied to the channel register CC6xR.This mode can be used if two capture events occur with very few time between them. The SW can then check the new captured value and has still the possibility to read the value captured before.

The selection of the capture mode is done by bitfield MSEL6x. According to the selected mode and the detected capture event, the signals tr_T_R (transfer T12 contents to register CC6xR), tr_T_SR (transfer T12 contents to register CC6xSR) or tr_SR_R (transfer contents of CC6xSR to register CC6xR) are activated.

Note: In capture mode, a shadow transfer can be requested according to the shadow transfer rules, except for the capture / compare registers, that are left unchanged.

4.7.1.10 Single Shot Mode

In single shot mode, the timer T12 stops automatically at the end of the its counting period. **Figure 4-24** shows the functionality at the end of the timer period in edge-aligned

and in center-aligned mode. If the end of period event is detected while bit T12SSC is set, the bits T12R and all CC6xST bits are reset.

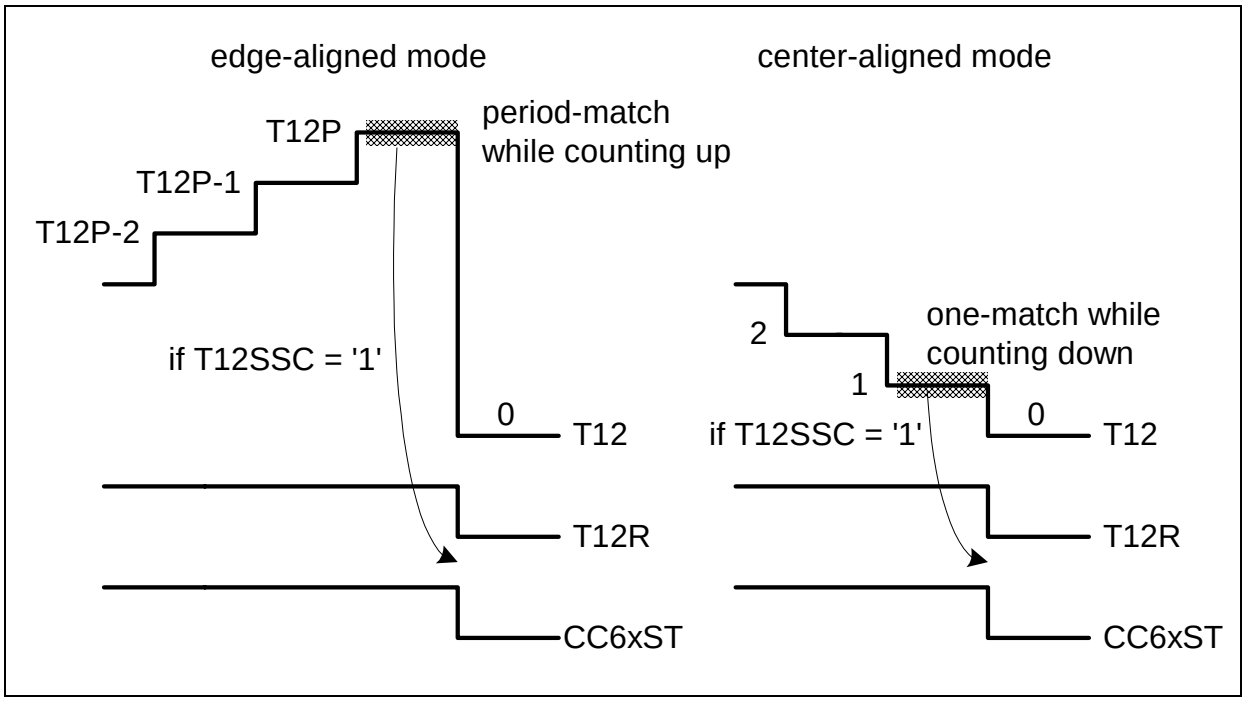


Figure 4-24 End of Single Shot Mode of T12

4.7.1.11 Hysteresis-Like Control Mode

The hysteresis-like control mode (MSEL6x = ’1001’) offers the possibility to switch off the PWM output if the input CCPOSx becomes ’0’ by resetting bit CC6xST. This can be used as a simple motor control feature by using a comparator indicating e.g. over current. While CCPOSx=’0’, the PWM outputs of the corresponding channel are driving their passive levels. The setting of bit CC6xST is only possible while CCPOSx=’1’.

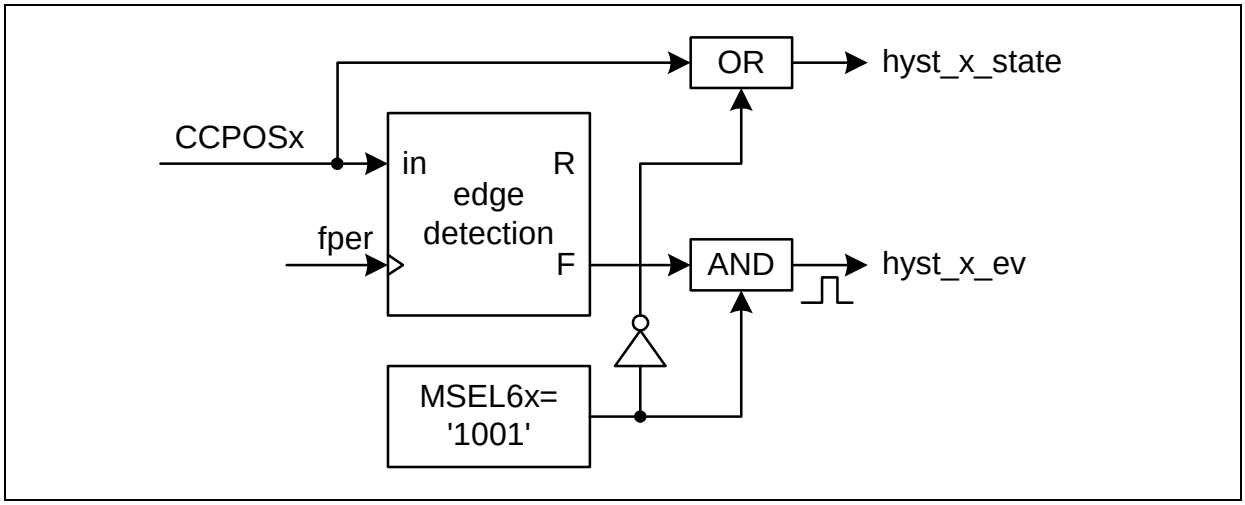


Figure 4-25 Hysteresis-Like Control Mode Logic

4.7.2 Timer T13

4.7.2.1 Overview

The timer T13 is built similar to T12, but only with one channel in compare mode. The counter can only count up (similar to the edge-aligned mode of T12). The T13 shadow transfer in case of a period-match is enabled by bit STE13 in register TCTR0. During the T13 shadow transfer, the contents of register CC63SR is transferred to register CC63R. Both registers can be read by SW, whereas only the shadow register can be written by SW.
The bits CC63PS, T13IM and PSL63 have shadow bits. The contents of the shadow bits is transferred to the actually used bits during the T13 shadow transfer. Write actions target the shadow bits, read actions deliver the value of the actually used bit.

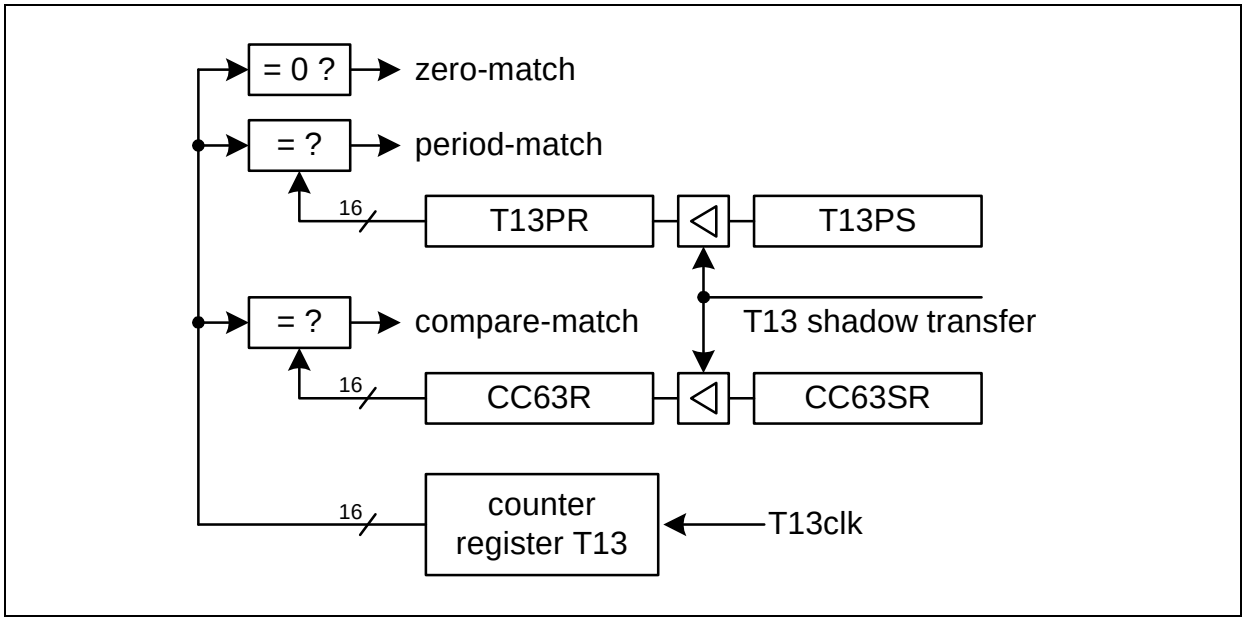


Figure 4-26 T13 Overview

Timer T13 counts according to the same counting and switching rules as timer T12 in edge-aligned mode.

4.7.2.2 Compare Mode

The compare structure of T13 is based on the compare signals T13_ST_se (compare - match detected) and T13_ST_re (zero-match detected without compare-match). These compare signals may modify bit CC63ST only while the timer is running (T13R=’1’).

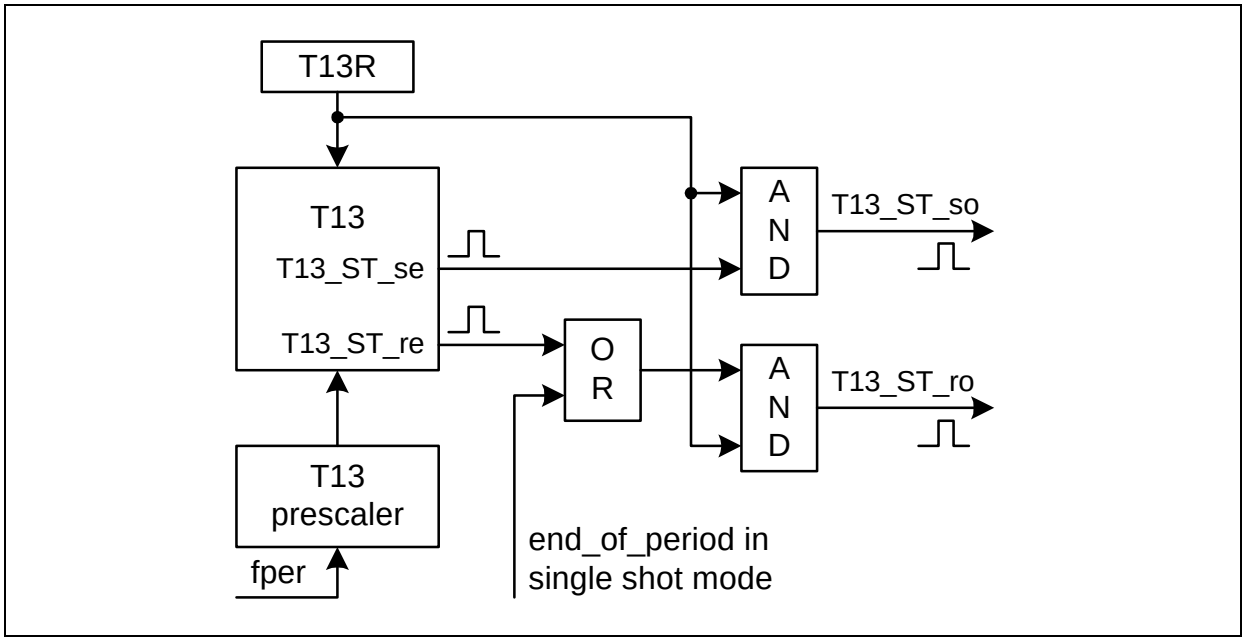


Figure 4-27 T13 Compare Logic

Similar to T12, bit CC63ST can be modified by SW by bits CC63S and CC63R. The output line COUT63_T13_o can generate a T13 PWM at the output pin COUT63. The signal MOD_T13_o can be used to modulate the other output signals with a T13 PWM. In order to decouple COUT63 from the internal modulation, the compare state leading to an active signal can be selected independently by bits T13IM and COUT63PS.

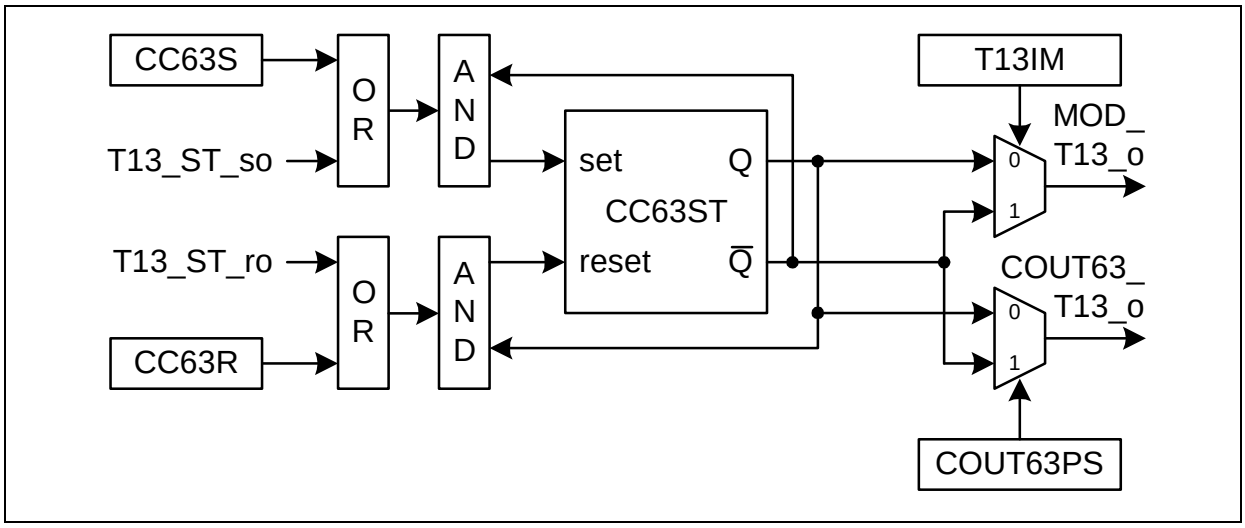


Figure 4-28 T13 Logic for CC6xST Control

4.7.2.3 Single Shot Mode

The single shot mode of T13 is similar to the single shot mode of T12 in edge-aligned mode.

4.7.2.4 Synchronization of T13 to T12

The timer T13 can be synchronized on a T12 event. The bit fields T13TEC and T13TED select the event, which is used to start timer T13. This event sets bit T13R per HW and T13 starts counting. Combined with the single shot mode, this feature can be used to generate a programmable delay after a T12 event.

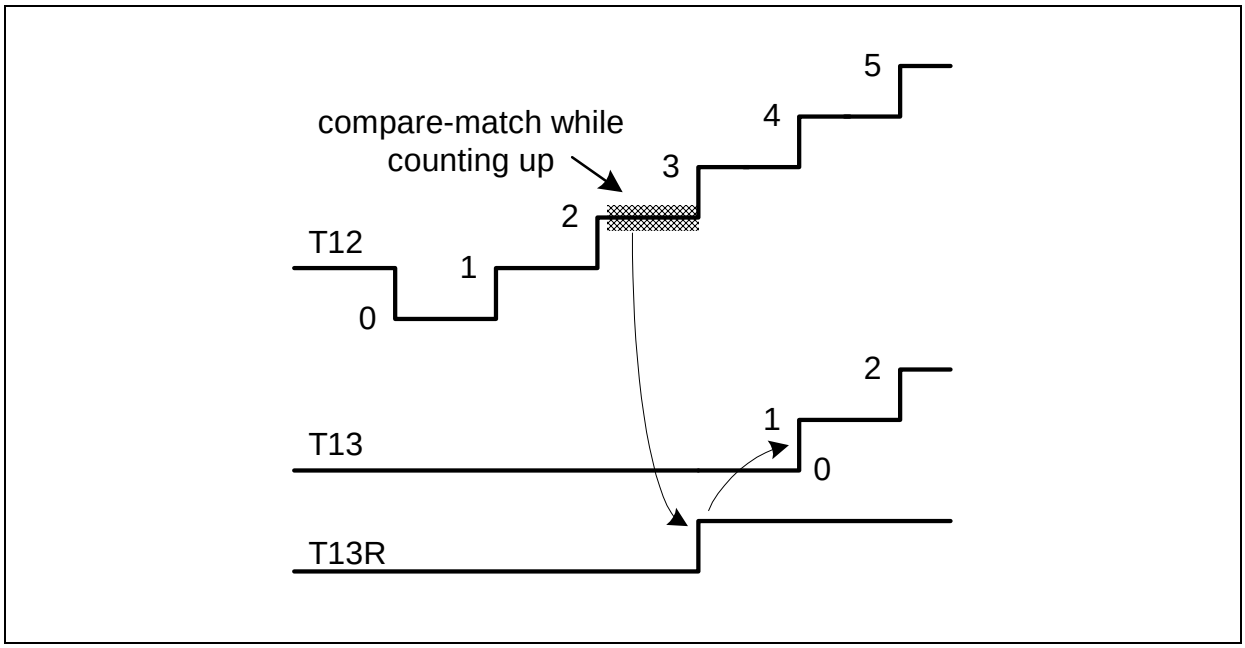


Figure 4-29 Synchronization of T13 to T12

This figure shows the synchronization of T13 to a T12 event. The selected event in this example is a compare-match (compare value = 2) while counting up. The clocks of T12 and T13 can be different (other prescaler factor), but for reasons of simplicity, this example shows the case for T12clk equal to T13clk.

4.7.3 Multi-channel Mode

The multi-channel mode offers a possibility to modulate all six T12-related output signals within one instruction. The bits in bit field MCMP are used to select the outputs that may become active. If the multi-channel mode is enabled (bit MCMEN='1'), only those outputs may become active, which have a '1' at the corresponding bit position in bit field MCMP.

This bit field has its own shadow bit field MCMPS, which can be written by SW. The transfer of the new value in MCMPS to the bit field MCMP can be triggered by and synchronized to T12 or T13 events. This structure permits the SW to write the new value, which is then taken into account by the HW at a well-defined moment and synchronized to a PWM period. This avoids unintended pulses due to unsynchronized modulation sources (T12, T13, SW).

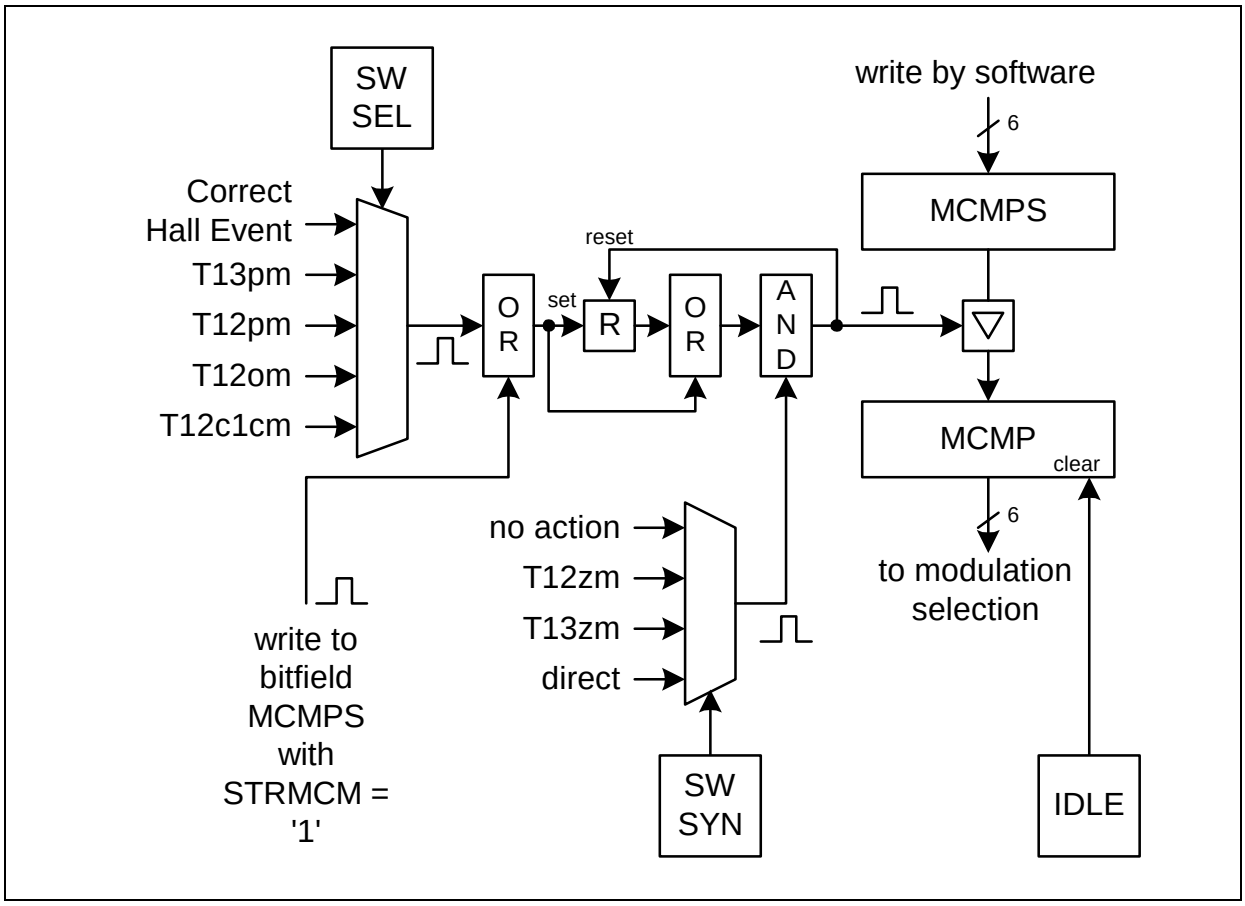


Figure 4-30 Modulation Selection and Synchronization

Figure 4-30 shows the modulation selection for the multi-channel mode. The event that triggers the update of bit field MCMP is chosen by SWSEL. If the selected switching event occurs, the reminder flag R is set. This flag monitors the update request and it is automatically reset when the update takes place. In order to synchronize the update of MCMP to a PWM generated by T12 or T13, bit field SWSYN allows the selection of the synchronization event, which leads to the transfer from MCMPS to MCMP. Due to this structure, an update takes place with a new PWM period.
If it is explicitly desired, the update takes place immediately with the setting of flag R when the direct synchronization mode is selected. The update can also be requested by SW by writing to bit field MCMPS with the shadow transfer request bit STRMCM set. If this bit is set during the write action to the register, the flag R is automatically set. By using the direct mode and bit STRMCM, the update takes place completely under SW control.

The possible HW request events are:

- a T12 period-match while counting up (T12pm)
- a T12 one-match while counting down (T12om)
- a T13 period-match (T13pm)

- a T12 compare-match of channel 1 (T12c1cm)
- a correct Hall event

The possible HW synchronization events are:

- a T12 zero-match while counting up (T12zm)
- a T13 zero-match (T13zm)
-

4.7.4 Trap Handling

The trap functionality permits the PWM outputs to react on the state of the input pin CTRAP. This functionality can be used to switch off the power devices if the trap input becomes active (e.g. as emergency stop).
During the trap state, the selected outputs are forced to the passive state and no active modulation is possible. The trap state is entered immediately by HW if the CTRAP input signal becomes active and the trap function is enabled by bit TRPPEN. It can also be entered by SW by setting bit TRPF (trap input flag), leading to TRPS='1' (trap state indication flag). The trap state can be left when the input is inactive, by SW control and synchronized to the following events:

- TRPF is automatically reset after CTRAP becomes inactive (if TRPM2='0')
- TRPF has to be reset by SW after CTRAP becomes inactive (if TRPM2='1')
- synchronized to T12 PWM after TRPF is reset
  (T12 period-match in edge-aligned mode or one-match while counting down in center-aligned mode)
- synchronized to T13 PWM after TRPF is reset
  (T13 period-match)
- no synchronization to T12 or T13

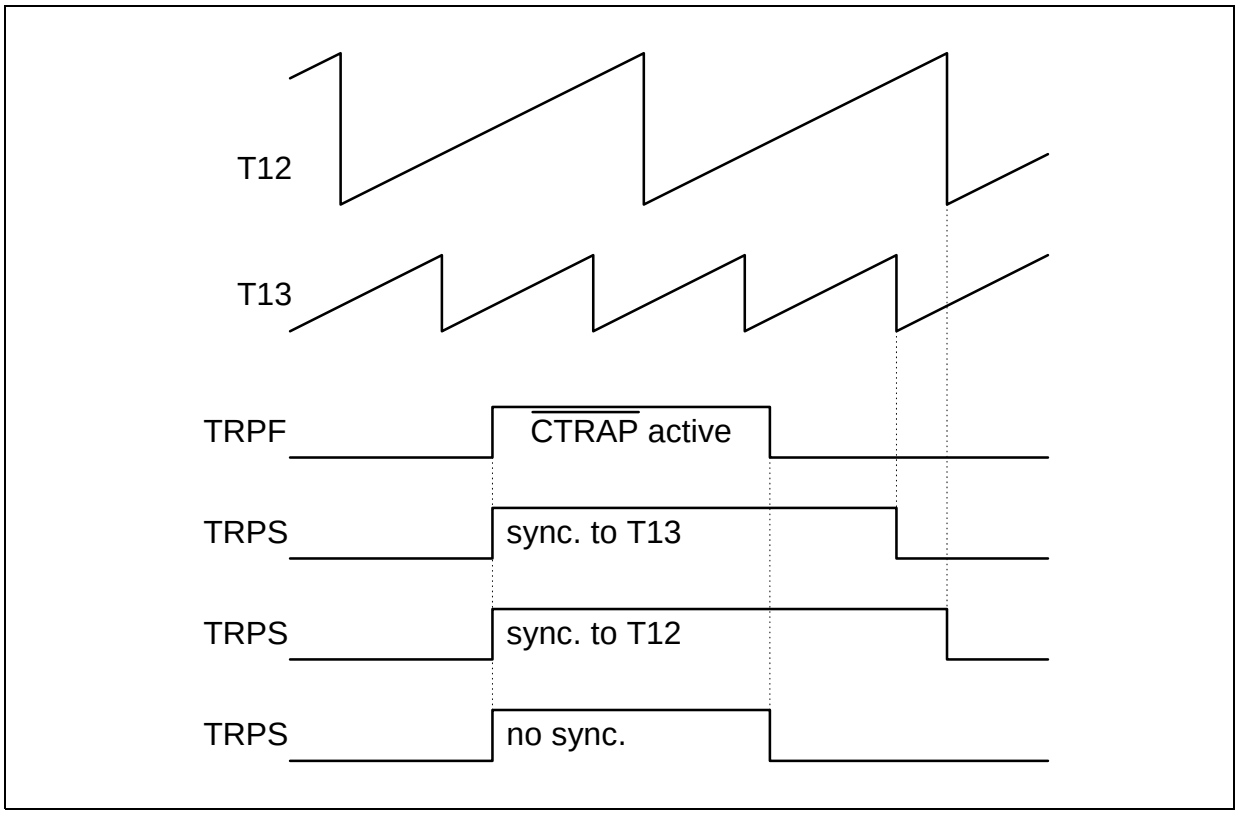


Figure 4-31 Trap State Synchronization (with TRM2=’0’)

4.7.5 Modulation Control

The modulation control part combines the different modulation sources (CC6x_T12_o, COUT6x_T12_o = six T12-related signals from the three compare channels), the T13-related signal (MOD_T13_o) and the multi-channel modulation signals (MCMP bits). each modulation source can be individually enabled for each output line. Furthermore, the trap functionality is taken into account to disable the modulation of the corresponding output line during the trap state (if enabled).

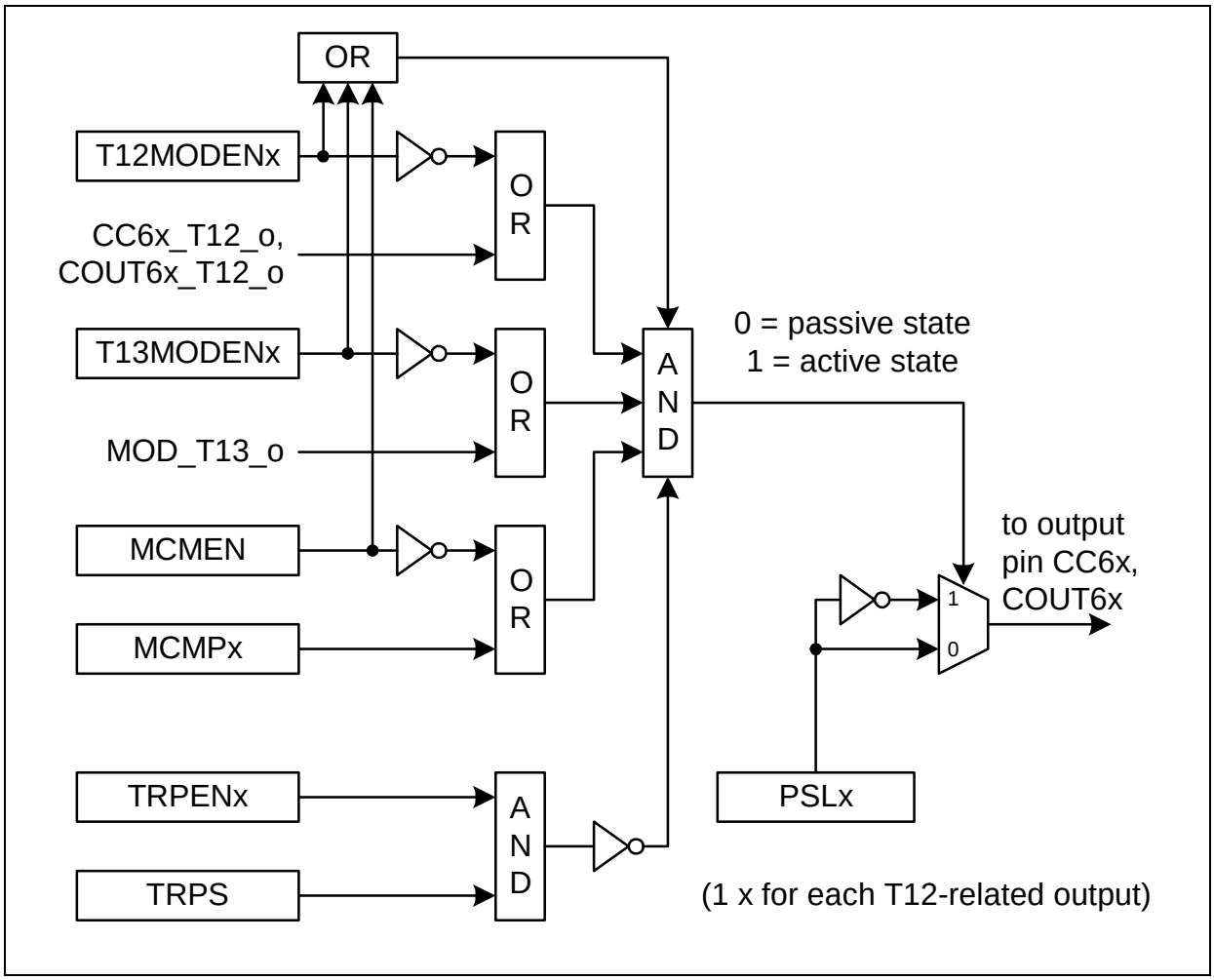


Figure 4-32 Modulation Control of T12-related Outputs

The logic shown in **Figure 4-32** has to be built separately for each of the six T12-related output lines, referring to the index ’x’ in the figure above.

The output level, that is driven while the output is in the passive state is defined by the corresponding bit in bit field PSL. If the resulting modulation signal is active, the inverted level of the PLSx bit is driven by the output stage.

The modulation control part for the T13-related output COUT63 combines the T13 output signal (COUT63_T13_o) and the enable bit ECT13O with the trap functionality. The output level of the passive state is selected by bit PSL63.

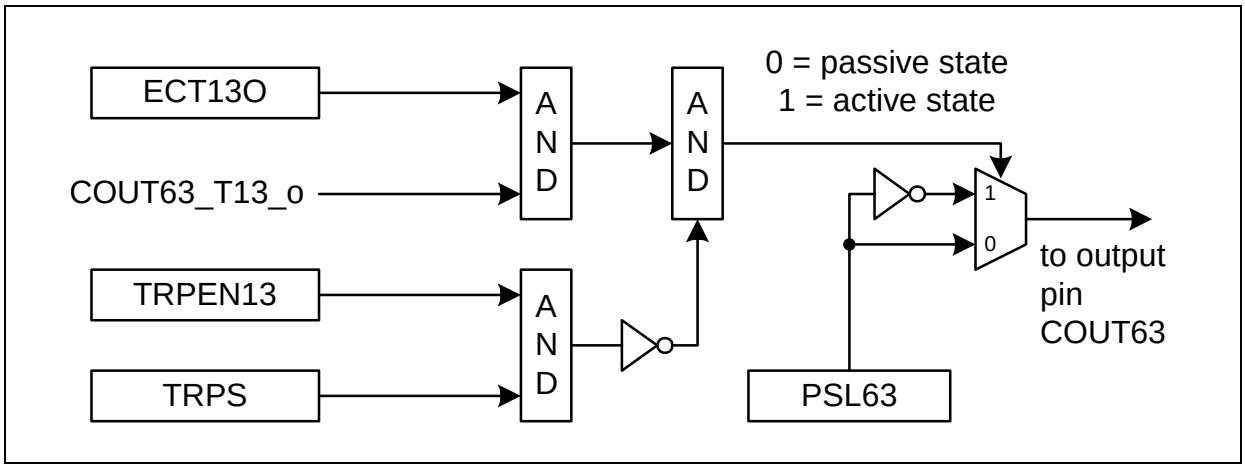


Figure 4-33 Modulation Control of the T13-related Output COUT63

Note: In order to avoid spikes on the output lines, the seven output signals (CC60, COUT60, CC61, COUT61, CC62, COUT62, COUT63) are registered out with the peripheral clock.

4.7.6 Hall Sensor Mode

In **Brushless-DC motors** the next multi-channel state values depend on the pattern of the Hall inputs. There is a strong correlation between the **Hall pattern** (CURH) and the **modulation pattern** (MCMP). Because of different machine types the modulation pattern for driving the motor can be different. Therefore it is wishful to have a wide flexibility in defining the correlation between the Hall pattern and the corresponding modulation pattern. The CCU6 offers this by having a register which contains the actual Hall pattern (CURHS), the next expected Hall pattern (EXPHS) and its output pattern (MCMPS). At every correct Hall event (CHE, see figure *Hall Event Actions*) a new Hall pattern with its corresponding output pattern can be loaded (from a predefined table) by software into the register MCMOUTS. Loading this shadow register can also be done by a write action on MCMOUTS with bit STRHP = ’1’.

The **sampling** of the Hall pattern (on CCPOSx) is done with the T12 clock. By using the dead-time counter DTC0 (mode MSEL6x= ’1000’) a hardware **noise filter** can be implemented to suppress spikes on the Hall inputs due to high di/dt in rugged inverter environment. In case of a Hall event the DTC0 is reloaded and starts counting. When the counter value of one is reached, the CCPOSx inputs are sampled (without noise and spikes) and are compared to the current Hall pattern (CURH) and to the expected Hall pattern (EXPH). If the sampled pattern equals to the current pattern the edge on CCPOSx was due to a noise spike and no action will be triggered (implicit noise filter). If

the sampled pattern equals to the next expected pattern the edge on CCPOSx was a correct Hall event, the bit CHE is set which causes an interrupt and the resets T12 (for speed measurement, see description mode ’1000’ below).

This correct Hall event can be used as a transfer request event for register MCMOUTS. The transfer from MCMOUTS to MCMOUT transfers the new CURH-pattern as well as the next EXPH-pattern. In case of the sampled Hall inputs were neither the current nor the expected Hall pattern, the bit WHE (wrong Hall event) is set which also can cause an interrupt and sets the IDLE mode clearing MCMP (modulation outputs are inactive). To restart from IDLE the transfer request of MCMOUTS have to be initiated by software (bit STRHP and bitfields SWSEL/SWSYN).

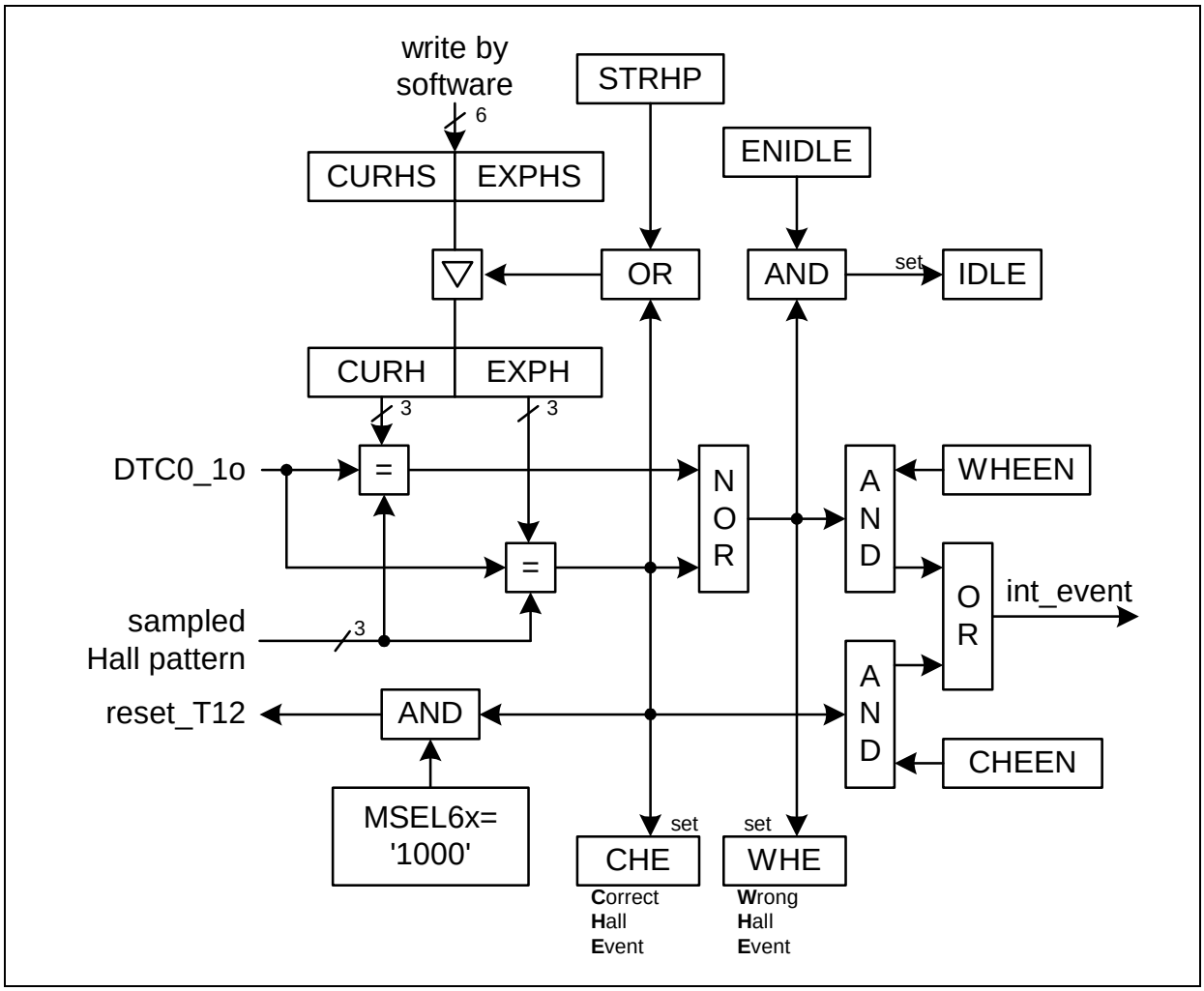


Figure 4-34 Hall Logic

For **Brushless-DC** motors there is a special mode (MSEL6x = ’1000b’) which is triggered by a change of the Hall-inputs (CCPOSx). This mode shows the capabilities of the CCU6 (see also figures *Multi-channel Selection and Synchronization, Hall Event Actions, Modulation Selection and Alternate Output Enable of T12* and *Timer T12 Brushless-DC Mode)*. Here T12’s channel 0 acts in capture function, channel 1 and 2 in compare function (without output modulation) and the multi-channel-block is used to trigger the output switching together with a possible modulation of T13.
After the detection of a valid Hall edge the T12 count value is captured to channel 0 (representing the actual motor speed) and resets the T12. When the timer reaches the compare value in channel 1, the next multi-channel state is switched by triggering the shadow transfer of bit field MCMP (if enabled in bit field SWEN). This trigger event can be combined with several conditions which are necessary to implement a noise filtering (correct Hall event) and to synchronize the next multi-channel state to the modulation sources (avoiding spikes on the output lines). This compare function of channel 1 can be used as a phase delay for the position input to the output switching which is necessary if a sensorless back-EMF technique is used instead of Hall sensors. The compare value in channel 2 can be used as a time-out trigger (interrupt) indicating that the motors destination speed is far below the desired value which can be caused by a abnormal load change. In this mode the modulation of T12 has to be disabled (T12MODENx = ’0’).

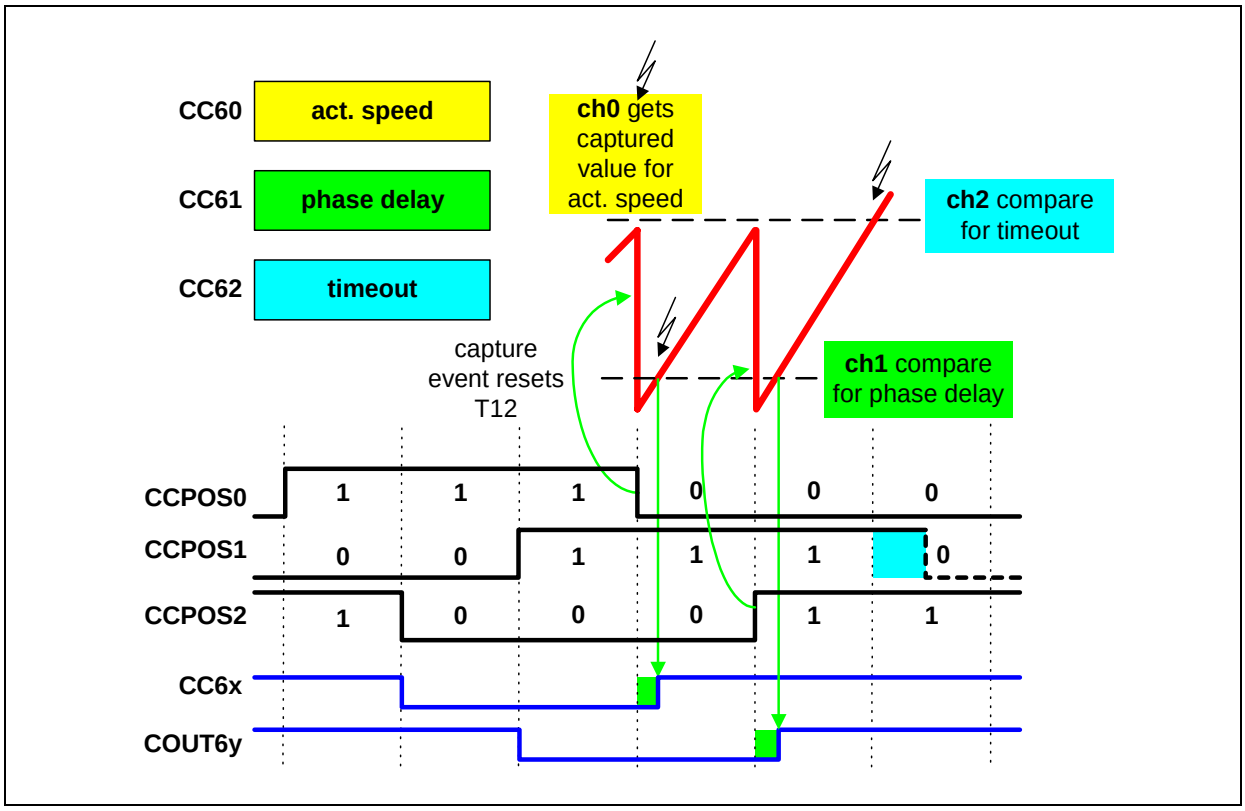


Figure 4-35 Timer T12 Brushless-DC Mode (MSEL6x = 1000)

4.7.7 Interrupt Generation

The interrupt structure is shown in **Figure 4-36**. The interrupt event or the corresponding interrupt set bit (in register ISS) can trigger the interrupt generation. The interrupt pulse is generated independently from the interrupt flag in register IS. The interrupt flag can be reset by SW by writing to the corresponding bit in register ISR.
If enabled by the related interrupt enable bit in register IEN, an interrupt pulse can be generated at one of the four interrupt output lines of the module (length 2 clock cycles). If more than one interrupt source is connected to the same interrupt node pointer (in register INP), the requests are combined to one common line.

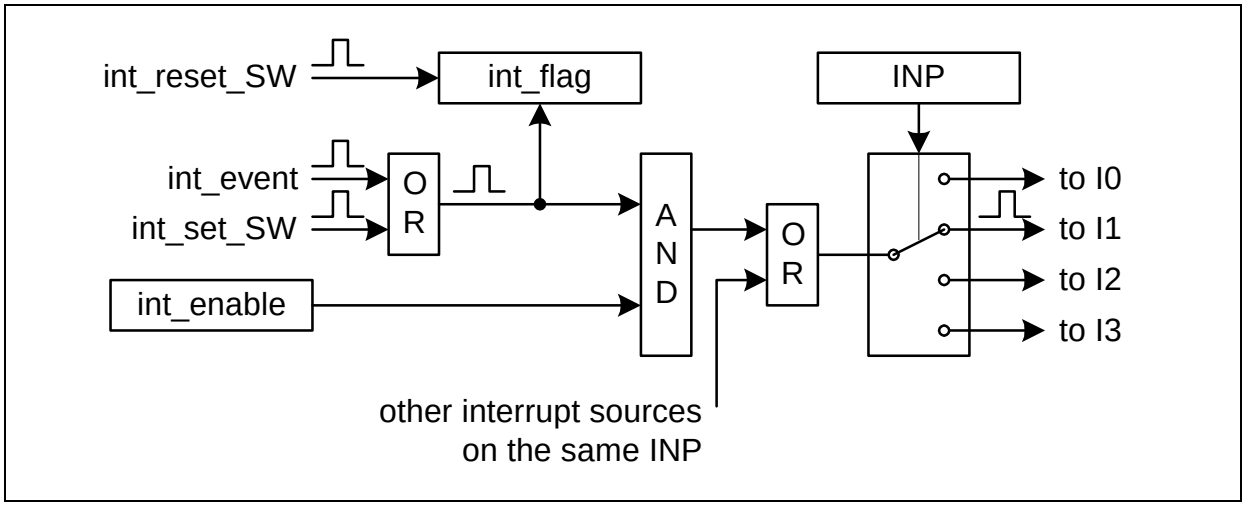


Figure 4-36 Interrupt Generation

4.7.8 Module Powerdown

The CCU6 is disabled when the chip goes into the powerdown mode as describe in . Or it can be individually disabled by setting CCUDIS in register PMCON1. This helps to reduce current consumption in the normal, slow down and idle modes of operation if the CCU6 is not utilized. Bit CCUST in register PMCON2 reflects the powerdown status of CCU6.

4.8 Kernel Description

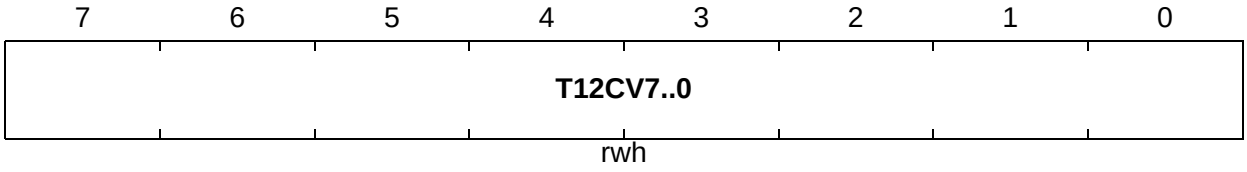
4.8.1 Register Overview

4.8.2 Timer12 - Related Registers

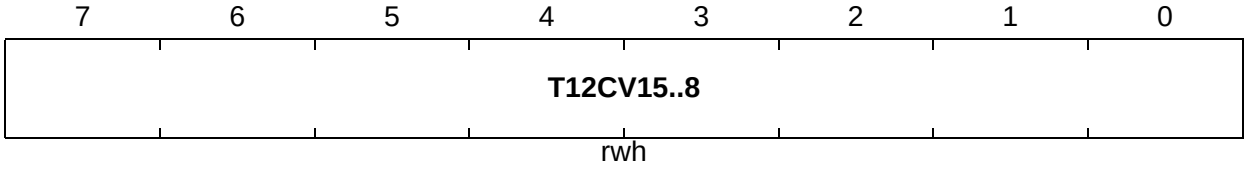
The generation of the patterns for a 3-channel pulse width modulation (PWM) is based on timer T12. The registers related to timer T12 can be concurrently updated (with well-defined conditions) in order to ensure consistency of the three PWM channels.
Timer T12 supports capture and compare modes, which can be independently selected for the three channels CC60, CC61 and CC62.

Register T12 represents the counting value of timer T12. It can only be written while the timer T12 is stopped. Write actions while T12 is running are not taken into account. Register T12 can always be read by SW.
In edge-aligned mode, T12 only counts up, whereas in center-aligned mode, T12 can count up and down.

T12L
Timer T12 Counter Register, Low Byte **[Reset value: 00_H]**



T12H
Timer T12 Counter Register, High Byte **[Reset value: 00_H]**



Field	Bits	Typ	Description
T12CV	[7:0] of T12 CVL, [7:0] of T12 CVH	rwh	**Timer 12 Counter Value** This register represents the 16-bit counter value of Timer12.

Note: While timer T12 is stopped, the internal clock divider is reset in order to ensure reproducible timings and delays.

Note: The timer period, compare values, passive state selects bits and passive levels bits for both timers are written to shadow registers and not directly to the actual registers. Thus the values for a new output signal can be programmed without disturbing the currently generated signal(s). The transfer from the shadow registers to the actual registers is enabled by setting the respective shadow transfer enable bit STEx.

If the transfer is enabled the shadow registers are copied to the respective registers as soon as the associated timer reaches the value zero the next time (being cleared in edge aligned mode or counting down from 1 in center aligned mode). When timer T12 is operating in center aligned mode, it will also copy the registers (if enabled by STE12) if it reaches the currently programmed period value (counting up).

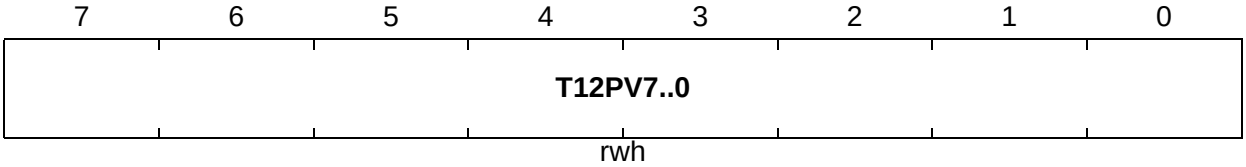
When a timer is stopped (TxR='0'), the shadow transfer takes place immediately if the corresponding bit STEx is set.

After the transfer the respective bit STEx is cleared automatically.

Register T12PR contains the period value for timer T12. The period value is compared to the actual counter value of T12 and the resulting counter actions depend on the defined counting rules. This register has a shadow register and the shadow transfer is controlled by bit STE12. A read action by SW delivers the value which is currently used for the compare action, whereas the write action targets a shadow register. The shadow register structure allows a concurrent update of all T12-related values.

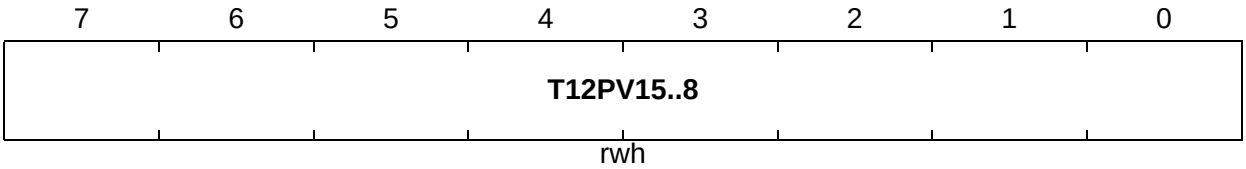
T12PRL

Timer T12 Period Register, Low Byte **[Reset value: 00_H]**



T12PRH

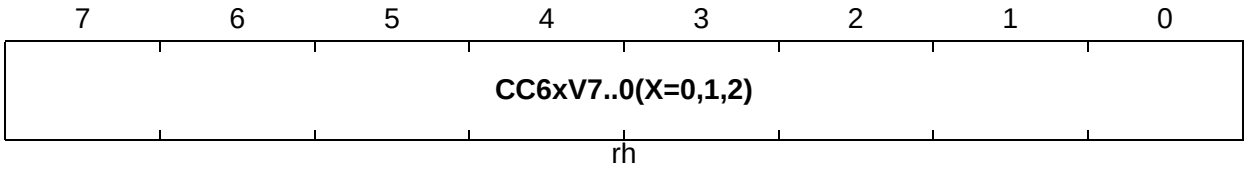
Timer T12 Period Register, High Byte **[Reset value: 00_H]**



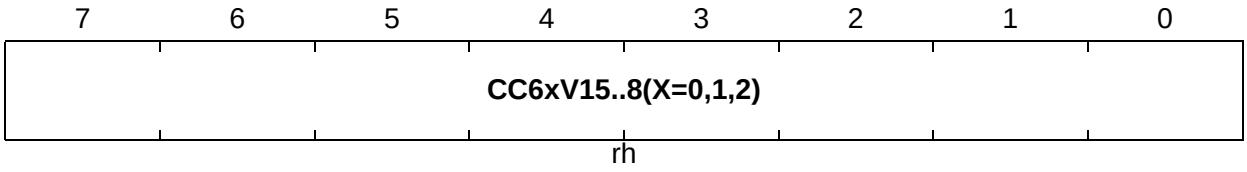
Field	Bits	Typ	Description
T12PV	[7:0] of T12 PRL, [7:0] of T12 PRH	rwh	**T12 Period Value** The value T12PV defines the counter value for T12, which leads to a period-match. When reaching this value, the timerT12 is set to zero (edge-aligned mode) or changes its count direction to down counting (center-aligned mode).

In compare mode, the registers CC6xR (x=0,1,2) are the actual compare registers for T12. The values stored in CC6xR are compared (all three channels in parallel) to the counter value of T12. In capture mode, the current value of the T12 counter register is captured by registers CC6xR if the corresponding capture event is detected.
The registers CC6xR can only be read by SW, the modification of the value is done by a shadow register transfer from register CC6xSR. The corresponding shadow registers CC6xSR can be read and written by SW. In capture mode, the value of the T12 counter register can also be captured by registers CC6xSR if the selected capture event is detected (depending on the selected mode).

CC6xRL (x=0,1,2)
Capture/Compare Register for Channel CC6x, Low Byte [Reset value: 00_H]

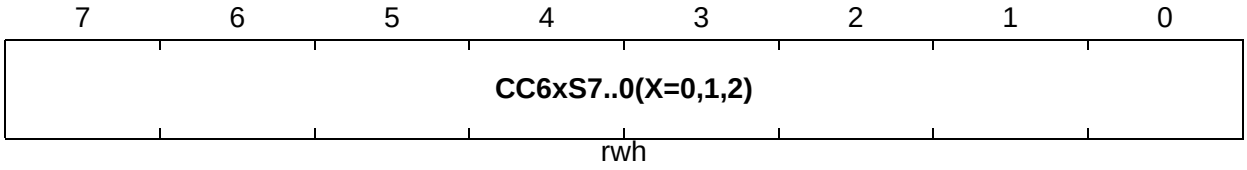


CC6xRH (X=0,1,2)
Capture/Compare Register for Channel CC6x, High Byte [Reset value: 00_H]

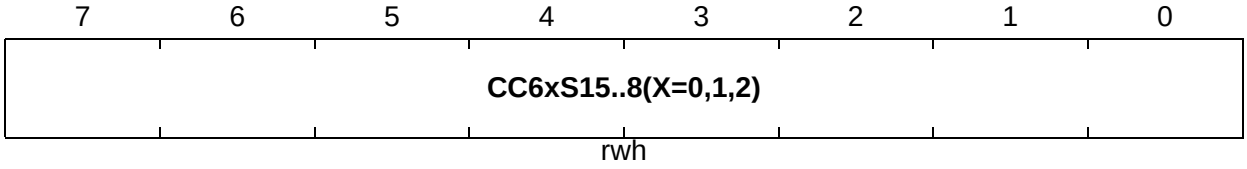


Field	Bits	Typ	Description
CC6xV (x=0, 1, 2)	[7:0] of CC6x RL, [7:0] of CC6x RH	rh	**Shadow Register for Channel x Capture/ Compare Value** In compare mode, the bitfields contents of CC6xS are transferred to the bitfields CC6xV during a shadow transfer. In capture mode, the captured value of T12 can be read from these registers.

CC6xSRL (x=0,1,2)
Capture/Compare Shadow Register for Channel CC6x, Low Byte[Reset value: 00_H]



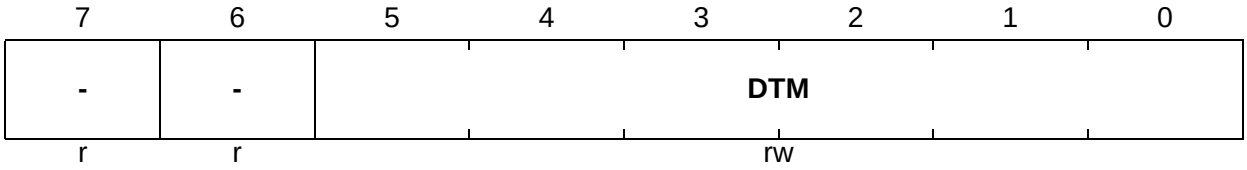
CC6xSRH (x=0,1,2)
Capture/Compare Shadow Register for Channel CC6x, High Byte[Reset value:00_H]



Field	Bits	Typ	Description
CC6xS (x=0, 1, 2)	[7:0] of CC6x SL, [7:0] of CC6x SH	rwh	**Shadow Register for Channel x Capture/ Compare Value** In compare mode, the bitfields contents of CC6xS are transferred to the bitfields CC6xV during a shadow transfer. In capture mode, the captured value of T12 can be read from these registers.

Register T12DTC controls the dead-time generation for the timer T12 compare channels. Each channel can be independently enabled/disabled for dead-time generation. If enabled, the transition from passive state to active state is delayed by the value defined by bit field DTM. The dead-time counter can only be reloaded while it is zero.

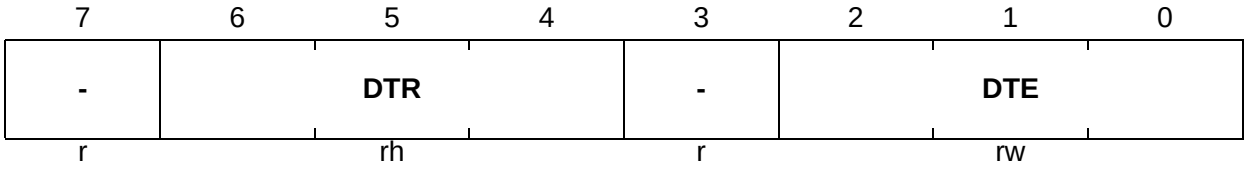
T12DTCL
Timer T12 Dead-Time Control Register,Low Byte **[Reset value: 00_H]**



Field	Bits	Typ	Description
DTM	[5:0]	rw	**Dead-Time** Bit field DTM determines the programmable delay between switching from the passive state to the active state of the selected outputs. The switching from the active state to the passive state is not delayed.
-	[7:6]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

T12DTCH

Timer T12 Dead-Time Control Register,High Byte **[Reset value: 00_H]**



Field	Bits	Typ	Description
DTE	[2:0]	rw	**Dead Time Enable Bits** Bits DTE0..DTE2 enable and disable the dead time generation for each compare channel (0, 1, 2) of timer T12. 0Dead time generation is disabled. The corresponding outputs switch from the passive state to the active state (according to the actual compare status) without any delay. 1Dead time generation is enabled. The corresponding outputs switch from the passive state to the active state (according to the compare status) with the delay programmed in bitfield DTM.
DTR	[6:4]	rh	**Dead Time Run Indication Bits** Bits DTR0..DTR2 indicate the status of the dead time generation for each compare channel (0, 1, 2) of timer T12. 0The value of the corresponding dead time counter channel is 0. 1The value of the corresponding dead time counter channel is not 0.
-	7,3	r	**reserved;** returns ’0’ if read; should be written with ’0’;

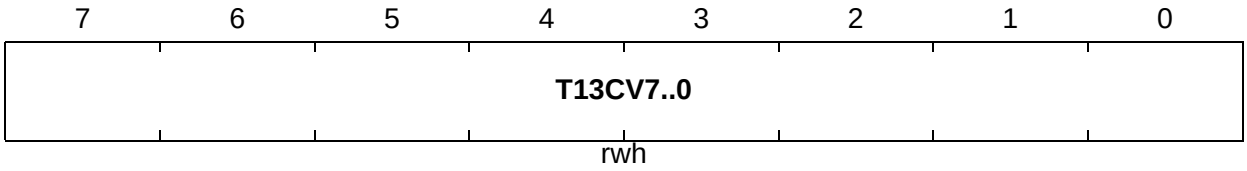
*Note: The dead time counters are clocked with the same frequency as T12. This structure allows symmetrical dead time generation in center-aligned and in edge-aligned PWM mode. A duty cycle of 50% leads to CC6x, COUT6x switched on for: 0.5 * period - dead time.*

Note: The dead-time counters are not reset by bit T12RES, but by bit DTRES.

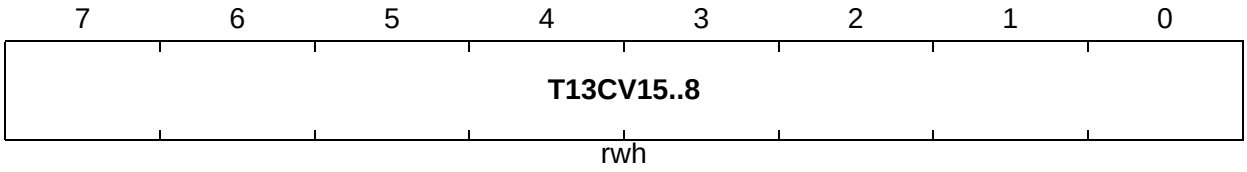
4.8.3 Timer13 - Related Registers

The generation of the patterns for a single channel pulse width modulation (PWM) is based on timer T13. The registers related to timer T13 can be concurrently updated (with well-defined conditions) in order to ensure consistency of the PWM signal. T13 can be synchronized to several timer T12 events.
Timer T13 only supports compare mode on its compare channel CC63.
Register T13 represents the counting value of timer T13. It can only be written while the timer T13 is stopped. Write actions while T13 is running are not taken into account. Register T13 can always be read by SW.
Timer T13 only supports edge-aligned mode (counting up).

T13L
Timer T13 Counter Register, Low Byte **[Reset value: 00_H]**



T13H
Timer T13 Counter Register, High Byte **[Reset value: 00_H]**

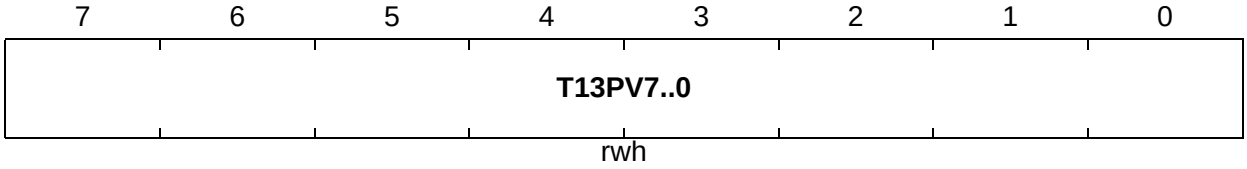


Field	Bits	Typ	Description
T13CV	[7:0] of T13L, [7:0] of T13H	rwh	**Timer 13 Counter Value** This register represents the 16-bit counter value of Timer13.

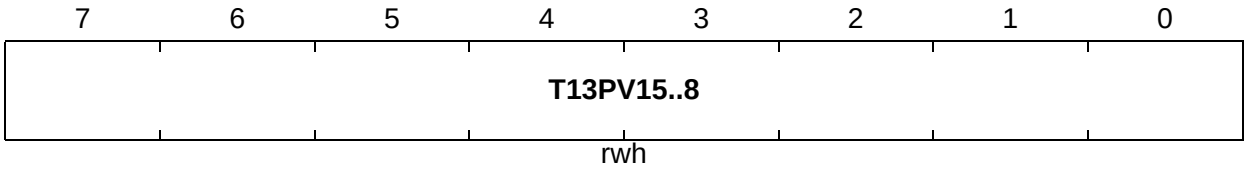
Note: While timer T13 is stopped, the internal clock divider is reset in order to ensure reproducible timings and delays.

Register T13PR contains the period value for timer T13. The period value is compared to the actual counter value of T13 and the resulting counter actions depend on the defined counting rules. This register has a shadow register and the shadow transfer is controlled by bit STE13. A read action by SW delivers the value which is currently used for the compare action, whereas the write action targets a shadow register. The shadow register structure allows a concurrent update of all T13-related values.

T13PRL
Timer T13 Period Register, Low Byte **[Reset value: 00_H]**



T13PRH
Timer T13 Period Register, High Byte **[Reset value: 00_H]**

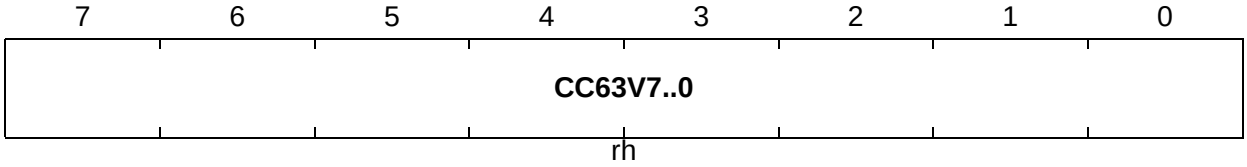


Field	Bits	Typ	Description
T13PV	[7:0] of T13L, [7:0] of T13H	rwh	**T13 Period Value** The value T13PV defines the counter value for T13, which leads to a period-match. When reaching this value, the timer T13 is set to zero.

Registers CC63R is the actual compare register for T13. The values stored in CC63R is compared to the counter value of T13. The register CC63R can only be read by SW, the modification of the value is done by a shadow register transfer from register CC63SR. The corresponding shadow register CC63SR can be read and written by SW.

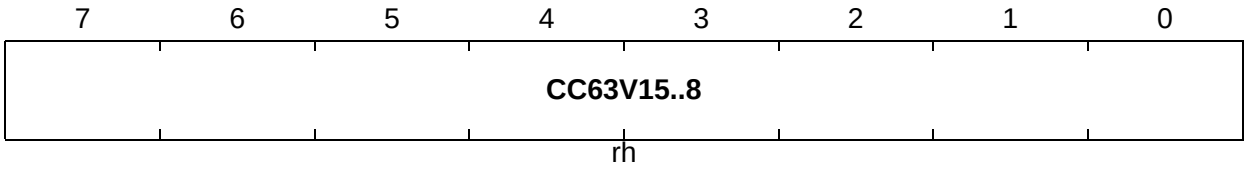
CC63RL

Compare Register for Channel CC63, Low Byte **[Reset value: 00_H]**



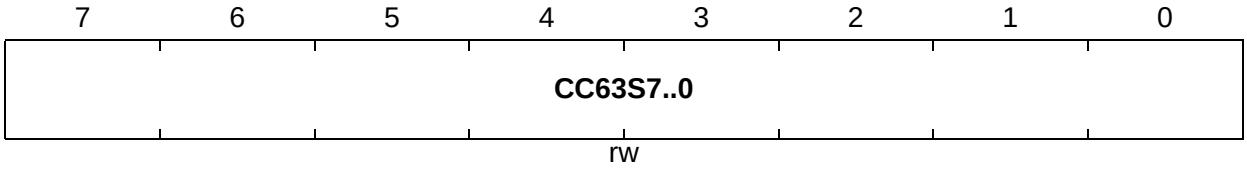
CC63RH

Compare Register for Channel CC63, High Byte **[Reset value: 00_H]**

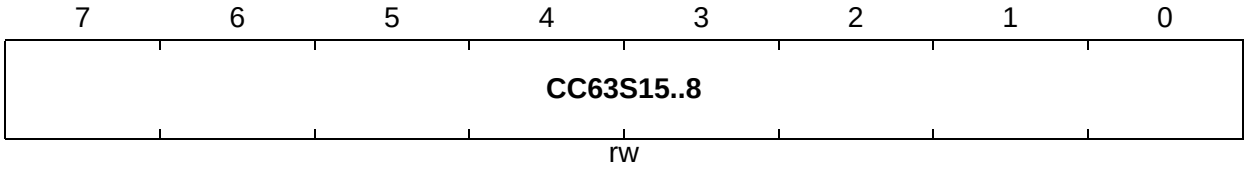


Field	Bits	Typ	Description
CC63V	[7:0] of CC63 RL, [7:0] of CC63 RH	rh	**Channel CC63 Compare Value** The bitfield CC63V contains the value, that is compared to the T13 counter value.

CC63SRL
Compare Shadow Register for CC63, Low Byte **[Reset value: 00_H]**



CC63SRH
Compare Shadow Register for CC63, High Byte **[Reset value: 00_H]**



Field	Bits	Typ	Description
CC63S	[7:0] of CC63 SRL, [7:0] of CC63 SRH	rw	**Shadow Register for Channel CC63 Compare Value** The bitfield contents of CC63S is transferred to the bitfield CC63V during a shadow transfer.

Capture/Compare Control Registers

The Compare State Register CMPSTAT contains status bits monitoring the current capture and compare state and control bits defining the active/passive state of the compare channels.

CMPSTATL
Compare State Register,Low Byte **[Reset value: 00_H]**

7	6	5	4	3	2	1	0
-	CC63ST	-	-	-	CC62ST	CC61ST	CC60ST
r	rh	r	r	r	rh	rh	rh

Field	Bits	Typ	Description
CC60ST **CC61ST** **CC62ST** **CC63ST** 1)	0 1 2 6	rh	**Capture/Compare State Bits** Bits CC6xST monitor the state of the capture/ compare channels. Bits CC6xST (x=0, 1, 2) are related to T12, bit CC63ST is related to T13. 0 In compare mode, the timer count is less than the compare value. In capture mode, the selected edge has not yet been detected since the bit has been reset by SW the last time. 1 In compare mode, the counter value is greater than or equal to the compare value. In capture mode, the selected edge has been detected.
-	[5:3], 7	r	**reserved;** returns ’0’ if read; should be written with ’0’;

1) These bits are set and reset according to the T12, T13 switching rules

CMPSTATH
Compare State Register,High Byte **[Reset value: 00_H]**

7	6	5	4	3	2	1	0
T13IM	COUT 63PS	COUT 62PS	CC62PS	COUT 61PS	CC61PS	COUT 60PS	CC60PS
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Typ	Description
CC60PS **CC61PS** **CC62PS** **COUT60PS** **COUT61PS** **COUT62PS** **COUT63PS** [1]	0 2 4 1 3 5 6	rwh	**Passive State Select for Compare Outputs** Bits CC6xPS, COUT6xPS select the state of the corresponding compare channel, which is considered to be the passive state. During the passive state, the passive level (defined in register PSLR) is driven by the output pin. Bits CC6xPS, COUT6xPS (x=0, 1, 2) are related to T12, bit CC63PS is related to T13. 0 The corresponding compare output drives passive level while CC6xST is '0'. 1 The corresponding compare output drives passive level while CC6xST is '1'. In capture mode, these bits are not used.
T13IM [2]	7	rwh	**T13 Inverted Modulation** Bit T13IM inverts the T13 signal for the modulation of the CC6x and COUT6x (x = 0, 1, 2) signals. 0 T13 output is not inverted. 1 T13 output is inverted for further modulation.

1) These bits have shadow bits and are updated in parallel to the capture/compare registers of T12, T13 respectively. A read action targets the actually used values, whereas a write action targets the shadow bits.

2) This bit has a shadow bit and is updated in parallel to the compare and period registers of T13. A read action targets the actually used values, whereas a write action targets the shadow bit.

The Compare Status Modification Register contains control bits allowing for modification by SW of the Capture/Compare state bits.

CMPMODIFL
Compare State Modification Register,Low Byte **[Reset value: 00_H]**

7	6	5	4	3	2	1	0
-	**MCC63S**	**-**	**-**	**-**	**MCC62S**	**MCC61S**	**MCC60S**
r	w	r	r	r	w	w	w

Field	Bits	Typ	Description
MCC60S **MCC61S** **MCC62S** **MCC63S**	0 1 2 7	w	**Capture/Compare Status Modification Bits** These bits are used to set (MCC6xR) the corresponding bits CC6xST by SW. This feature allows the user to individually change the status of the output lines by SW, e.g. when the corresponding compare timer is stopped. This allows a bit manipulation of CC6xST-bits by a single data write action. The following functionality of a write access to bits concerning the same capture/compare state bit is provided: MCC6xR(CMPMODIFH), MCC6xS = 0,0 Bit CC6xST is not changed. 0,1 Bit CC6xST is set. 1,0 Bit CC6xST is reset. 1,1 reserved (toggle)
-	[5:3], 7	r	**reserved;** returns ’0’ if read; should be written with ’0’;

CMPMODIFH
Compare State Modification Register,High Byte **[Reset value: 00_H]**

7	6	5	4	3	2	1	0
-	**MCC63R**	**-**	**-**	**-**	**MCC62R**	**MCC61R**	**MCC60R**
r	w	r	r	r	w	w	w

Field	Bits	Typ	Description
MCC60R **MCC61R** **MCC62R** **MCC63R**	0 1 2 7	w	**Capture/Compare Status Modification Bits** These bits are used to reset (MCC6xR) the corresponding bits CC6xST by SW. This feature allows the user to individually change the status of the output lines by SW, e.g. when the corresponding compare timer is stopped. This allows a bit manipulation of CC6xST-bits by a single data write action. The following functionality of a write access to bits concerning the same capture/compare state bit is provided: MCC6xR, MCC6xS(CMPMODIFL) = 0,0 Bit CC6xST is not changed. 0,1 Bit CC6xST is set. 1,0 Bit CC6xST is reset. 1,1 reserved (toggle)
-	[5:3], 7	r	**reserved;** returns ’0’ if read; should be written with ’0’;

Register TCTR0 controls the basic functionality of both timers T12 and T13.

TCTR0L
Timer Control Register 0,Low Byte **[Reset value: 00_H]**

7	6	5	4	3	2	1	0
CTM	**CDIR**	**STE12**	**T12R**	**T12PRE**	**T12CLK**		
rw	rh	rh	rh	rw	rw		

Field	Bits	Type	Description
T12CLK	[2:0]	rw	**Timer T12 Input Clock Select** Selects the input clock for timer T12 which is derived from the peripheral clock according to the equation $f_{T12} = f_{per} / 2^{<T12CLK>}$. 000 $f_{T12} = f_{per}$ 001 $f_{T12} = f_{per} / 2$ 010 $f_{T12} = f_{per} / 4$ 011 $f_{T12} = f_{per} / 8$ 100 $f_{T12} = f_{per} / 16$ 101 $f_{T12} = f_{per} / 32$ 110 $f_{T12} = f_{per} / 64$ 111 $f_{T12} = f_{per} / 128$
T12PRE	3	rw	**Timer T12 Prescaler Bit** In order to support higher clock frequencies, an additional prescaler factor of 1/256 can be enabled for the prescaler for T12. 0 The additional prescaler for T12 is disabled. 1 The additional prescaler for T12 is enabled.
T12R[1)]	4	rh	**Timer T12 Run Bit** T12R starts and stops timer T12. It is set/reset by SW by setting bits T12RR orT12RS or it is reset by HW according to the function defined by bitfield T12SSC. 0 Timer T12 is stopped. 1 Timer T12 is running.

Field	Bits	Type	Description
STE12	5	rh	**Timer T12 Shadow Transfer Enable** Bit STE12 enables or disables the shadow transfer of the T12 period value, the compare values and passive state select bits and levels from their shadow registers to the actual registers if a T12 shadow transfer event is detected. Bit STE12 is cleared by hardware after the shadow transfer. A T12 shadow transfer event is a period-match while counting up or a one-match while counting down. 0 The shadow register transfer is disabled. 1 The shadow register transfer is enabled.
CDIR	6	rh	**Count Direction of Timer T12** This bit is set/reset according to the counting rules of T12. 0 T12 counts up. 1 T12 counts down.
CTM	7	rw	**T12 Operating Mode** 0 Edge-aligned Mode: T12 always counts up and continues counting from zero after reaching the period value. 1 Center-aligned Mode: T12 counts down after detecting a period-match and counts up after detecting a one-match.

1) A concurrent set/reset action on T12R (from T12SSC, T12RR or T12RS) will have no effect. The bit T12R will remain unchanged.

Note: A write action to the bit fields T12CLK or T12PRE is only taken into account while the timer T12 is not running (T12R=0).

TCTR0H
Timer Control Register 0,High Byte **[Reset value: 00_H]**

7	6	5	4	3	2	1	0
-	-	STE13	T13R	T13PRE	T13CLK		
r	r	rh	rh	rw	rw		

Field	Bits	Type	Description
T13CLK	[2:0]	rw	**Timer T13 Input Clock Select** Selects the input clock for timer T13 which is derived from the peripheral clock according to the equation $f_{T13} = f_{per} / 2^{<T13CLK>}$. 000 $f_{T13} = f_{per}$ 001 $f_{T13} = f_{per} / 2$ 010 $f_{T13} = f_{per} / 4$ 011 $f_{T13} = f_{per} / 8$ 100 $f_{T13} = f_{per} / 16$ 101 $f_{T13} = f_{per} / 32$ 110 $f_{T13} = f_{per} / 64$ 111 $f_{T13} = f_{per} / 128$
T13PRE	3	rw	**Timer T13 Prescaler Bit** In order to support higher clock frequencies, an additional prescaler factor of 1/256 can be enabled for the prescaler for T13. 0 The additional prescaler for T13 is disabled. 1 The additional prescaler for T13 is enabled.
T13R[1)]	4	rh	**Timer T13 Run Bit** T13R starts and stops timer T13. It is set/reset by SW by setting bits T13RR orT13RS or it is set/reset by HW according to the function defined by bitfields T13SSC, T13TEC and T13TED. 0 Timer T13 is stopped. 1 Timer T13 is running.

Field	Bits	Type	Description
STE13	5	rh	**Timer T13 Shadow Transfer Enable** Bit STE13 enables or disables the shadow transfer of the T13 period value, the compare value and passive state select bit and level from their shadow registers to the actual registers if a T13 shadow transfer event is detected. Bit STE13 is cleared by hardware after the shadow transfer. A T13 shadow transfer event is a period-match. 0 The shadow register transfer is disabled. 1 The shadow register transfer is enabled.
-	[7: 6]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

1) A concurrent set/reset action on T13R (from T13SSC, T13TEC, T13RR or T13RS) will have no effect. The bit T12R will remain unchanged.

Note: A write action to the bit fields T13CLK or T13PRE is only taken into account while the timer T13 is not running (T13R=0).

Register TCTR2 controls the single-shot and the synchronization functionality of both timers T12 and T13. Both timers can run in single-shot mode. In this mode they stop their counting sequence automatically after one counting period with a count value of zero. The single-shot mode and the synchronization feature of T13 to T12 allows the generation of events with a programmable delay after well-defined PWM actions of T12. For example, this feature can be used to trigger AD conversions after a specified delay (to avoid problems due to switching noise) synchronously to a PWM event.

TCTR2L
Timer Control Register 2 **[Reset value: 00_H]**

7	6	5	4	3	2	1	0
-	T13TED		T13TEC			T13SSC	T12SSC
r	rw		rw			rw	rw

Field	Bits	Type	Description
T12SSC	0	rw	**Timer T12 Single Shot Control** This bit controls the single shot-mode of T12. 0 The single-shot mode is disabled, no HW action on T12R. 1 The single shot mode is enabled, the bit T12R is reset by HW if - T12 reaches its period value in edge-aligned mode - T12 reaches the value 1 while down counting in center-aligned mode. In parallel to the reset action of bit T12R, the bits CC6xST (x=0, 1, 2) are reset.
T13SSC	1	rw	**Timer T13 Single Shot Control** This bit controls the single shot-mode of T13. 0 No HW action on T13R 1 The single-shot mode is enabled, the bit T13R is reset by HW if T13 reaches its period value. In parallel to the reset action of bit T13R, the bit CC63ST is reset.

Field	Bits	Type	Description
T13TEC	[4:2]	rw	**T13 Trigger Event Control** Bitfield T13TEC selects the trigger event to start T13 (automatic set of T13R for synchronization to T12 compare signals) according to following combinations: 000 no action 001 set T13R on a T12 compare event on channel 0 010 set T13R on a T12 compare event on channel 1 011 set T13R on a T12 compare event on channel 2 100 set T13R on any T12 compare event (ch. 0, 1, 2) 101 set T13R upon a period-match of T12 110 set T13R upon a zero-match of T12 (while counting up) 111 set T13R on any hall state change
T13TED[1)]	[6:5]	rw	**Timer T13 Trigger Event Direction** Bitfield T13TED delivers additional information to control the automatic set of bit T13R in the case that the trigger action defined by T13TEC is detected. 00 reserved, no action 01 while T12 is counting up 10 while T12 is counting down 11 independent on the count direction of T12
0	7	r	**reserved;** returns ’0’ if read; should be written with ’0’;

1) Example:
If the timer T13 is intended to start at any compare event on T12 (T13TEC=’100’) the trigger event direction can be programmed to
- counting up >> a T12 channel 0, 1, 2 compare match triggers T13R only while T12 is counting up
- counting down >> a T12 channel 0, 1, 2 compare match triggers T13R only while T12 is counting down
- independent from bit CDIR >> each T12 channel 0, 1, 2 compare match triggers T13R

The timer count direction is taken from the value of bit CDIR. As a result, if T12 is running in edge-aligned mode (counting up only), T13 can only be started automatically if bitfield T13TED=’01’ or ’11’.

Register TCTR4 allows the SW control of the run bits T12R and T13R by independent set and reset conditions. Furthermore, the timers can be reset (while running) and the bits STE12 and STE13 can be controlled by SW.

TCTR4L
Timer Control Register 4,Low Byte **[Reset value: 00_H]**

7	6	5	4	3	2	1	0
T12STD	**T12STR**	**-**	**-**	**DTRES**	**T12RES**	**T12RS**	**T12RR**
w	w	r	r	w	w	w	w

Field	Bits	Type	Description
T12RR	0	w	**Timer T12 Run Reset** Setting this bit resets the T12R bit. 0 T12R is not influenced. 1 T12R is cleared, T12 stops counting.
T12RS	1	w	**Timer T12 Run Set** Setting this bit sets the T12R bit. 0 T12R is not influenced. 1 T12R is set, T12 starts counting.
T12RES	2	w	**Timer T12 Reset** 0 No effect on T12. 1 The T12 counter register is reset to zero. The switching of the output signals is according to the switching rules. Setting of T12RES has no impact on bit T12R.
DTRES	3	w	**Dead-Time Counter Reset** 0 No effect on the dead-time counters. 1 The three dead-time counter channels are reset to zero.
T12STR	6	w	**Timer T12 Shadow Transfer Request** 0 No action 1 STE12 is set, enabling the shadow transfer.
T12STD	7	w	**Timer T12 Shadow Transfer Disable** 0 No action 1 STE12 is reset without triggering the shadow transfer.
-	[5:4]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

Note: A simultaneous write of a ’1’ to bits which set and reset the same bit will trigger no action. The corresponding bit will remain unchanged.

TCTR4H
Timer Control Register 4,High Byte **[Reset value: 00_H]**

7	6	5	4	3	2	1	0
T13STD	**T13STR**	**-**	**-**	**-**	**T13RES**	**T13RS**	**T13RR**
w	w	r	r	r	w	w	w

Field	Bits	Type	Description
T13RR	0	w	**Timer T13 Run Reset** Setting this bit resets the T13R bit. 0 T13R is not influenced. 1 T13R is cleared, T13 stops counting.
T13RS	1	w	**Timer T13 Run Set** Setting this bit sets the T13R bit. 0 T13R is not influenced. 1 T13R is set, T13 starts counting.
T13RES	2	w	**Timer T13 Reset** 0 No effect on T13. 1 The T13 counter register is reset to zero. The switching of the output signals is according to the switching rules. Setting of T13RES has no impact on bit T13R.
T13STR	6	w	**Timer T13 Shadow Transfer Request** 0 No action 1 STE13 is set, enabling the shadow transfer.
T13STD	7	w	**Timer T13 Shadow Transfer Disable** 0 No action 1 STE13 is reset without triggering the shadow transfer.
-	[5:3]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

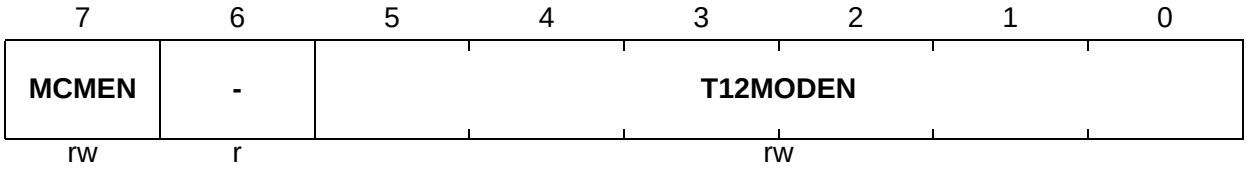
Note: A simultaneous write of a ’1’ to bits which set and reset the same bit will trigger no action. The corresponding bit will remain unchanged.

4.8.4 Modulation Control Registers

4.8.4.1 Global Module Control

Register MODCTR contains control bits enabling the modulation of the corresponding output signal by PWM pattern generated by the timers T12 and T13. Furthermore, the multi-channel mode can be enabled as additional modulation source for the output signals.

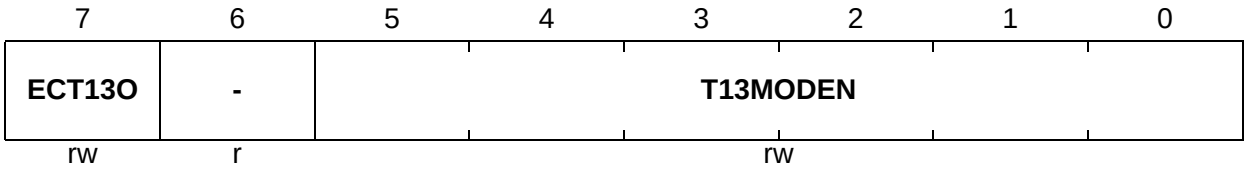
MODCTRL
Modulation Control Register ,Low Byte **[Reset value: 00_H]**



Field	Bits	Type	Description
T12MODEN	[5:0]	rw	**T12 Modulation Enable** Setting these bits enables the modulation of the corresponding compare channel by a PWM pattern generated by timer T12. The bit positions are corresponding to the following output signals: bit 0 modulation of CC60 bit 1 modulation of COUT60 bit 2 modulation of CC61 bit 3 modulation of COUT61 bit 4 modulation of CC62 bit 5 modulation of COUT62 The enable feature of the modulation is defined as follows: 0 The modulation of the corresponding output signal by a T12 PWM pattern is disabled. 1 The modulation of the corresponding output signal by a T12 PWM pattern is enabled.
MCMEN	7	rw	**Multi-Channel Mode Enable** 0 The modulation of the corresponding output signal by a multi-channel pattern according to bitfield MCMOUT is disabled. 1 The modulation of the corresponding output signal by a multi-channel pattern according to bitfield MCMOUT is enabled.

Field	Bits	Type	Description
-	6	r	**reserved;** returns ’0’ if read; should be written with ’0’;

MODCTRH
Modulation Control Register ,High Byte **[Reset value: 00_H]**



Field	Bits	Type	Description
T13MODEN	[5:0]	rw	**T13 Modulation Enable** Setting these bits enables the modulation of the corresponding compare channel by a PWM pattern generated by timer T13. The bit positions are corresponding to the following output signals: bit 0 modulation of CC60 bit 1 modulation of COUT60 bit 2 modulation of CC61 bit 3 modulation of COUT61 bit 4 modulation of CC62 bit 5 modulation of COUT62 The enable feature of the modulation is defined as follows: 0 The modulation of the corresponding output signal by a T13 PWM pattern is disabled. 1 The modulation of the corresponding output signal by a T13 PWM pattern is enabled.
ECT13O	7	rw	**Enable Compare Timer T13 Output** 0 The alternate output function COUT63 is disabled. 1 The alternate output function COUT63 is enabled for the PWM signal generated by T13.
-	14	r	**reserved;** returns ’0’ if read; should be written with ’0’;

The register TRPCTR controls the trap functionality. It contains independent enable bits for each output signal and control bits to select the behavior in case of a trap condition. The trap condition is a low level on the CTRAP input pin, which is monitored (inverted level) by bit TRPF (in register IS). While TRPF='1' (trap input active), the trap state bit TRPS (in register IS) is set to '1'.

TRPCTRL
Trap Control Register ,Low Byte **[Reset value: 00_H]**

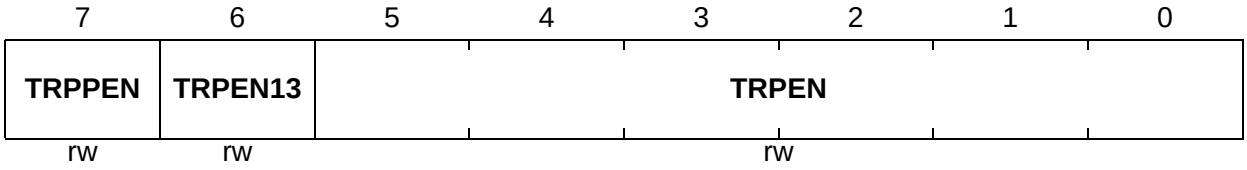
7	6	5	4	3	2	1	0
-	-	-	-	-	**TRPM2**	**TRPM1**	**TRPM0**
r	r	r	r	r	rw	rw	rw

Field	Bits	Type	Description
TRPM1, TRPM0	[1:0]	rw	**Trap Mode Control Bits 1, 0** These two bits define the behavior of the selected outputs when leaving the trap state after the trap condition has become inactive again. A synchronization to the timer driving the PWM pattern permits to avoid unintended short pulses when leaving the trap state. The combination (TRPM1, TRPM0) leads to: 00 The trap state is left (return to normal operation according to TRPM2) when a zero-match of T12 (while counting up) is detected (synchronization to T12). 01 The trap state is left (return to normal operation according to TRPM2) when a zero-match of T13 is detected (synchronization to T13). 10 reserved 11 The trap state is left (return to normal operation according to TRPM2) immediately without any synchronization to T12 or T13.

Field	Bits	Type	Description
TRPM2	2	rw	**Trap Mode Control Bit 2** 0 The trap state can be left (return to normal operation = bit TRPS=’0’) as soon as the input CTRAP becomes inactive. Bit TRPF is automatically cleared by HW if the input pin CTRAP becomes ’1’. Bit TRPS is automatically cleared by HW if bit TRPF is ’0’ and if the synchronization condition (according to TRPM0,1) is detected. 1 The trap state can be left (return to normal operation = bit TRPS=’0’) as soon as bit TRPF is reset by SW after the input CTRAP becomes inactive (TRPF is not cleared by HW). Bit TRPS is automatically cleared by HW if bit TRPF=’0’ and if the synchronization condition (according to TRPM0,1) is detected.
-	[7:3]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

TRPCTRH

Trap Control Register ,High Byte **[Reset value: 00_H]**

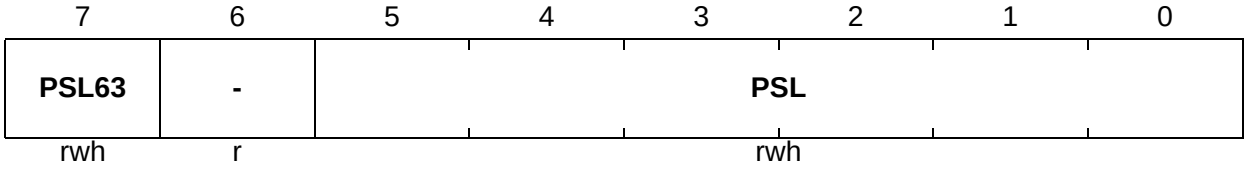


Field	Bits	Type	Description
TRPEN	[5:0]	rw	**Trap Enable Control** Setting these bits enables the trap functionality for the following corresponding output signals: bit 0 trap functionality of CC60 bit 1 trap functionality of COUT60 bit 2 trap functionality of CC61 bit 3 trap functionality of COUT61 bit 4 trap functionality of CC62 bit 5 trap functionality of COUT62 The enable feature of the trap functionality is defined as follows: 0 The trap functionality of the corresponding output signal is disabled. The output state is independent from bit TRPS. 1 The trap functionality of the corresponding output signal is enabled. The output is set to the passive state while TRPS='1'.
TRPEN13	6	rw	**Trap Enable Control for Timer T13** 0 The trap functionality for T13 is disabled. Timer T13 (if selected and enabled) provides PWM functionality even while TRPS='1'. 1 The trap functionality for T13 is enabled. The timer T13 PWM output signal is set to the passive state while TRPS='1'.
TRPPEN	7	rw	**Trap Pin Enable** 0 The trap functionality based on the input pin CTRAP is disabled. A trap can only be generated by SW by setting bit TRPF. 1 The trap functionality based on the input pin CTRAP is enabled. A trap can be generated by SW by setting bit TRPF or by CTRAP='0'.

Register PSLR defines the passive state level driven by the output pins of the module. The passive state level is the value that is driven by the port pin during the passive state of the output. During the active state, the corresponding output pin drives the active state level, which is the inverted passive state level. The passive state level permits to adapt the driven output levels to the driver polarity (inverted, not inverted) of the connected power stage.

PSLRL

Passive State Level Register ,High Byte **[Reset value: 00_H]**



Field	Bits	Type	Description
PSL[1)]	[5:0]	rwh	**Compare Outputs Passive State Level** The bits of this bitfield define the passive level driven by the module outputs during the passive state. The bit positions are: bit 0 passive level for output CC60 bit 1 passive level for output COUT60 bit 2 passive level for output CC61 bit 3 passive level for output COUT61 bit 4 passive level for output CC62 bit 5 passive level for output COUT62 The value of each bit position is defined as: 0 The passive level is ’0’. 1 The passive level is ’1’.
PSL63[2)]	7	rwh	**Passive State Level of Output COUT63** This bitfield defines the passive level of the output pin COUT63. 0 The passive level is ’0’. 1 The passive level is ’1’.
-	6	r	**reserved;** returns ’0’ if read; should be written with ’0’;

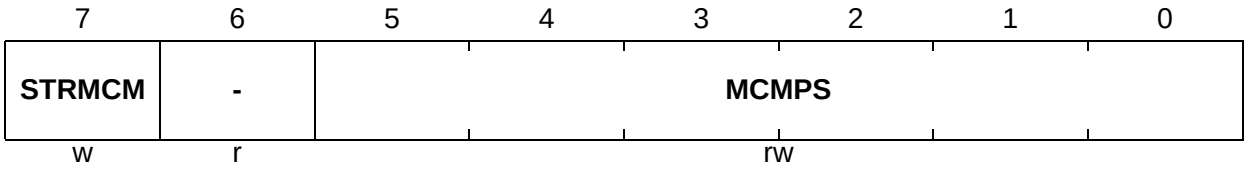
1) Bitfield PSL has a shadow registers to allow for updates without undesired pulses on the output lines. The bits are updated with the T12 shadow transfer. A read action targets the actually used values, whereas a write action targets the shadow bits.

2) Bit PSL63 has a shadow register to allow for updates without undesired pulses on the output line. The bit is updated with the T13 shadow transfer. A read action targets the actually used values, whereas a write action targets the shadow bits.

4.8.4.2 Multi-Channel Control

Register MCMOUTS contains bits controlling the output states for multi-channel mode. Furthermore, the appropriate signals for the block commutation by Hall sensors can be selected. This register is a shadow register (that can be written) for register MCMOUT, which indicates the currently active signals.

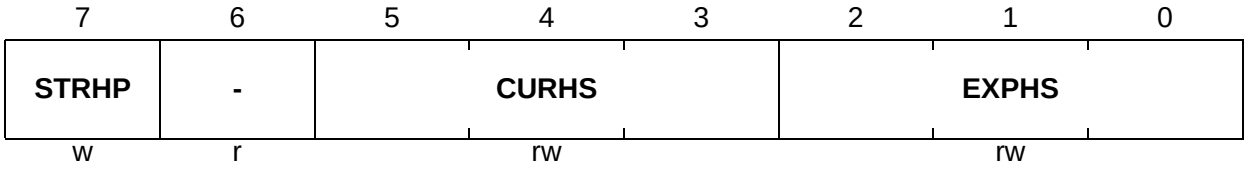
MCMOUTSL
Multi-Channel Mode Output Shadow Register ,Low Byte [Reset value: 00_H]



Field	Bits	Type	Description
MCMPS	[5:0]	rw	**Multi-Channel PWM Pattern Shadow** Bitfield MCMPS is the shadow bitfield for bitfield MCMP. The multi-channel shadow transfer is triggered according to the transfer conditions defined by register MCMCTR.
STRMCM	7	w	**Shadow Transfer Request for MCMPS** Setting this bits during a write action leads to an immediate update of bitfield MCMP by the value written to bitfield MCMPS. This functionality permits an update triggered by SW. When read, this bit always delivers ’0’. 0 Bitfield MCMP is updated according to the defined HW action. The write access to bitfield MCMPS doesn’t modify bitfield MCMP. 1 Bitfield MCMP is updated by the value written to bitfield MCMPS.
-	6	r	**reserved;** returns ’0’ if read; should be written with ’0’;

MCMOUTSH

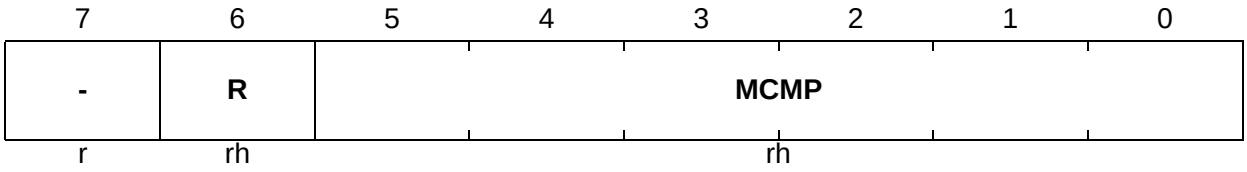
Multi-Channel Mode Output Shadow Register ,High Byte [Reset value: 00_H]



Field	Bits	Type	Description
EXPHS	[2:0]	rw	**Expected Hall Pattern Shadow** Bitfield EXPHS is the shadow bitfield for bitfield EXPH. The bitfield is transferred to bitfield EXPH if an edge on the hall input pins CCPOSx (x=0, 1, 2) is detected.
CURHS	[5:3]	rw	**Current Hall Pattern Shadow** Bitfield CURHS is the shadow bitfield for bitfield CURH. The bitfield is transferred to bitfield CURH if an edge on the hall input pins CCPOSx (x=0, 1, 2) is detected.
STRHP	7	w	**Shadow Transfer Request for the Hall Pattern** Setting this bits during a write action leads to an immediate update of bitfields CURH and EXPH by the value written to bitfields CURHS and EXPH. This functionality permits an update triggered by SW. When read, this bit always delivers ’0’. 0 The bitfields CURH and EXPH are updated according to the defined HW action. The write access to bitfields CURHS and EXPH doesn’t modify the bitfields CURH and EXPH. 1 The bitfields CURH and EXPH are updated by the value written to the bitfields CURHS and EXPHS.
-	6	r	**reserved;** returns ’0’ if read; should be written with ’0’;

Register MCMOUT shows the multi-channel control bits, that are currently used. Register MCMOUT is defined as follows:

MCMOUTL
Multi-Channel Mode Output Register ,Low Byte **[Reset value: 00_H]**



Field	Bits	Type	Description
MCMP[1)]	[5:0]	rh	**Multi-Channel PWM Pattern** Bitfield MCMP is written by a shadow transfer from bitfield MCMPS. It contains the output pattern for the multi-channel mode. If this mode is enabled by bit MCMEN in register MODCTR, the output state of the following output signal can be modified: bit 0 multi-channel state for output CC60 bit 1 multi-channel state for output COUT60 bit 2 multi-channel state for output CC61 bit 3 multi-channel state for output COUT61 bit 4 multi-channel state for output CC62 bit 5 multi-channel state for output COUT62 The multi-channel patterns can set the related output to the passive state. 0 The output is set to the passive state. The PWM generated by T12 or T13 are not taken into account. 1 The output can deliver the PWM generated by T12 or T13 (according to register MODCTR).

Field	Bits	Type	Description
R	6	rh	**Reminder Flag** This reminder flag indicates that the shadow transfer from bitfield MCMPS to MCMP has been requested by the selected trigger source. This bit is cleared when the shadow transfer takes place and while MCMEN='0'. 0 Currently, no shadow transfer from MCMPS to MCMP is requested. 1 A shadow transfer from MCMPS to MCMP has been requested by the selected trigger source, but it has not yet been executed, because the selected synchronization condition has not yet occurred.
-	7	r	**reserved;** returns '0' if read; should be written with '0';

1) While IDLE='1', bit field MCMP is cleared.

MCMOUTH

Multi-Channel Mode Output Register ,High Byte **[Reset value: 00_H]**

7	6	5	4	3	2	1	0
-	-	CURH			EXPH		
r	r	rw			rw		

Field	Bits	Type	Description
EXPH[1)]	[2:0]	rh	**Expected Hall Pattern** Bitfield EXPH is written by a shadow transfer from bitfield EXPHS. The contents is compared after every detected edge at the hall input pins with the pattern at the hall input pins in order to detect the occurrence of the next desired (=expected) hall pattern or a wrong pattern. If the current hall pattern at the hall input pins is equal to the bitfield EXPH, bit CHE (correct hall event) is set and an interrupt request is generated (if enabled by bit ENCHE). If the current hall pattern at the hall input pins is not equal to the bitfields CURH or EXPH, bit WHE (wrong hall event) is set and an interrupt request is generated (if enabled by bit ENWHE).
CURH	[5:3]	rh	**Current Hall Pattern** Bitfield CURH is written by a shadow transfer from bitfield CURHS.The contents is compared after every detected edge at the hall input pins with the pattern at the hall input pins in order to detect the occurrence of the next desired (=expected) hall pattern or a wrong pattern. If the current hall input pattern is equal to bitfield CURH, the detected edge at the hall input pins has been an invalid transition (e.g. a spike).
-	[7:6]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

1) The bits in the bit fields EXPH and CURH correspond to the hall patterns at the input pins CCPOSx (x=0, 1, 2) in the order (EXPH.2, EXPH.1, EXPH.0), (CURH.2, CURH.1, CURH.0), (CCPOS2, CCPOS.1, CCPOS0).

Register MCMCTR contains control bits for the multi-channel functionality.

MCMCTRLL
Multi-Channel Mode Control Register **[Reset value: 00_H]**

7	6	5	4	3	2	1	0
-	-	SWSYN		-	SWSEL		
r	r	rw		r	rw		

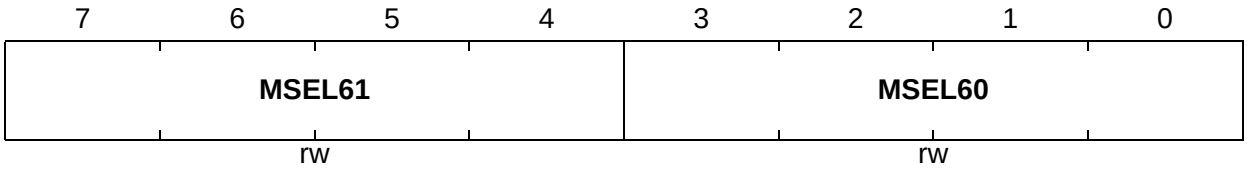
Field	Bits	Type	Description
SWSEL	[2:0]	rw	**Switching Selection** Bitfield SWSEL selects one of the following trigger request sources (next multi-channel event) for the shadow transfer from MCMPS to MCMP. The trigger request is stored in the reminder flag R until the shadow transfer is done and flag R is cleared automatically with the shadow transfer. The shadow transfer takes place synchronously with an event selected in bitfield SWSYN. 000 no trigger request will be generated 001 correct hall pattern on CCPOSx detected 010 T13 period-match detected (while counting up) 011 T12 one-match (while counting down) 100 T12 channel 1 compare-match detected (phase delay function) 101 T12 period match detected (while counting up) else reserved, no trigger request will be generated
SWSYN	[5:4]	rw	**Switching Synchronization** Bitfield SWSYN triggers the shadow transfer between MCMPS and MCMP if it has been requested before (flag R set by an event selected by SWSEL). This feature permits the synchronization of the outputs to the PWM source, that is used for modulation (T12 or T13). 00 direct; the trigger event directly causes the shadow transfer 01 T13 zero-match triggers the shadow transfer 10 a T12 zero-match (while counting up) triggers the shadow transfer 11 reserved; no action

Field	Bits	Type	Description
-	3, [7:6]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

Note: The generation of the shadow transfer request by HW is only enabled if bit MCMEN=’1’.

Register T12MSEL contains control bits to select the capture/compare functionality of the three channels of timer T12.

T12MSELL
T12 Capture/Compare Mode Select Register, Low Byte **[Reset value: 00_H]**

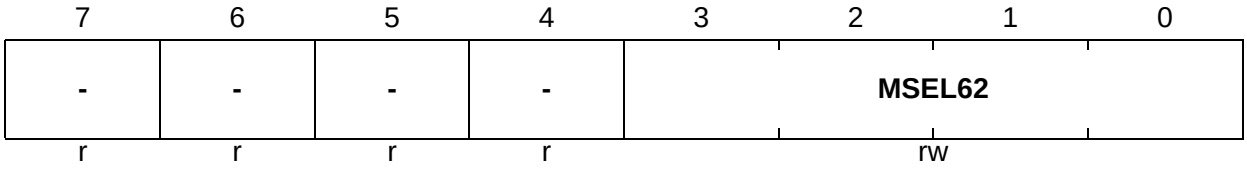


Field	Bits	Type	Description
MSEL60, MSEL61	[3:0], [7:4]	rw	**Capture/Compare Mode Selection** These bitfields select the operating mode of the three timer T12 capture/compare channels. Each channel (n=0, 1, 2) can be programmed individually either for compare or capture operation according to: 0000 Compare outputs disabled, pins CC6n and COUT6n can be used for IO. No capture action. 0001 Compare output on pin CC6n, pin COUT6n can be used for IO. No capture action. 0010 Compare output on pin COUT6n, pin CC6n can be used for IO. No capture action. 0011 Compare output on pins COUT6n and CC6n. 01XX Double-Register Capture modes, see **Table 4-4**. 1000 Hall Sensor mode, see **Table 4-5**. In order to enable the hall edge detection, all three MSEL6x have to be programmed to Hall Sensor mode. 1001 Hysteresis-like mode, see **Table 4-5**. 101X Multi-Input Capture modes, see **Table 4-6**. 11XX Multi-Input Capture modes, see **Table 4-6**.

Note: In the capture modes, all edges at the CC6x inputs are leading to the setting of the corresponding interrupt status flags in register IS. In order to monitor the selected capture events at the CCPOSx inputs in the multi-input capture modes, the CC6xST bits of the corresponding channel are set when detecting the selected event. The interrupt status bits and the CC6xST bits have to be reset by SW.

T12MSELH

T12 Capture/Compare Mode Select Register, High Byte [Reset value: 00_H]



Field	Bits	Type	Description
MSEL62	[3:0]	rw	**Capture/Compare Mode Selection** These bitfields select the operating mode of the three timer T12 capture/compare channels. Each channel (n=0, 1, 2) can be programmed individually either for compare or capture operation according to: 0000 Compare outputs disabled, pins CC6n and COUT6n can be used for IO. No capture action. 0001 Compare output on pin CC6n, pin COUT6n can be used for IO. No capture action. 0010 Compare output on pin COUT6n, pin CC6n can be used for IO. No capture action. 0011 Compare output on pins COUT6n and CC6n. 01XX Double-Register Capture modes, see **Table 4-4**. 1000 Hall Sensor mode, see **Table 4-5**. In order to enable the hall edge detection, all three MSEL6x have to be programmed to Hall Sensor mode. 1001 Hysteresis-like mode, see **Table 4-5**. 101X Multi-Input Capture modes, see **Table 4-6**. 11XX Multi-Input Capture modes, see **Table 4-6**.
-	[7:4]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

Note: In the capture modes, all edges at the CC6x inputs are leading to the setting of the corresponding interrupt status flags in register IS. In order to monitor the selected capture events at the CCPOSx inputs in the multi-input capture modes, the CC6xST bits of the corresponding channel are set when detecting the selected event. The interrupt status bits and the CC6xST bits have to be reset by SW.

Table 4-4 Description of the Double-Register Capture modes.

Description
Double-Register Capture modes
0100The contents of T12 is stored in CC6nR after a rising edge and in CC6nSR after a falling edge on the input pin CC6n.
0101The value stored in CC6nSR is copied to CC6nR after a rising edge on the input pin CC6n. The actual timer value of T12 is simultaneously stored in the shadow register CC6nSR. This feature is useful for time measurements between consecutive rising edges on pins CC6n. COUT6n is IO.
0110The value stored in CC6nSR is copied to CC6nR after a falling edge on the input pin CC6n. The actual timer value of T12 is simultaneously stored in the shadow register CC6nSR. This feature is useful for time measurements between consecutive falling edges on pins CC6n. COUT6n is IO.
0111The value stored in CC6nSR is copied to CC6nR after any edge on the input pin CC6n. The actual timer value of T12 is simultaneously stored in the shadow register CC6nSR. This feature is useful for time measurements between consecutive edges on pins CC6n. COUT6n is IO.

Table 4-5 Description of the Combined-T12 modes.

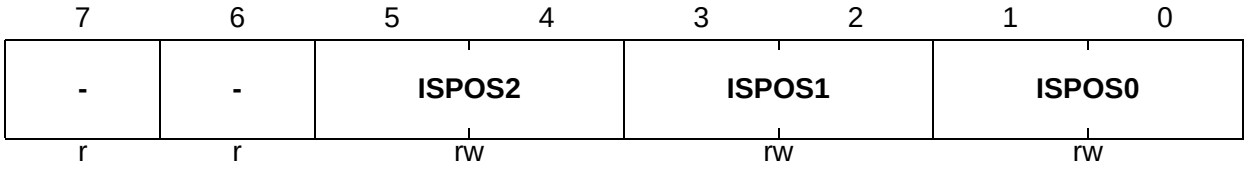
Description
Combined-T12 modes
1000Hall Sensor mode: Capture mode for channel 0, compare mode for channels 1 and 2. The contents of T12 is captured into CC60 at a valid hall event (which is a reference to the actual speed). CC61 can be used for a phase delay function between hall event and output switching. CC62 can act as a time-out trigger if the expected hall event comes too late. The value ’1000’ has to be programmed to MSEL0, MSEL1 and MSEL2 if the hall signals are used. In this mode, the contents of timer T12 is captured in CC60 and T12 is reset after the detection of a valid hall event. In order to avoid noise effects, the dead-time counter channel 0 is started after an edge has been detected at the hall inputs. When reaching the value of ’000001’, the hall inputs are sampled and the pattern comparison is done.
1001Hysteresis-like control mode with dead time generation: The negative edge of the CCPOSx input signal is used to reset bit CC6nST. As a result, the output signals can be switched to passive state immediately and switch back to active state (with dead time) if the CCPOSx is high and the bit CC6nST is set by a compare event.

Table 4-6 Description of the Multi-Input Capture modes.

Description
Multi-Input Capture modes
1010The timer value of T12 is stored in CC6nR after a rising edge at the input pin CC6n. The timer value of T12 is stored in CC6nSR after a falling edge at the input pin CCPOSx.
1011The timer value of T12 is stored in CC6nR after a falling edge at the input pin CC6n. The timer value of T12 is stored in CC6nSR after a rising edge at the input pin CCPOSx.
1100The timer value of T12 is stored in CC6nR after a rising edge at the input pin CC6n. The timer value of T12 is stored in CC6nSR after a rising edge at the input pin CCPOSx.
1101The timer value of T12 is stored in CC6nR after a falling edge at the input pin CC6n. The timer value of T12 is stored in CC6nSR after a falling edge at the input pin CCPOSx.
1110The timer value of T12 is stored in CC6nR after any edge at the input pin CC6n. The timer value of T12 is stored in CC6nSR after any edge at the input pin CCPOSx.
1111reserved (no capture or compare action)

Register PISELH contains bitfields selecting the actual input signal for the module inputs.

PISELH
Port Input Select Register **Reset Value: [00_H]**



Field	Bits	Type	Description
ISPOS0	[1:0]	rw	**Input Select for CCPOS0** This bitfield defines the source, that is used for the CCPOS0 input signal. 00 The input for CCPOS0 is external (from P1.5/ CCPOS0/T2/INT0/AN0). 01 The input for CCPOS0 is internal (from P1.5 latch). else reserved (is held on ’1’)
ISPOS1	[3:2]	rw	**Input Select for CCPOS1** This bitfield defines thesource, that is used for the CCPOS1 input signal. 00 The input for CCPOS1 is external (from P1.6/ CCPOS1/T2EX/INT1/AN1). 01 The input for CCPOS1 is internal (from P1.6 latch). else reserved (is held on ’1’)
ISPOS2	[5:4]	rw	**Input Select for CCPOS2** This bitfield defines the source, that is used for the CCPOS2 input signal. 00 The input for CCPOS2 is external (from P1.7/ CCPOS2/INT2/AN2). 01 The input for CCPOS2 is internal (from P1.7 latch). else reserved (is held on ’1’)
0	[7:6]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

PMCON1
Peripheral Management Control Register **[Reset value: XXXXX000$_B$]**

7	6	5	4	3	2	1	0
-	-	-	-	-	CCUDIS	T2DIS	ADCDIS
r	r	r	r	r	rw	rw	rw

The functions of the shaded bits are not described here

Field	Bits	Typ	Description
CCUDIS	2	rw	**CCU6 Disable Request.** 0 : CCU6 will continue normal operation. (default) 1 : Request to disable the CCU6 is active.

PMCON2
Peripheral Management Status Register **[Reset value: XXXXX000$_B$]**

7	6	5	4	3	2	1	0
-	-	-	-	-	CCUST	T2ST	ADCST
r	r	r	r	r	rh	rh	rh

The functions of the shaded bits are not described here

Field	Bits	Typ	Description
CCUST	2	rh	**CCU6 Disable Status** 0 : CCU6 is not disabled. (default) 1 : CCU6 is disabled, clock is gated off.

4.9 Serial Interface

The serial port of the C868 is full duplex, meaning it can transmit and receive simultaneously. It is also receive-buffered, meaning it can commence reception of a second byte before a previously received byte has been read from the receive register (however, if the first byte still hasn't been read by the time reception of the second byte is complete, one of the bytes will be lost). The serial port receive and transmit registers are both accessed at special function register SBUF. Writing to SBUF loads the transmit register, and reading SBUF accesses a physically separate receive register.

The serial port can operate in 3 asynchronous modes. The baud rate clock for the serial port is derived from the oscillator frequency (mode 2) or generated either by timer 1 or by a dedicated baud rate generator (mode 1, 3). Mode 0 is reserved.

Mode 1, 8-Bit UART, Variable Baud Rate:

In mode 1, ten bits are transmitted (through TxD) or received (through RxD). They are a start bit (0), 8 data bits (LSB first), and a stop bit (1). On receive, the stop bit goes into RB8 in special function register SCON. The baud rate is variable. (See section 4.9.2 for more detailed information)

Mode 2, 9-Bit UART, Fixed Baud Rate:

11 bits are transmitted (through TxD) or received (through RxD): a start bit (0), 8 data bits (LSB first), a programmable 9th bit, and a stop bit (1). On transmit, the 9th data bit (TB8 in SCON) can be assigned to the value of 0 or 1. Or, for example, the parity bit (P, in the PSW) could be moved into TB8. On receive, the 9th data bit goes into RB8 in special function register SCON, while the stop bit is ignored. The baud rate is programmable to either 1/32 or 1/64 of the oscillator frequency. (See section 4.9.3 for more detailed information)

Mode 3, 9-Bit UART, Variable Baud Rate:

11 bits are transmitted (through TxD) or received (through RxD): a start bit (0), 8 data bits (LSB first), a programmable 9th data bit, and a stop bit (1). In fact, mode 3 is the same as mode 2 in all respects except the baud rate. The baud rate in mode 3 is variable. (See section 4.9.3 for more detailed information)

In all modes, transmission is initiated by any instruction that uses SBUF as a destination register. Reception is initiated in the modes by the incomming start bit if REN = 1.

The serial interface also provides interrupt requests when transmission or reception of a frames have been completed. The corresponding interrupt request flags are TI or RI, resp. See chapter 7 of this user manual for more details about the interrupt structure. The interrupt request flags TI and RI can also be used for polling the serial interface, if the serial interrupt is not to be used (i.e. serial interrupt not enabled).

Multiprocessor Communication

Modes 2 and 3 have a special provision for multiprocessor communications. In these modes, 9 data bits are received. The 9th one goes into RB8. Then comes a stop bit. The port can be programmed such that when the stop bit is received, the serial port interrupt will be activated only if RB8 = 1. This feature is enabled by setting bit SM2 in SCON. A way to use this feature in multiprocessor systems is as follows.

When the master processor wants to transmit a block of data to one of several slaves, it first sends out an address byte which identifies the target slave. An address byte differs from a data byte in that the 9th bit is 1 in an address byte and 0 in a data byte. With SM2 = 1, no slave will be interrupted by a data byte. An address byte, however, will interrupt all slaves, so that each slave can examine the received byte and see if it is being addressed. The addressed slave will clear its SM2 bit and prepare to receive the data bytes that will be coming. The slaves that weren't being addressed leave their SM2s set and go on about their business, ignoring the incoming data bytes.

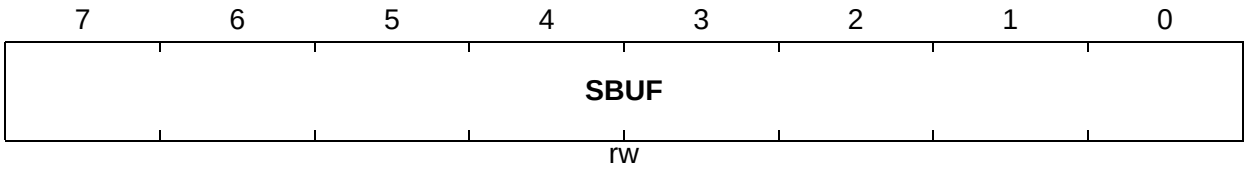
SM2 can be used in mode 1 to check the validity of the stop bit. In a mode 1 reception, if SM2 = 1, the receive interrupt will not be activated unless a valid stop bit is received.

Serial Port Registers

The serial port control and status register is the special function register SCON. This register contains not only the mode selection bits, but also the 9th data bit for transmit and receive (TB8 and RB8), and the serial port interrupt bits (TI and RI).

SBUF is the receive and transmit buffer of serial interface. Writing to SBUF loads the transmit register and initiates transmission. Reading out SBUF accesses a physically separate receive register.

SBUF
Serial Data Buffer Register **[Reset value: 00_H]**



Field	Bits	Typ	Description
SBUF	[7:0]	rw	**Serial Interface Buffer Register**

SCON
Serial Channel Control Register [Reset value: 00_H]

$9F_H$	$9E_H$	$9D_H$	$9C_H$	$9B_H$	$9A_H$	99_H	98_H
SM0	**SM1**	**SM2**	**REN**	**TB8**	**RB8**	**TI**	**RI**
rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Typ	Description
RI	0	rw	**Serial port receiver interrupt flag** RI is set by hardware at the end of the 8th bit time in mode 0, or halfway through the stop bit time in the other modes, in any serial reception (exception see SM2). RI must be cleared by software.
TI	1	rw	**Serial port transmitter interrupt flag** TI is set by hardware at the end of the 8th bit time in mode 0, or at the beginning of the stop bit in the other modes, in any serial transmission. TI must be cleared by software.
RB8	2	rw	**Serial port receiver bit 9** In modes 2 and 3, RB8 is the 9th data bit that was received. In mode 1, if SM2 = 0, RB8 is the stop bit that was received. In mode 0, RB8 is not used.
TB8	3	rw	**Serial port transmitter bit 9** TB8 is the 9th data bit that will be transmitted in modes 2 and 3. Set or cleared by software as desired.
REN	4	rw	**Enable receiver of serial port** Enables serial reception. Set by software to enable serial reception. Cleared by software to disable serial reception.
SM2	5	rw	**Enable serial port multiprocessor communication in modes 2 and 3** In mode 2 or 3, if SM2 is set to 1 then RI will not be activated if the received 9th data bit (RB8) is 0. In mode 1, if SM2 = 1 then RI will not be activated if a valid stop bit was not received. In mode 0, SM2 should be 0.

Field	Bits	Typ	Description
SM0 **SM1**	[7:6]	rw	**Serial port 0 operating mode selection bits** **Table 1 :**

SM0	SM1	Selected operating mode
0	0	mode 0 :reserved
0	1	mode 1 :8-bit UART, variable baud rate
1	0	mode 2 :9-bit UART, fixed baud rate (f_{sys}/32 or f_{sys}/64)
1	1	Mode 3 :9-bit UART, variable baud rate

4.9.1 Baud Rate Generation

There are several possibilities to generate the baud rate clock for the serial port depending on the mode in which it is operating.

For clarification, some terms regarding the difference between "baud rate clock" and "baud rate" should be mentioned. The serial interface requires a clock rate which is 16 times the baud rate for internal synchronization. Therefore, the baud rate generators have to provide a "baud rate clock" to the serial interface which - there divided by 16 - results in the actual "baud rate". However, all formulas given in the following section already include the factor and calculate the final baud rate. Further, the abrevation f_{SYS} refers to the system frequency.

The baud rate of the serial port is controlled by a bit which are located in the special function registers as shown below.

PCON
Power Control Register **[Reset value: 0XXX0000$_B$]**

7	6	5	4	3	2	1	0
SMOD	**-**	**-**	**SD**	**GF1**	**GF0**	**PDE**	**IDLE**
rw	r	r	rw	rw	rw	rw	rw

The functions of the shaded bits are not described here

Field	Bits	Typ	Description
SMOD	7	rw	**Double baud rate** When set, the baud rate of serial interface in modes 1, 2, 3 is doubled. After reset this bit is cleared.

Depending on the programmed operating mode different paths are selected for the baud rate clock generation.

4.9.1.1 Baud Rate in Mode 2

The baud rate in mode 2 depends on the value of bit SMOD in special function register PCON. If SMOD = 0 (which is the value after reset), the baud rate is 1/64 of the system frequency. If SMOD = 1, the baud rate is 1/32 of the system frequency.

$$\text{Mode 2 baud rate} = \frac{2^{\text{SMOD}}}{64} \times \text{system frequency}$$

4.9.1.2 Baud Rate in Mode 1 and 3

In these modes the baud rate is variable and can be generated alternatively by a baud rate generator or by timer 1.

Using the Timer 2 as Baud Rate Generator

In modes 1 and 3, the C868 can use timer 2 as the baud rate generator for the serial port. To enable the baud generator for transmit, bit TCLK (bit 4 of special function register

T2CON) must be set. To enable the baud generator for receive, bit RCLK (bit 5 of special function register T2CON) must be set.

With the timer 2 as clock source for the serial port in mode 1 and 3, the baud rate can be determined as follows:

$$\text{Mode 1, 3 baud rate} = \frac{2^{SMOD}}{32} \times \text{(timer 2 overflow rate)}$$

$$\text{Timer 2 overflow rate} = \text{system frequency} / 2x(2^{16} - RC2)$$

with RC2 = RC2H.7..0, RC2L.7..0 and timer2 count direction is set to up.

Using Timer 1 to Generate Baud Rates

In modes 1 and 3 of the serial interface timer 1 can also be used for generating baud rates. Then the baud rate is determined by the timer 1 overflow rate and the value of SMOD as follows:

$$\text{Mode 1, 3 baud rate} = \frac{2^{SMOD}}{32} \times \text{(timer 1 overflow rate)}$$

The timer 1 interrupt is usually disabled in this application. Timer 1 itself can be configured for either "timer" or "counter" operation, and in any of its operating modes. In most typical applications, it is configured for "timer" operation in the auto-reload mode (high nibble of TMOD = 0010_B). In this case the baud rate is given by the formula:

$$\text{Mode 1, 3 baud rate} = \frac{2^{SMOD} \times \text{system frequency}}{32 \times 12 \times (256 - (TH1))}$$

Very low baud rates can be achieved with timer 1 if leaving the timer 1 interrupt enabled, configuring the timer to run as 16-bit timer (high nibble of TMOD = 0001_B), and using the timer 1 interrupt for a 16-bit software reload.

4.9.2 Details about Mode 1

Ten bits are transmitted (through TxD), or received (through RxD): a start bit (0), 8 data bits (LSB first), and a stop bit (1). On reception, the stop bit goes into RB8 in SCON. The baud rate is determined either by the timer 1 overflow rate or by the internal baud rate generator.

Figure 4-37 shows a simplified functional diagram of the serial port in mode 1. The associated timings for transmit/receive are illustrated in **Figure 4-38**.

Transmission is initiated by an instruction that uses SBUF as a destination register. The "Write-to-SBUF" signal also loads a 1 into the 9th bit position of the transmit shift register and flags the TX control unit that a transmission is requested. Transmission starts at the next rollover in the divide-by-16 counter. (Thus, the bit times are synchronized to the divide-by-16 counter, not to the "Write-to-SBUF" signal).

The transmission begins with activation of $\overline{\text{SEND}}$, which puts the start bit at TxD. One bit time later, DATA is activated, which enables the output bit of the transmit shift register to TxD. The first shift pulse occurs one bit time after that.

As data bits shift out to the right, zeroes are clocked in from the left. When the MSB of the data byte is at the output position of the shift register, then the 1 that was initially loaded into the 9th position is just to the left of the MSB, and all positions to the left of that contain zeroes. This condition flags the TX control unit to do one last shift and then deactivate $\overline{\text{SEND}}$ and set TI. This occurs at the 10th divide-by-16 rollover after "Write-to-SBUF".

Reception is initiated by a detected 1-to-0 transition at RxD. For this purpose RxD is sampled at a rate of 16 times whatever baud rate has been established. When a transition is detected, the divide-by-16 counter is immediately reset, and $1FF_H$ is written into the input shift register, and reception of the rest of the frame will proceed.

The 16 states of the counter divide each bit time into 16ths. At the 7th, 8th and 9th counter states of each bit time, the bit detector samples the value of RxD. The value accepted is the value that was seen in at least 2 of the 3 samples. This is done for the noise rejection. If the value accepted during the first bit time is not 0, the receive circuits are reset and the unit goes back to looking for another 1-to-0 transition. This is to provide rejection or false start bits. If the start bit proves valid, it is shifted into the input shift register, and reception of the rest of the frame will proceed.

As data bits come in from the right, 1s shift out to the left. When the start bit arrives at the leftmost position in the shift register, (which in mode 1 is a 9-bit register), it flags the RX control block to do one last shift, load SBUF and RB8, and set RI. The signal to load SBUF and RB8, and to set RI, will be generated if, and only if, the following conditions are met at the time the final shift pulse is generated.

1)RI = 0,and
2)either SM2 = 0, or the received stop bit = 1

If one of these two condtions is not met, the received frame is irretrievably lost. If both conditions are met, the stop bit goes into RB8, the 8 data bit goes into SBUF, and RI is activated. At this time, whether the above conditions are met or not, the unit goes back to looking for a 1-to-0 transition in RxD.

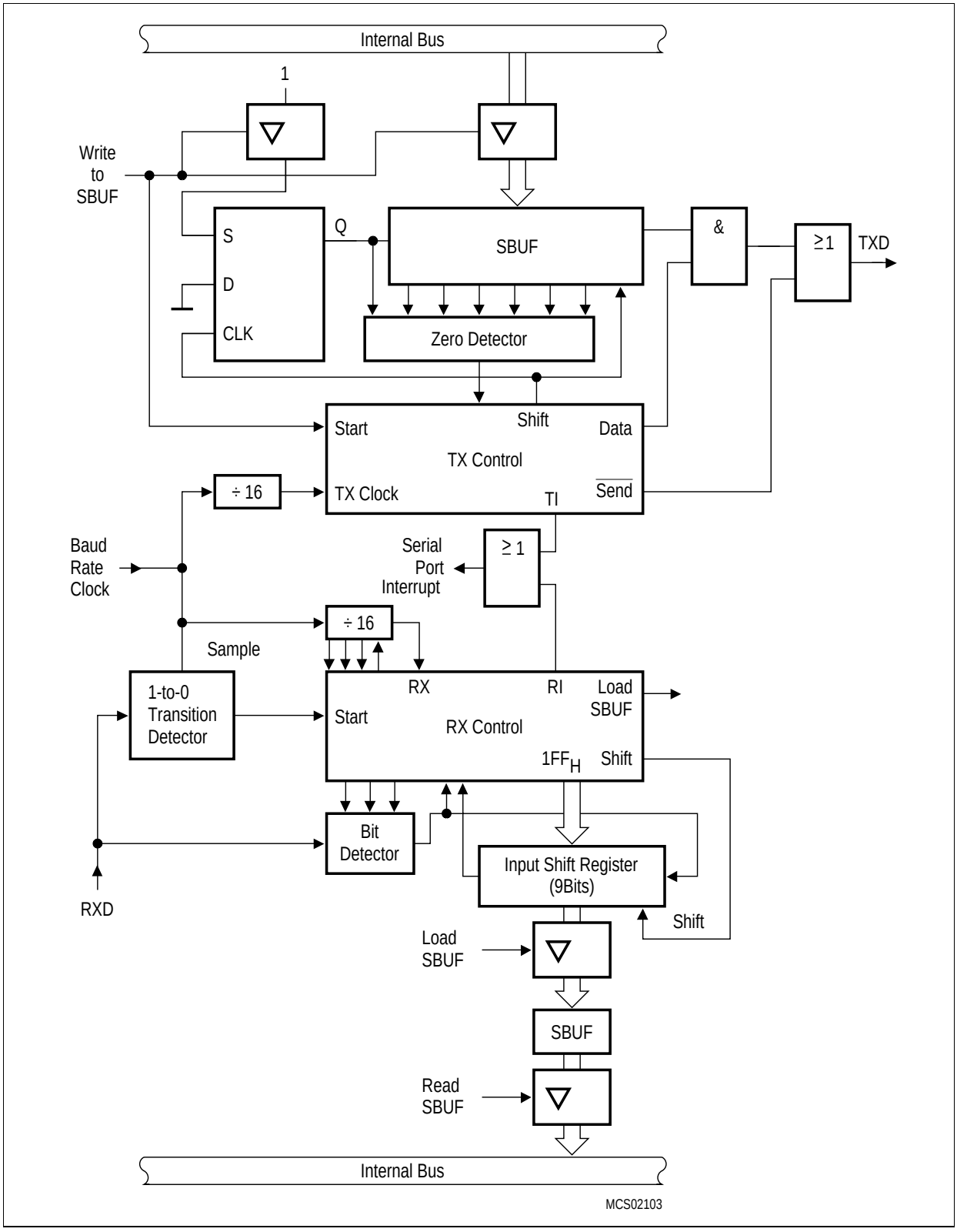


Figure 4-37 Serial Interface, Mode 1, Functional Diagram

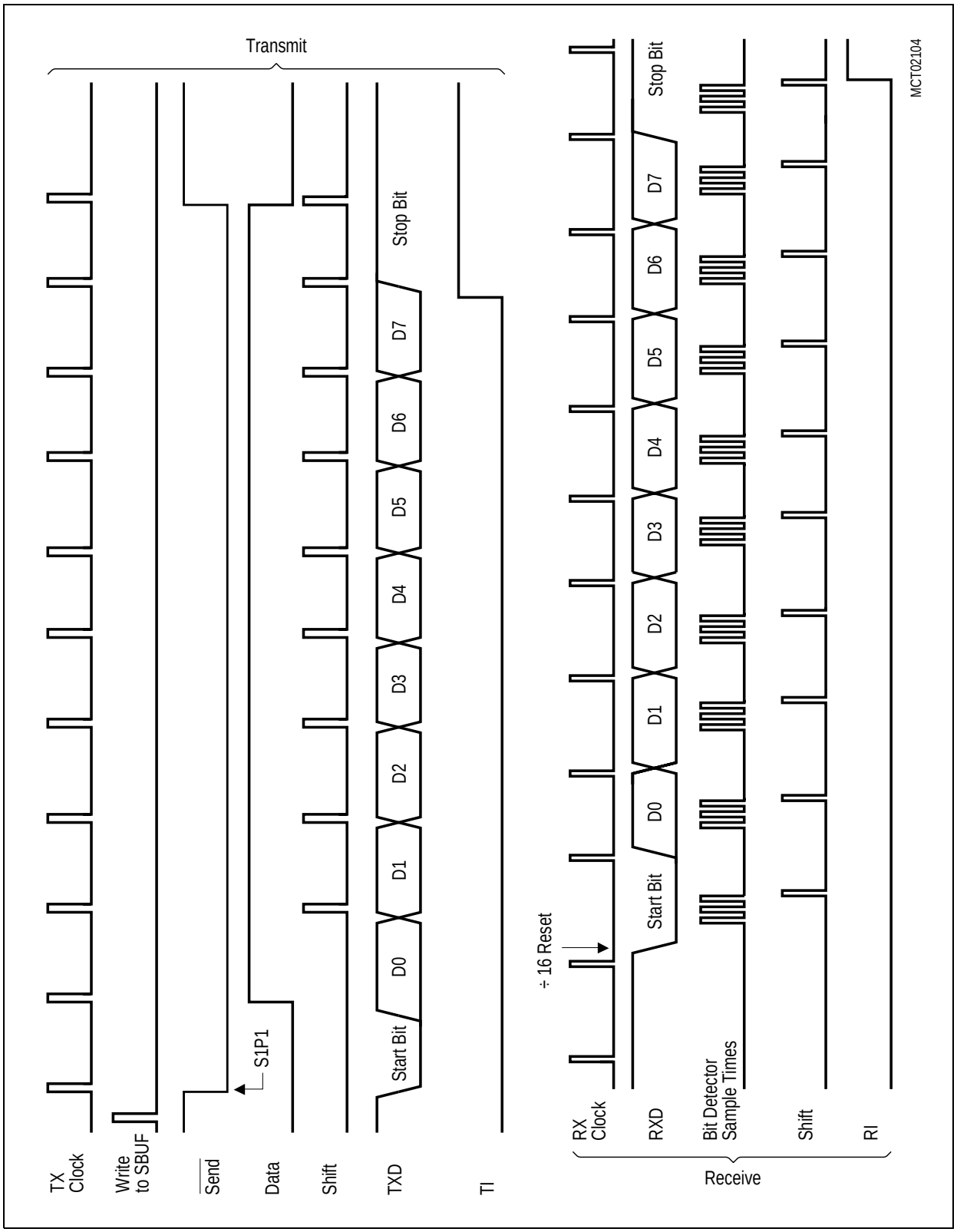


Figure 4-38 Serial Interface, Mode 1, Timing Diagram

4.9.3 Details about Modes 2 and 3

Eleven bits are transmitted (through TxD), or received (through RxD): a start bit (0), 8 data bits (LSB first), a programmable 9th data bit, and a stop bit (1). On transmission, the 9th data bit (TB8) can be assigned the value of 0 or 1. On reception, the 9th data bit goes into RB8 in SCON. The baud rate is programmable to either 1/32 or 1/64 the system frequency in mode 2 (When bit SMOD in SFR PCON.7 is set, the baud rate is $f_{SYS}/32$). In mode 3 the baud rate clock is generated by timer 1, which is incremented by a rate of $f_{SYS}/12$ or by the internal baud rate generator.

Figure 4-39 shows a functional diagram of the serial port in modes 2 and 3. The receive portion is exactly the same as in mode 1. The transmit portion differs from mode 1 only in the 9th bit of the transmit shift register. The associated timings for transmit/receive are illustrated in **Figure 4-40**.

Transmission is initiated by any instruction that uses SBUF as a destination register. The "Write-to-SBUF" signal also loads TB8 into the 9th bit position of the transmit shift register and flags the TX control unit that a transmission is requested. Transmission starts at the next rollover in the divide-by-16 counter. (Thus, the bit times are synchronized to the divide-by-16 counter, not to the "Write-to-SBUF" signal.)

The transmission begins with activation of SEND, which puts the start bit at TxD. One bit time later, DATA is activated, which enables the output bit of the transmit shift register to TxD. The first shift pulse occurs one bit time after that. The first shift clocks a 1 (the stop bit) into the 9th bit position of the shift register. Thereafter, only zeroes are clocked in. Thus, as data bits shift out to the right, zeroes are clocked in from the left. When TB8 is at the output position of the shift register, then the stop bit is just to the left of TB8, and all positions to the left of that contain zeroes. This condition flags the TX control unit to do one last shift and then deactivate SEND and set TI. This occurs at the 11th divide-by-16 rollover after "Write-to-SBUF".

Reception is initiated by a detected 1-to-0 transition at RxD. For this purpose RxD is sampled at a rate of 16 times whatever baud rate has been established. When a transition is detected, the divide-by-16 counter is immediately reset, and $1FF_H$ is written to the input shift register.

At the 7th, 8th and 9th counter states of each bit time, the bit detector samples the value of RxD. The value accepted is the value that was seen in at least 2 of the 3 samples. If the value accepted during the first bit time is not 0, the receive circuits are reset and the unit goes back to looking for another 1-to-0 transition. If the start bit proves valid, it is shifted into the input shift register, and reception of the rest of the frame will proceed.

As data bit come from the right, 1s shift out to the left. When the start bit arrives at the leftmost position in the shift register (which in modes 2 and 3 is a 9-bit register), it flags the RX control block to do one last shift, load SBUF and RB8, and to set RI. The signal to load SBUF and RB8, and to set RI, will be generated if, and only if, the following conditions are met at the time the final shift pulse is generated:

1)RI = 0, and
2)Either SM2 = 0 or the received 9th data bit = 1

If either of these conditions is not met, the received frame is irretrievably lost, and RI is not set. If both conditions are met, the received 9th data bit goes into RB8, and the first 8 data bit goes into SBUF. One bit time later, whether the above conditions were met or not, the unit goes back to looking for a 1-to-0 transition at the RxD input.

Note that the value of the received stop bit is irrelevant to SBUF, RB8 or RI.

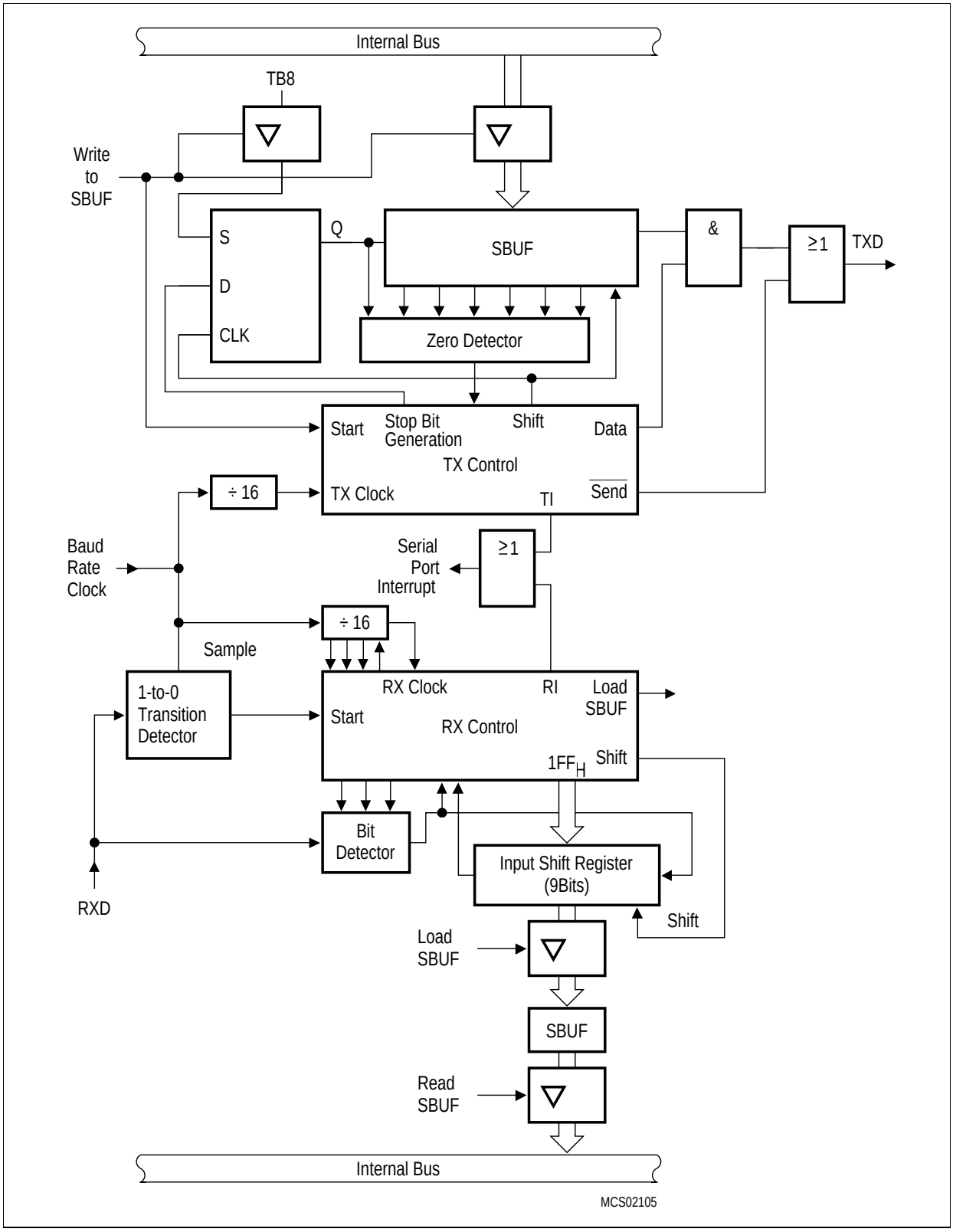


Figure 4-39 Serial Interface, Mode 2 and 3, Functional Diagram

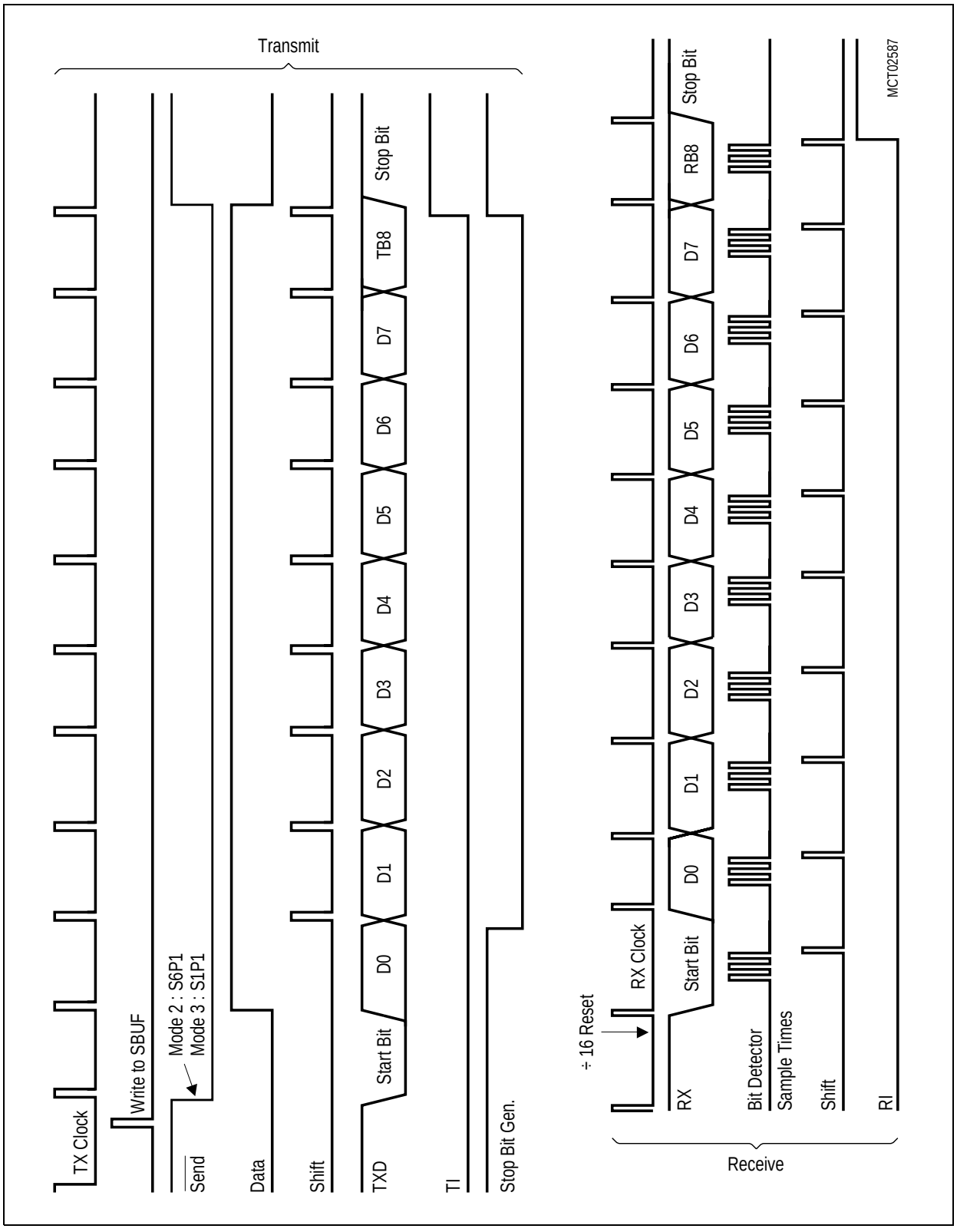


Figure 4-40 Serial Interface, Mode 2 and 3, Timing Diagram

4.10 A/D Converter

The C868 includes a high performance / high speed 8-bit A/D-Converter (ADC) with 5 analog input channels. It operates with a successive approximation technique. The A/D converter provides the following features:

- 5 multiplexed input channels, which can also be used as digital inputs
- 8-bit resolution with TUE of +/- 2 LSB8.
- Single or continuous conversion mode
- Start of conversion by software or hardware methods
- Interrupt request generation after each conversion
- Using successive approximation conversion technique via a capacitor array
- Powerdown in normal, idle and slow-down modes

4.10.1 Register Definition of the ADC

The ADCON0 and ADCON1 registers are used to configure and control the ADC. It also indicates the status of the ADC functions (flags).

ADCON0
A/D Converter Control Register 0 **[Reset value: 00000000_B]**

7	6	5	4	3	2	1	0
ADST	ADBSY	ADM		CCU-ADEX	ADCH		
rw	rh	rw		r	rw		

Field	Bits	Typ	Description
ADCH	[2:0]	rw	**Analog Input Selection.** The number of bits implemented depends on the actual number of channels required for the product. Bit 0 of the register shall be the least significant bit.
CCU-ADEX	3	rw	**A/D Conversion Start Control Source** 0: Conversion can be started by software method only(default) 1: Conversion can be started by CCU T13PM trigger. Setting T13PM flag in ISH will not trigger conversion.
ADM	[5:4]	rw	**ADC Mode Selection.** 00 : Single Conversion on Fixed Channel (default) 01 : Continuous Conversion on Fixed Channel 10 : Reserved 11 : Reserved Bit 4 is used for mode selection while bit 5 is reserved.
ADBSY	6	rh	**ADC Busy Flag** 1 : Conversion is in progress.
ADST	7	rw	**A/D Conversion Start Bit.** Set by user to begin a conversion. Cleared by hardware at the beginning of conversion. For continuous conversion, this bit is cleared at the beginning of first conversion.
-	3	r	**reserved;** returns ’0’ if read; should be written with ’0’;

ADCON1
A/D Converter Control Register 1 **[Reset value: XX000000$_B$]**

7	6	5	4	3	2	1	0
-	-	ADSTC			ADCTC		
r	r	rw			rw		

Field	Bits	Typ	Description
ADCTC	[2:0]	rw	**ADC Conversion Time Control**
ADSTC	[5:3]	rw	**ADC Sample Time Control**
-	7,6	r	**reserved;** returns ’0’ if read; should be written with ’0’;

The ADDATH register stores the result of the conversion, together with the channel number.

ADDATH
A/D Converter Data Register **[Reset value: 00$_H$]**

7	6	5	4	3	2	1	0
ADDATH							
r							

Field	Bits	Typ	Description
ADDATH	[7:0]	r	**Result of ADC conversion.**

PMCON1
Peripheral Management Control Register **[Reset value: XXXXX000$_B$]**

7	6	5	4	3	2	1	0
-	-	-	-	-	CCUDIS	T2DIS	ADCDIS
r	r	r	r	r	rw	rw	rw

The functions of the shaded bits are not described here

Field	Bits	Typ	Description
ADCDIS	0	rw	**ADC Disable Request.** 0 : ADC will continue normal operation. (default) 1 : Request to disable the ADC is active.

PMCON2

Peripheral Management Status Register **[Reset value: XXXXX000$_B$]**

7	6	5	4	3	2	1	0
-	-	-	-	-	CCUST	T2ST	ADCST
r	r	r	r	r	rh	rh	rh

The functions of the shaded bits are not described here

Field	Bits	Typ	Description
ADCST	0	rh	**ADC Disable Status** 0 : ADC is not disabled. (default) 1 : ADC is disabled, clock is gated off.

4.10.2 Operation of the ADC

The ADC supports two conversion modes - single and continuous conversions. For each mode, there are two ways in which conversion can be started - by software and by the T13PM signal from the CCU module.

Writing a ’0’ to bit CCU-ADEX select conversion control by ADST. Writing a ’1’ to bit field ADST starts conversion on the channel that is specified by ADCH. In single conversion mode, bit field ADM is cleared to ’0’. This is the default mode selected after hardware reset. When a conversion is started, the channel specified is sampled. The busy flag ADBSY is set and ADST is cleared. When the conversion is completed, the interrupt request is asserted and the 8-bit result is transferred to the result register ADDATH.

In continuous conversion mode, bit field ADM is set to ’1’. In this mode, the ADC repeatedly converts the channel specified by ADCH. Bit ADST is cleared at the beginning of the first conversion. The busy flag ADBSY is asserted until the last conversion is completed. At the end of each conversion, the interrupt request will be asserted. To stop conversion, ADM has to be reset by software. If the channel number ADCH is changed while continuous conversion is in progress, the new channel specified will be sampled in the conversions that follow.

A new request to start conversion will be allowed only after the completion of any conversion that is in progress.

Writing a ’1’ to bit CCU_ADEX select conversion control by T13PM trigger signal from the CCU module.

Note: Caution must be taken when changing conversion start source. To change conversion source from software to hardware trigger, it is best to let remaining software conversion to complete before changing. To change conversion source from hardware trigger to software, it is best to change source first, let any remaining hardware conversion to complete before beginning a software conversion.

4.10.3 Module Powerdown

The ADC is disabled when the chip goes into the powerdown mode as describe in . Or it can be individually disabled by setting ADCDIS in register PMCON1. This helps to reduce current consumption in the normal, slow down and idle modes of operation if the ADC is not utilized. Bit ADCST in register PMCON2 reflects the powerdown status of ADC. If the ADC is disabled during an A/D conversion, ADC will be disabled (ADCST=’1’) only after the conversion is completed.

Note : Generally, before entering the power-down mode, an A/D conversion in progress must be stopped. If a single A/D conversion is running, it must be terminated by polling the ADBSY bit or waiting for the A/D conversion interrupt. In continuous conversion mode, ADM must be cleared and the last A/D conversion must be terminated before entering the power-down mode.

4.11 Conversion and Sample Time Control

The conversion and sample times are programmed via the bit fields ADCTC and ADSTC respectively of the register ADCON1. Bit field ADCTC (conversion time control) selects the internal ADC clock - adc_clk. Bit field ADSTC (sample time control) selects the sample time. The data in ADCTC and ADSTC can be modified while a conversion is in progress, but will only be evaluated after the current conversion has completed. Thus the change will only affect the subsequent conversion. The internal ADC clock, adc_clk is derived from the peripheral clock f_{sys} according to :

$$adc\_clk = f_{SYS} / clock\ divider$$

The A/D conversion procedure is divided into four parts :

Synchronizing phase (t_{SYNC}), delay before actual conversion commence.

Sample phase (t_S), used for sampling the analog input voltage.

Conversion phase (t_{CO}), used for the real A/D conversion (includes calibration).

Write result phase (t_{WR}), used for writing the conversion result to the ADDATH registers.

The total A/D conversion time is defined by t_{ADCC} which is the sum of the four phase periods, t_{SYNC}, t_S ,t_{CO} and t_{WR}. T_{ADCC} is computed with the following formula:

$$t_{ADCC} = 2/f_{SYS} + t_S + 8/adc\_clk$$

The sample time t_S is configured in periods of the selected internal ADC clock. The table below lists the possible combinations.

ADCTC	Clock Divider (TVC)	ADC Basic Clock adc_clk	ADSTC	Sample Time t_S (Periods of adc_clk, STC)
000 (default)	32	f_{SYS} / 32	000 (default)	2
001	28	f_{SYS} / 28	001	4
010	24	f_{SYS} / 24	010	6
011	20	f_{SYS} / 20	011	8
100	16	f_{SYS} / 16	100	10
101	12	f_{SYS} / 12	101	12
110	8	f_{SYS} / 8	110	14
111	4	f_{SYS} / 4	111	16

Sample Time t_S :
During this time the internal capacitor array is connected to the selected analog input channel and is loaded with the analog voltage to be converted. The analog voltage is internally fed to a voltage comparator. With beginning of the sample phase the ADBSY bit in SFR ADCON0 is set.

Conversion Time t_{CO} :
During the conversion time the analog voltage is converted into a 8-bit digital value using the successive approximation technique with a binary-weighted capacitor network. At the end of the conversion time the ADBSY bit is reset and the IADC bit in SFR IRCON1 is set indicating an A/D converter interrupt condition.

Write Result Time t_{WR} :
At the result phase the conversion result is written into the ADDATH registers.
A/D Conversion Timing in Relation to Processor Cycles

Depending on the application, typically there are three methods to handle the A/D conversion in the C868.

Software delay
Using the software method, the machine cycles of the A/D conversion are counted and the program executes a software delay (e.g. NOPs) before reading the A/D conversion result in the write result cycle. This is the fastest method to get the result of an A/D conversion.

Polling ADBSY bit
Using the software method, the ADBSY bit is polled and the program waits until ADBSY=0. Attention : a polling JB instruction which is two machine cycles long, possibly may not recognize the ADBSY=0 condition during the write result cycle in the continuous conversion mode.

A/D conversion interrupt
Using the software or hardware methods, after the start of an A/D conversion the A/D converter interrupt is enabled. The result of the A/D conversion is read in the interrupt service routine. If other C868 interrupts are enabled, the interrupt latency must be regarded. Therefore, this software method is the slowest method to get the result of an A/D conversion.

C868

5 Reset, Brownout and System Clock Operation

5.1 Hardware Reset Operation

The hardware reset function incorporated in the C868 allows for an easy automatic start-up at a minimum of additional hardware and forces the controller to a predefined default state. The hardware reset function can also be used during normal operation in order to restart the device. This is particularly done when the power-down mode is to be terminated.

Additional to the hardware reset, which is applied externally to the C868, there are three internal reset sources, the watchdog timer, the brownout and the PLL. This chapter deals only with the external hardware reset and brownout.

The reset input is an active low input. An internal Schmitt trigger is used at the input for noise rejection. The $\overline{\text{RESET}}$ pin must be held low for at least tbd usec. But the CPU will only exit from reset condition after the PLL lock had been detected.

During $\overline{\text{RESET}}$ at transition from low to high, C868 will go into normal mode if ALE/BSL is high and bootstrap loading mode if ALE/BSL is low. A pullup or pulldown to V_{DDP} is recommended for pin ALE/BSL depending on the intended chipmode because when reset is exited, ALE/BSL is set to output by default. TxD should have a pullup to V_{DDP} and should not be stimulated externally during reset, as a logic low at this pin will cause the chip to go into test mode if ALE/BSL is low.

At the $\overline{\text{RESET}}$ pin, a pullup resistor is connected to V_{DDP} and a capacitor is connected to ground to allow a power-up reset. After V_{DDP} has been turned on, the capacitor must hold the voltage level at the reset pin for a specific time to effect a complete reset.

The time required for a reset operation must be at least tbd - tbd usec. The same considerations apply if the reset signal is generated externally (**Figure 5-1 b**). In each case it must be assured that the logic at ALE/BSL and TxD are latched properly.

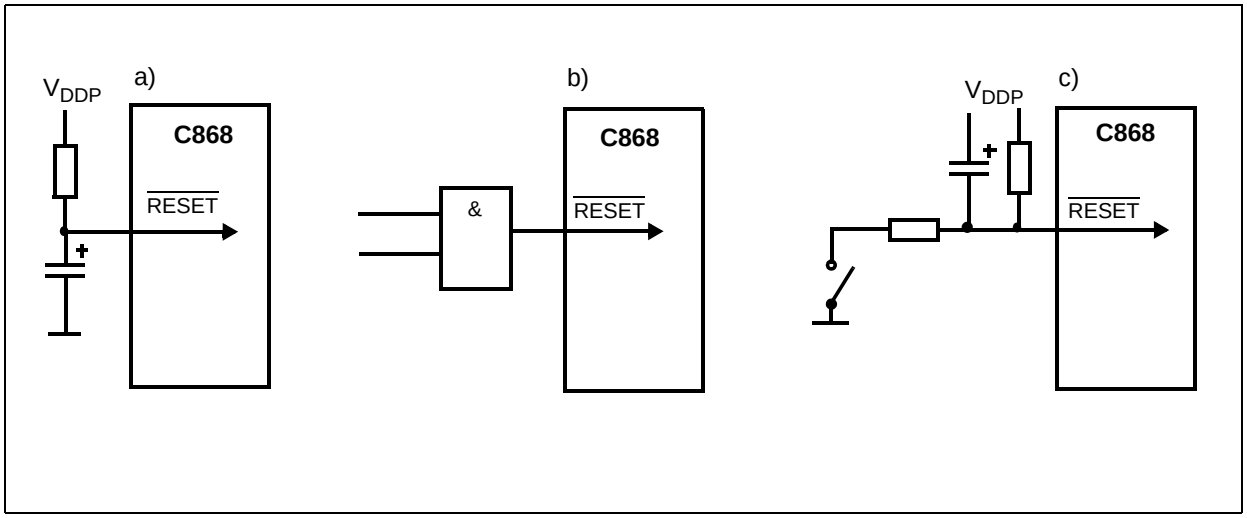


Figure 5-1 Reset Circuitries

A correct reset leaves the processor in a defined state. The program execution starts at location 0000_H. After reset is internally accomplished the port latches of ports 1 and 3 defaulted to FF_H, and they are set to input.

The contents of the internal RAM and XRAM of the C868 are not affected by a reset. After power-up the contents are undefined, while it remains unchanged during a reset if the power supply is not turned off.

5.2 Internal Reset after Power-On

Figure 5-2 shows the power-on sequence.

For the C868, the device enter into default reset state once $\overline{\text{RESET}}$ has gone low with all I/O ports set to input or high impedance. The internal reset is released only after the PLL has locked. In (**Figure 5-2**,II) the internal reset remains active even after the $\overline{\text{RESET}}$ pin had gone high, the I/O ports 1 and 3 remain as input. In (**Figure 5-2**,III), detection for continuous PLL lock is done before internal reset is released. The 4096 cycles of continuous lock detection ensures that a reset due to PLL unlock will not happen during the transient period after the PLL started functioning. After continuous PLL lock is detected, the C868 starts operation.(**Figure 5-2**,IV)

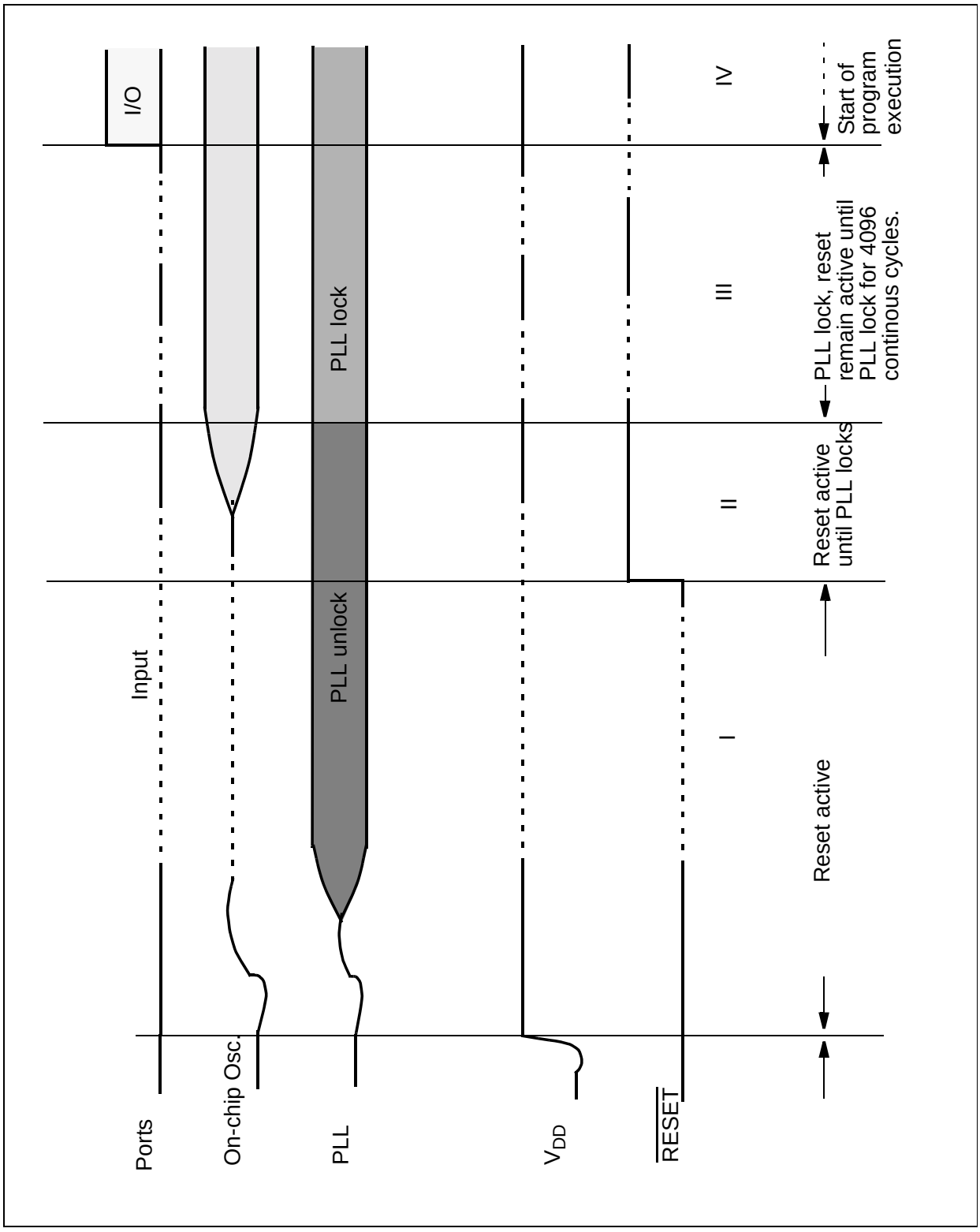


Figure 5-2 Power-On Reset of the C868

5.3 Brownout

An on-chip analog circuit detects brownout, if the voltage V_{DDC} dips below the threshold voltage $V_{THRESHOLD}$ momentarily while RESET pin is high. If this detection is active for tbd usec then the device will reset. When supply voltage V_{DDC} recovers by exceeding $V_{THRESHOLD}$ while RESET is high, the reset is released once PLL is locked for 4096 clocks. Bit BO in the PMCON0 register is set when brownout detected if brownout detection was enabled, this bit is cleared by hardware reset RESET and software. All ports are tristated during brownout.

The $V_{THRESHOLD}$ has a nominal value of 1.47V, a minimum value of 1.1V and a maximum value of 1.8V.

PMCON0
Wake-up Control Register **[Reset value: $XXX000000_B$]**

7	6	5	4	3	2	1	0
-	-	-	EBO	BO	SDSTAT	WS	EWPD
r	r	r	rw	rw	rh	rw	rw

The functions of the shaded bits are not described here

Field	Bits	Typ	Description
BO	3	rw	**Brownout Status Bit.** 0 : Brownout not detected. 1 : Brownout detected before the last power on if EBO was set before the occurence of brownout.. This bit is set by hardware only, it is cleared by hardware reset and software.
EBO	4	rw	**Enable Brownout detect.** 0: Brownout module is disabled. Occurence of brownout will not cause an internal reset. 1: Occurence of brownout will cause an internal reset and BO will be set.
-	[7:5]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

5.4 Clock Generation

The top-level view of the system clock generation of the C868 is shown in **Figure 5-3**.

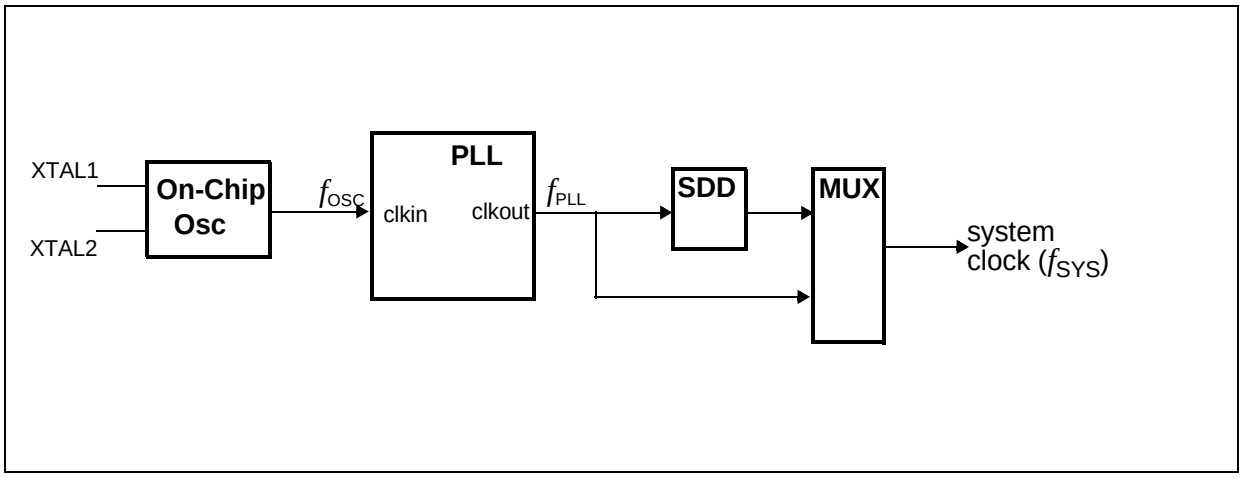


Figure 5-3 Block Diagram of the Clock Generation

5.5 PLL Operation

The PLL consists of a voltage controlled oscillator (VCO) with a feedback path. A divider in the feedback path divides the VCO frequency down. The resulting frequency is then compared to the externally applied frequency. The phase detection logic determines the difference between the two clock signals and accordingly controls the frequency of the VCO. During start-up, the VCO increases its frequency until the divided feedback clock matches the external clock frequency. A lock detection logic monitors and signals this condition. The phase detection logic continues to monitor the two clock signals and adjusts the VCO clock if required.

The PLL provides mechanisms to detect a failure of the external clock and to bring the C868 into a safe state in such a case. When the PLL loses the lock to the external clock, either due to a break of the crystal or an external line, it generate an internal reset. The PLLR flag in the SCUWDT register is set, this flag can only be reset by a hardware reset or by software.

Due to this operation, the VCO clock of the PLL has a frequency which is a multiple of the externally applied clock. The factor for this is controlled through the value applied to the divider in the feedback path. That is why this factor is often called a multiplier, although it actually controls a divider. This parameter called the feedback divider has a fixed value N = 15.

When software power down mode is entered, the PLL is powered down.

5.5.1 VCO Frequency Ranges

The frequency range for f_{VCO} is:

$$100\ MHz \leq f_{VCO} \leq 160\ MHz \quad [5\text{-}1]$$

5.5.2 K-Divider

The K-Divider is a software controlled divider. The bit field KDIV is provided in register CMCON. Software can write to this field in order to change the PLL frequency f_{PLL}. The default KDIV value is 4. **Table 5-2** lists the possible values for KDIV and the resulting division factor.

The divider is designed such that a synchronous switching of the clock is performed without spurious or shortened clock pulses when software changes the divider factor KDIV. However, special attention has to be paid concerning the effect of such a clock change to the various modules in the system.

5.5.3 Determining the PLL Clock Frequency

This section gives the formulas for the determination of the PLL clock frequency. In PLL operation, the PLL clock is derived from the VCO frequency f_{VCO} divided by the K-factor. f_{VCO} is generated from the external clock multiplied by 15.

The PLL clock frequency f_{PLL} can be made proportional to the ratio 15 / K, where bit field CMCON.KDIV determines the clock scale factor K. The VCO output frequency is determined by:

$$f_{VCO} = 15 \times f_{OSC} \quad [5.2]$$

and the resulting PLL clock is determined by:

$$f_{PLL} = f_{VCO} / K = \frac{15}{K} \times f_{OSC} \quad [5.3]$$

Since stable operation of the VCO is only guaranteed if f_{VCO} remains inside of the defined frequency range for the VCO (see **Equation [5-1]**), the external frequency f_{OSC} is also confined to certain ranges. **Table 5-1** list the range.

Table 5-1 Input Frequencies and N Factor=15 for f_{VCO}

f_{VCO} = 100 MHz	f_{VCO} = 160 MHz
6.67	10.67

Table 5-2 Output Frequencies f_{PLL} Derived from Various Output Factors

K-Factor		f_{PLL}		Duty Cycle [%]	Jitter
Selected Factor	KDIV	f_{VCO} = 100 MHz	f_{VCO} = 160 MHz		
2	000_B	50	80	50	linear depending on f_{VCO} at f_{VCO} =100MHz: +/-300ps at f_{VCO} =160MHz: +/-250ps additional jitter for odd Kdiv factors tbd.
4	010_B	25	40	50	
5[1)]	011_B	20	32	40	
6	100_B	16.67	26.67	50	
8	101_B	12.5	20	50	
9[1)]	110_B	11.11	17.78	44	
10	111_B	10	16	50	
16	001_B	6.25	10	50	

1) These odd factors should not be used (not tested because off the unsymmetrical duty cycle).

2) Shaded combinations should not be used because they are above the maximum CPU frequency of 40MHz.

5.6 Slow Down Operation

The programmable Slow Down Divider (SDD) divides the PLL output clock frequency by a factor of 1...32 which is specified via CMCON.REL. When CMCON.REL is written during SDD operation the reload counter will output one more clock pulse with the ‘old’ frequency in order to synchronize it internally before generating the ‘new’ frequency.

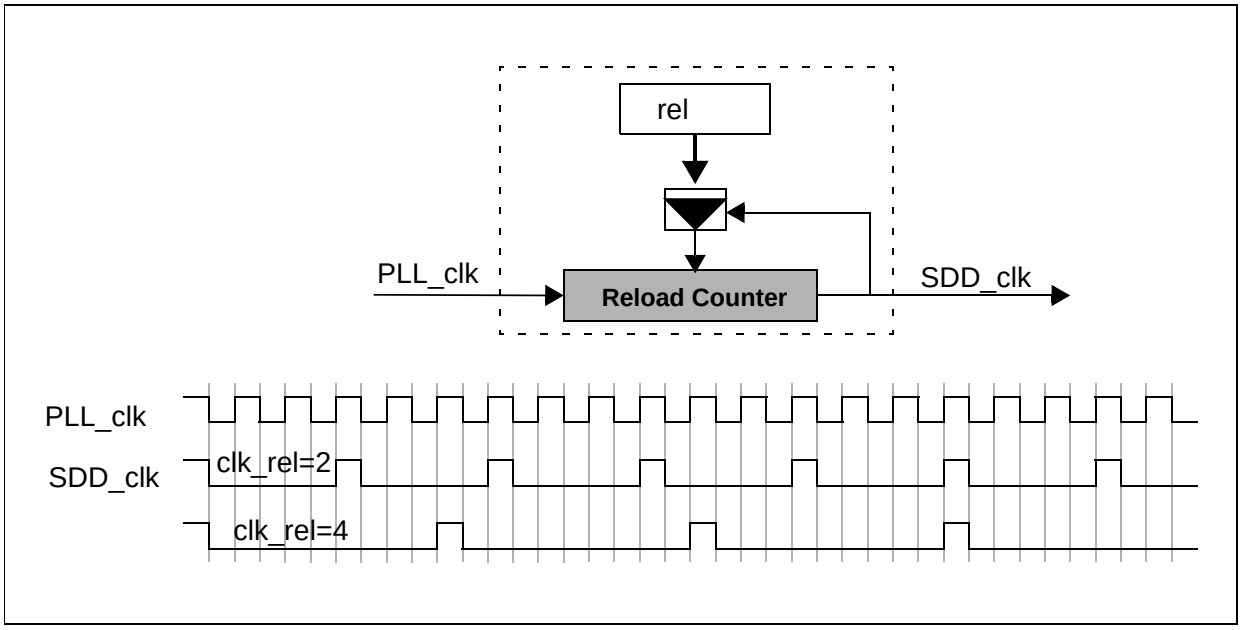


Figure 5-4 Slow Down Divider Operation

SDD_clk = PLL_clk / (CMCON.REL_B +1)

For a 20 MHz basic clock the on-chip logic may be run at a frequency down to 625 KHz without an external hardware change. During Slow Down operation the whole device (including bus interface) is clocked with the symmetrical SDD clock (see figure above).

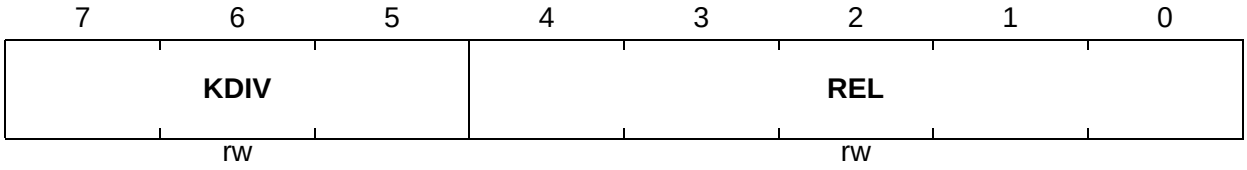
5.6.1 Switching Between PLL Clock and SDD Clock

Switching Control logic controls the switching mechanism itself and ensures a continuous and glitch-free clock signal to the on-chip logic.

Note: When switch from slow down mode to PLL operation (if configured), Master clock will be switched to PLL clock only after PLL (pll_locked) is locked.

Switching to Slow Down operation affects frequency sensitive peripherals like serial interfaces, timers, PWM, etc.If these units are to be operated in Slow Down mode their Prescalers or reload values must be adapted. Please note that the reduced CPU frequency decreases e.g. timer resolution and increases the step width e.g. for baudrate generation. The basic clock frequency in such a case should be chosen to accommodate the required resolutions and/or baudrates.

CMCON
Clock Control Register **[Reset value: $9F_H$]**



Field	Bits	Typ	Description
REL	[4..0]	rw	**Slowdown divider** REL is used to divide down the system clock during slow down mode
KDIV	[7..5]	rw	**K-divider** KDIV selects the PLL division factor according to **Table 5-2**

5.7 Oscillator and Clock Circuit

XTAL2 and XTAL1 are the input and output of a single-stage on-chip inverter which can be configured with off-chip components as a Pierce oscillator. The oscillator, in any case, drives the internal clock generator. The clock generator provides the internal clock signals to the chip. These signals define the internal phases, states and machine cycles.

Figure 5-5 shows the recommended oscillator circuit.

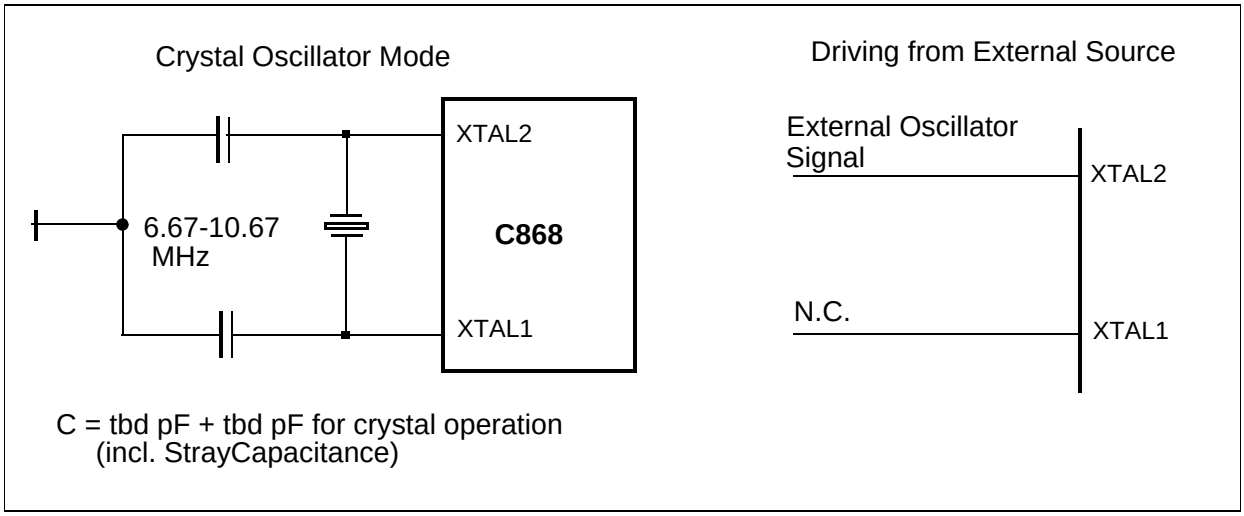


Figure 5-5 Recommended Oscillator Circuit

In this application the on-chip oscillator is used as a crystal-controlled, positive-reactance oscillator (a more detailed schematic is given in **Figure 5-6**). It is operated in its fundamental response mode as an inductive reactor in parallel resonance with a capacitor external to the chip. The crystal specifications and capacitances are non-critical. In this circuit tbd pF can be used as single capacitance at any frequency together with a good quality crystal. A ceramic resonator can be used in place of the crystal in cost-critical applications. If a ceramic resonator is used, the two capacitors normally have different values depending on the oscillator frequency. We recommend consulting the manufacturer of the ceramic resonator for value specifications of these capacitors.

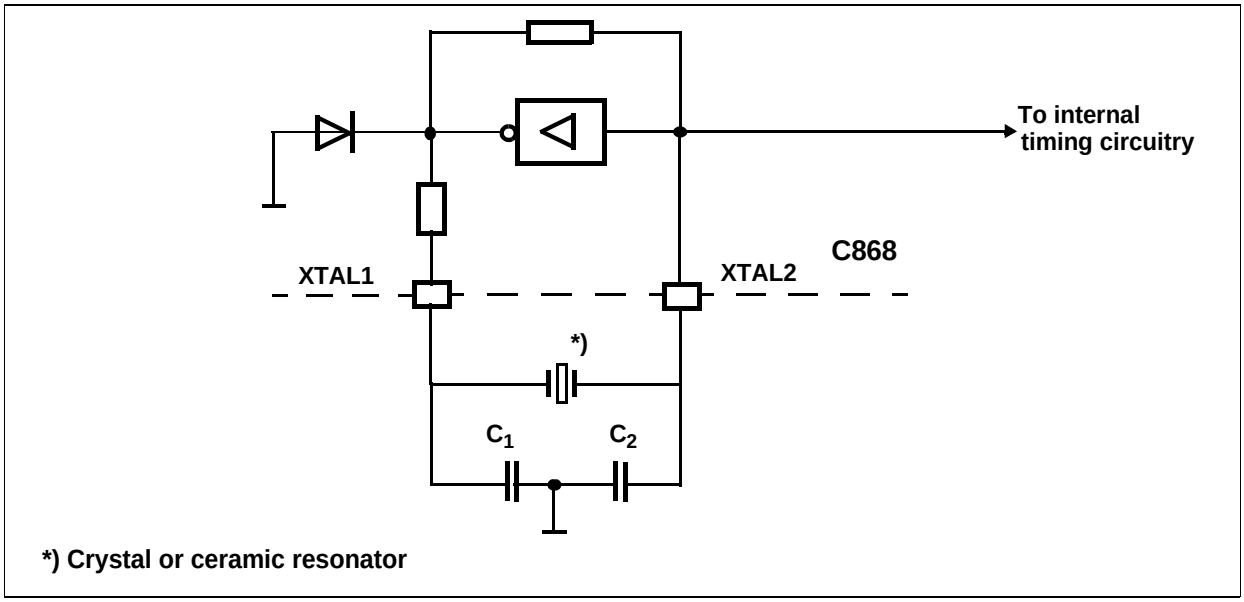


Figure 5-6 On-Chip Oscillator Circuitry

To drive the C868 with an external clock source, the external clock signal has to be applied to XTAL2, as shown in **Figure 5-7**. XTAL1 has to be left unconnected. A pullup resistor is suggested (to increase the noise margin), but is optional if V_{OH} of the driving gate corresponds to the V_{IH2} specification of XTAL2.

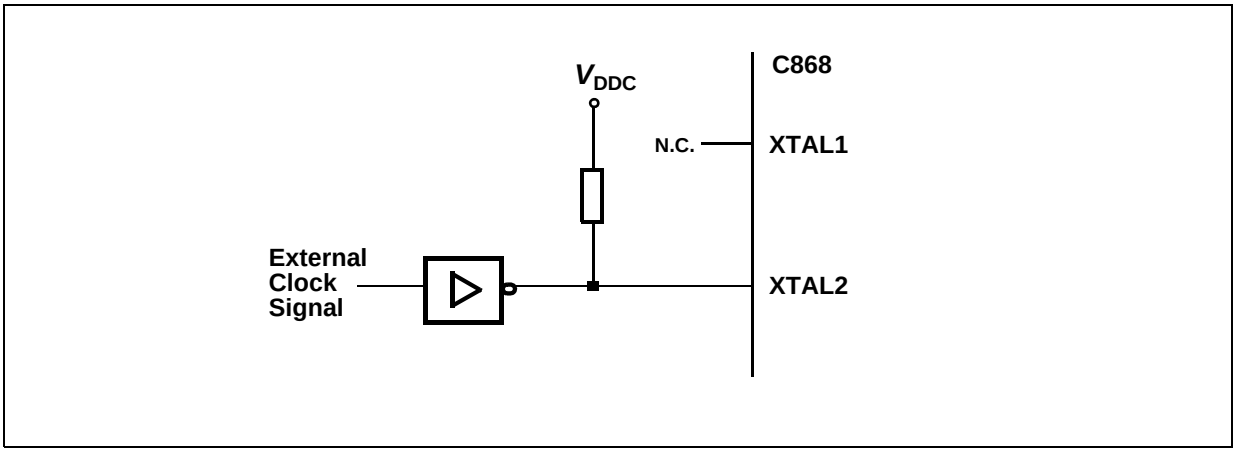


Figure 5-7 External Clock Source

6 Fail Save Mechanism

The C868 offers enhanced fail save mechanisms, which allow an automatic recovery from software upset or hardware failure :

a programmable watchdog timer (WDT), with variable time-out period from 12.8µs to 819.2µs at f_{SYS} = 40 MHz.

6.1 Programmable Watchdog Timer

To protect the system against software failure, the user's program has to clear this watchdog within a previously programmed time period. If the software fails to do this periodical refresh of the watchdog timer, an internal reset will be initiated. The software can be designed so that the watchdog times out if the program does not work properly. It also times out if a software error is based on hardware-related problems.

The watchdog timer in the C868 is a 16-bit timer, which is incremented by a count rate of $f_{SYS}/2$ up to $f_{SYS}/128$. The machine clock of the C868 is divided by a prescaler, a divide-by-two or a divide-by-128 prescaler. The upper 8 bits of the Watchdog Timer can be preset to a user-programmable value via a watchdog service access in order to vary the watchdog expire time. The lower 8 bits are reset on each service access. **Figure 6-1** shows the block diagram of the watchdog timer unit.

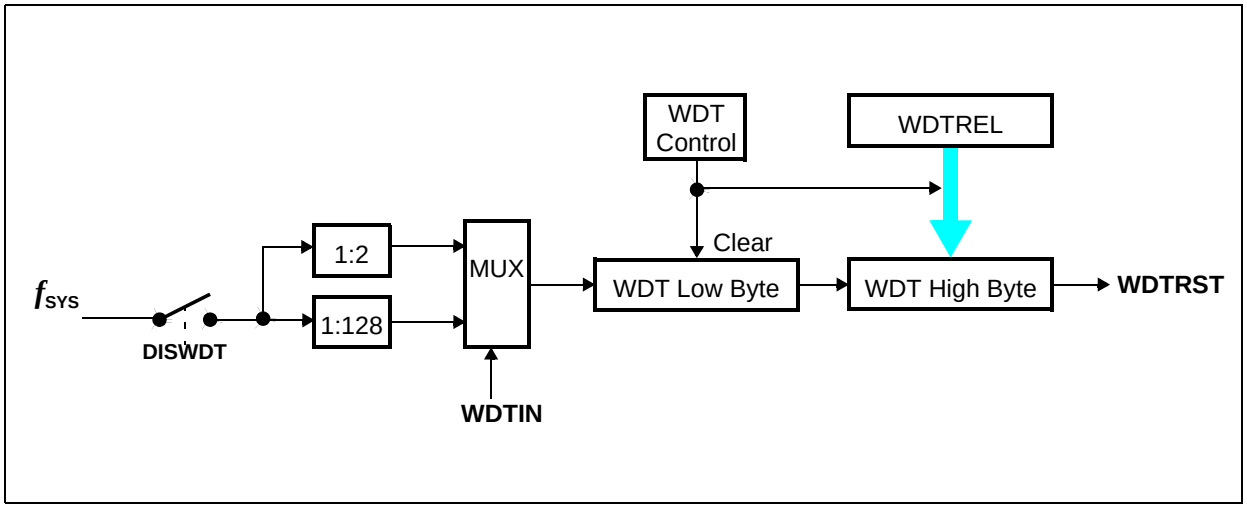


Figure 6-1 Block Diagram of the Programmable Watchdog Timer

6.1.1 Register Definition of the Watchdog Timer

The current count value of the Watchdog Timer is contained in the Watchdog Timer Register WDT, which is a non-bitaddressable read-only register. The operation of the Watchdog Timer is controlled by its bitaddressable Watchdog Timer Control Register

WDTCON. This register specifies the reload value for the high byte of the timer and selects the input clock prescaling factor.

WDTREL
Watchdog Timer Reload Register **[Reset value: 00_H]**

7	6	5	4	3	2	1	0
WDTREL							
rw							

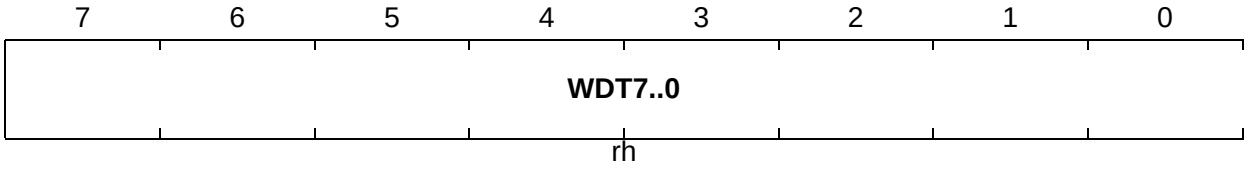
Field	Bits	Typ	Description
WDTREL	[7:0]	rw	**Watchdog Timer Reload Value** (for the high byte of WDT)

WDTCON
Watchdog Timer Register **[Reset value: $XXXXXX00_B$]**

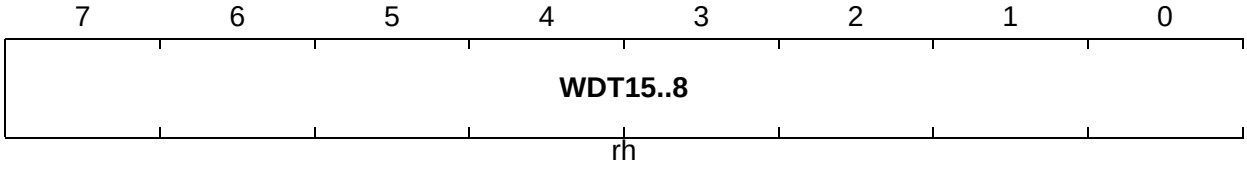
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	WDTIN
r	r	r	r	r	r	r	rw

Field	Bits	Typ	Description
WDTIN	0	rw	**Watchdog Timer Input Frequency Selection** ‘0’: Input frequency is f_{SYS}/ 2 ‘1’: Input frequency is f_{SYS} / 128
-	[7:2]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

WDTL
Watchdog Timer,Low Byte **[Reset value: 00_H]**



WDTH
Watchdog Timer,High Byte **[Reset value: 00_H]**



Field	Bits	Typ	Description
WDT	[7:0] of WDTL, [7:0] of WDTH	rh	**Watchdog Timer Current Value**

SCUWDT
SCU/Watchdog Control Register **[Reset value: 00_H]**

7	6	5	4	3	2	1	0
-	**PLLR**	**-**	**WDTR**	**WDTEOI**	**WDTDIS**	**WDTRS**	**WDTRE**
r	rwh	r	rwh	rw	rw	rw	rw

Field	Bits	Typ	Description
WDTRE	0	rw	**WDT Refresh Enable.** Active high. Set to enable a refresh of the watchdog timer. Must be set before WDTRS.
WDTRS	1	rw	**WDT Refresh Start.** Active high. Set to start refresh operation on the watchdog timer. Must be set after WDTRE.
WDTDIS	2	rw	**WDT Disable.** Active high. Set by software and cleared by general reset. Writing to this bit has no effect if WDTEOI is set.
WDTEOI	3	rw	**WDT End of Initialization.** Active high. Set by software and cleared by general reset.
WDTR	4	rwh	**WDT Reset Indication Bit.** Active high. Set by hardware when a watchdog timer reset occurs. Cleared by reset or WDT_RFSH or software.
PLLR	6	rwh	**PLL Reset Indication Bit.** Active high. Set by hardware when PLL reset occurs. Cleared by reset or software.
-	7,5	r	**reserved;** returns ’0’ if read; should be written with ’0’;

6.1.2 Starting the Watchdog Timer

When the reset input to the Watchdog Timer, the Watchdog Timer is automatically enabled. Once disabled by setting SCUWDT.WDTDIS, it can only be enabled again by a reset. Setting SCUWDT.WDTEOI will render SCUWDT.WDTDIS ineffective. If the software fails to clear the watchdog timer an internal reset will be initiated. The reset cause (external reset or reset caused by the watchdog) can be examined by software (status flag SCUWDT.WDTR). A refresh of the watchdog timer is done by setting bits SCUWDT.WDTRE and SCUWDT.WDTRS consecutively. This double instruction sequence has been implemented to increase system security.

It must be noted, however, that the watchdog timer is halted during the idle mode and power-down mode of the processor (see section "Power Saving Modes"). It is not possible to use the idle mode in combination with the watchdog timer function. Therefore, even the watchdog timer cannot reset the device when one of the power saving modes has been entered accidentally.

6.1.3 Refreshing the Watchdog Timer

At the same time the watchdog timer is started, the 8-bit register WDTH is preset by the contents of WDTREL. Once enabled and the SCUWDT.WDTEOI is set the watchdog cannot be stopped by software but can only be refreshed to the reload value by first setting bit SCUWDT.WDTRE and SCUWDT.WDTRS consecutively. Bit SCUWDT.WDTR will automatically be cleared during the second machine cycle after having been set. For this reason, setting SCUWDT.WDTRS bit has to be a one cycle instruction (e.g. SETB WDTRS). This double-instruction refresh of the watchdog timer is implemented to minimize the chance of an unintentional reset of the watchdog.

The reload register WDTREL can be written to at any time, as already mentioned. Therefore, a periodical refresh of WDTREL can be added to the above mentioned starting procedure of the watchdog timer. Thus a wrong reload value caused by a possible distortion during the write operation to the WDTREL can be corrected by software.

6.1.4 Input Clock Selection

The time period for an overflow of the Watchdog Timer is programmable in two ways :

- **the input frequency** to the Watchdog Timer can be selected via bit WDTIN in register WDTCON to be either $f_{SYS}/2$ or $f_{SYS}/128$.
- **the reload value** WDTREL for the high byte of WDT can be programmed in register WDTCON.

The period P_{WDT} between servicing the Watchdog Timer and the next overflow can therefore be determined by the following formula:

[6.1]

$$P_{WDT} = \frac{2^{(1 + WDTIN*6)} * (2^{16} - WDTREL * 2^{8})}{f_{SYS}}$$

Table 6-1 lists the possible ranges for the watchdog time which can be achieved using a certain module clock. Some numbers are rounded to 3 significant digits.

Table 6-1 Watchdog Time Ranges

Reload value in WDTREL	Prescaler for f_{SYS}					
	2 (WDTIN = '0')			128 (WDTIN = '1')		
	40 MHz	20 MHz	16 MHz	40 MHz	20 MHz	16 MHz
FF_H	12.8 µs	25.6 µs	32.0 µs	819.2 µs	1.64 ms	2.05 ms
$7F_H$	1.65 ms	3.3 ms	4.13 ms	105.7 ms	211.3 ms	264 ms
00_H	3.28 ms	6.55 ms	8.19 ms	209.7 ms	419.4 ms	524 ms

Note : For safety reasons, the user is advised to rewrite WDTCON each time before the Watchdog Timer is serviced.

7 Interrupt System

The C868 provides 13 interrupt vectors with four priority levels. Nine interrupt requests are generated by the on-chip peripherals (timer 0, timer 1, timer 2, serial channel, A/D converter, and the capture/compare unit with 4 interrupts) and four interrupts may be triggered externally.

The wake-up from power-down mode interrupt has a special functionality which allows the software power-down mode to be terminated by a short negative pulse at pins CCPOS0/T2/INT0/AN0 or P1.4/RxD.

The 13 interrupt sources are divided into six groups. Each group can be programmed to one of the four interrupt priority levels. Additionally, 4 of these interrupt sources are channeled from 7 Capture/Compare (CCU6) interrupt sources.

7.1 Structure of the Interrupt System

Figure 7-1 to **Figure 7-6** give a general overview of the interrupt sources and illustrate the request and control flags which are described in the next sections.

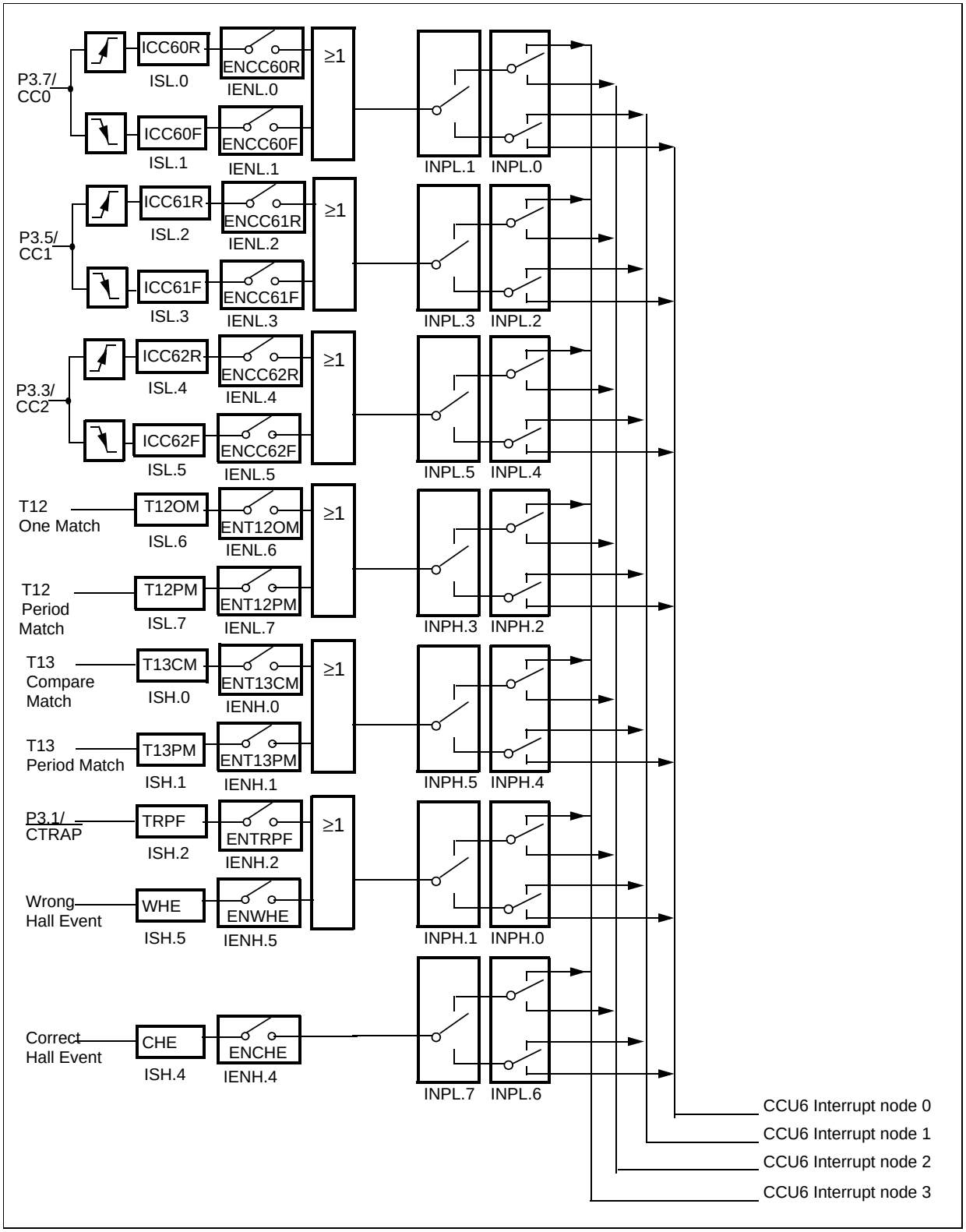


Figure 7-1 Capture/Compare module interrupt structure

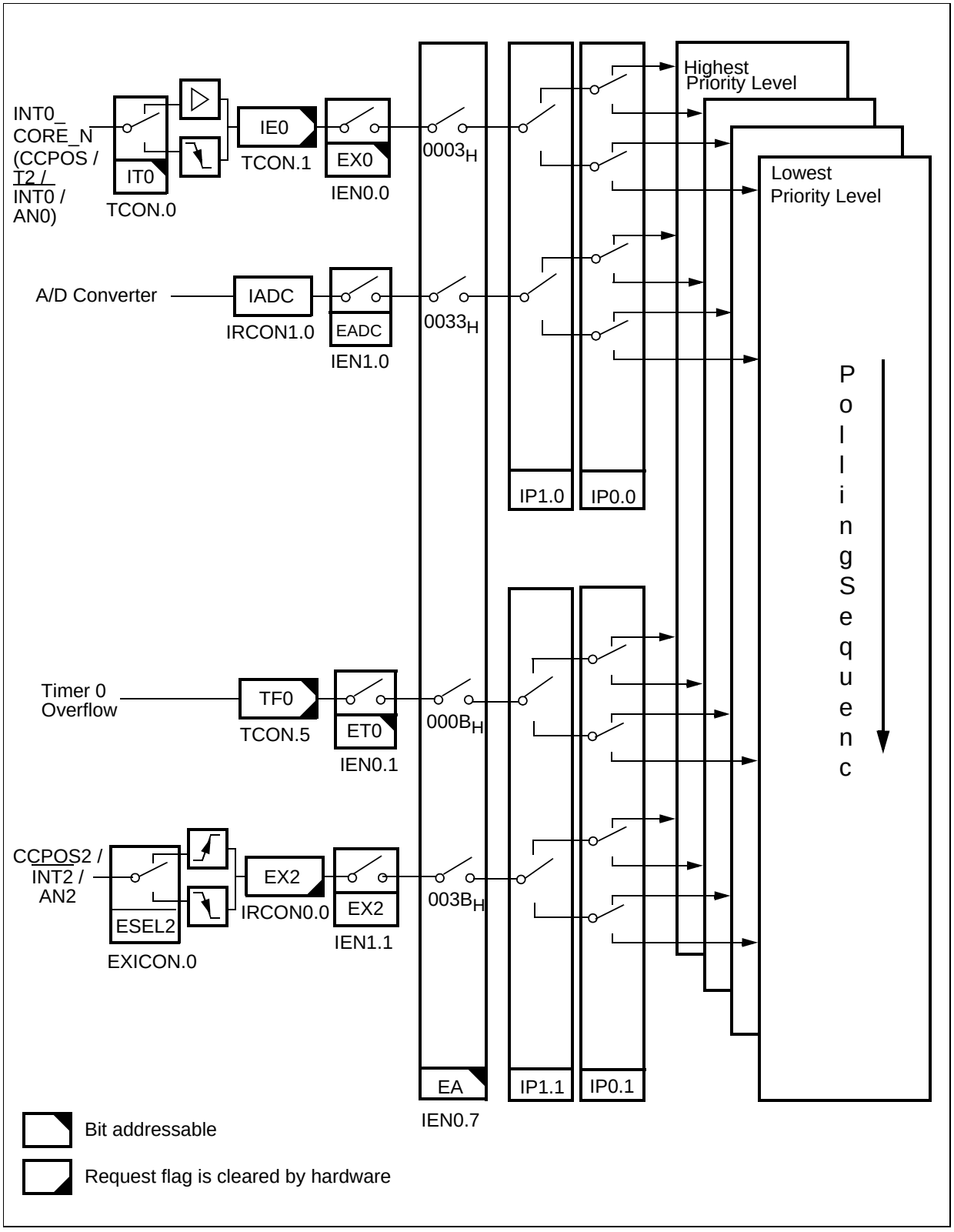


Figure 7-2 Interrupt Structure, Overview Part 1

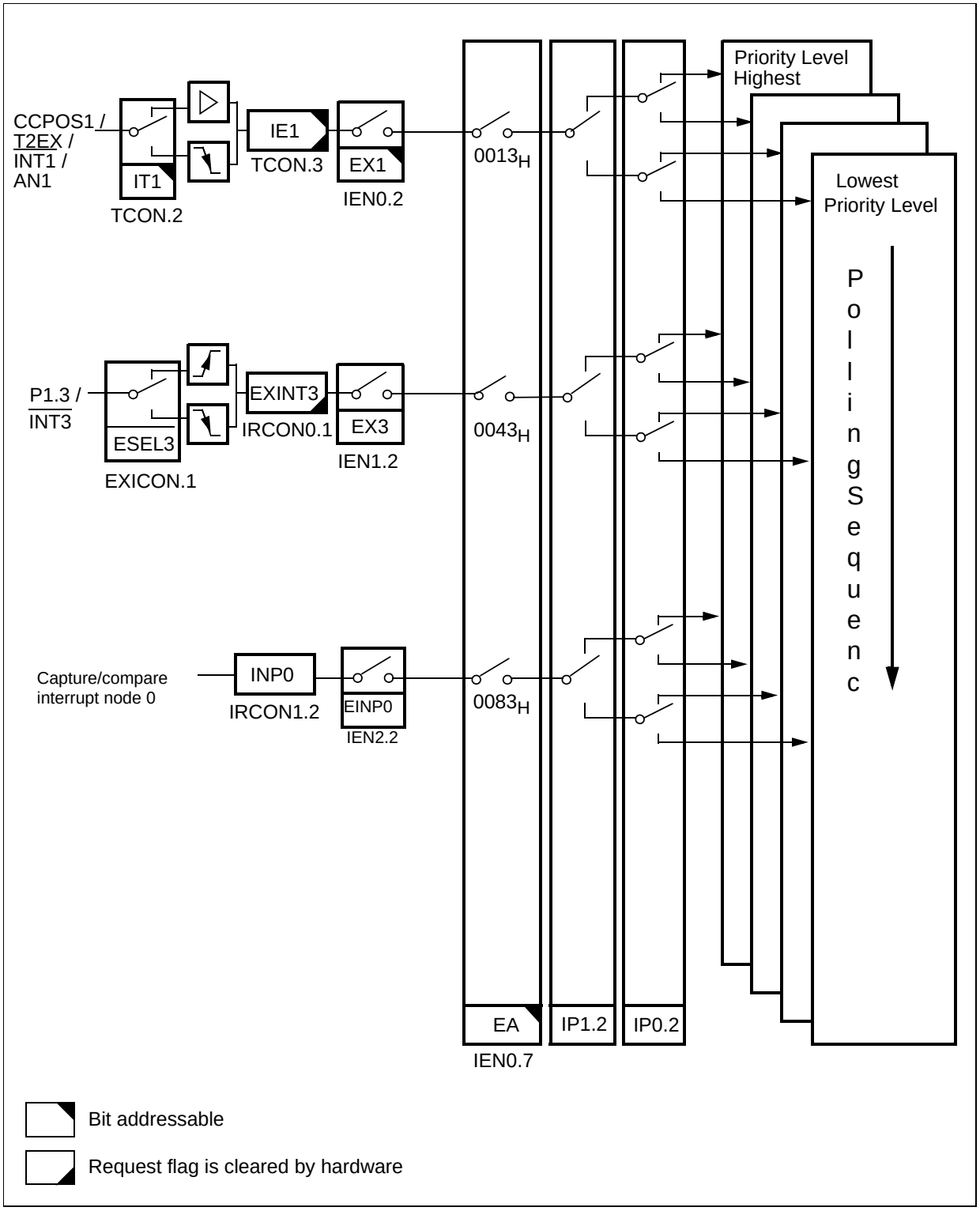


Figure 7-3 Interrupt Structure, Overview Part 2

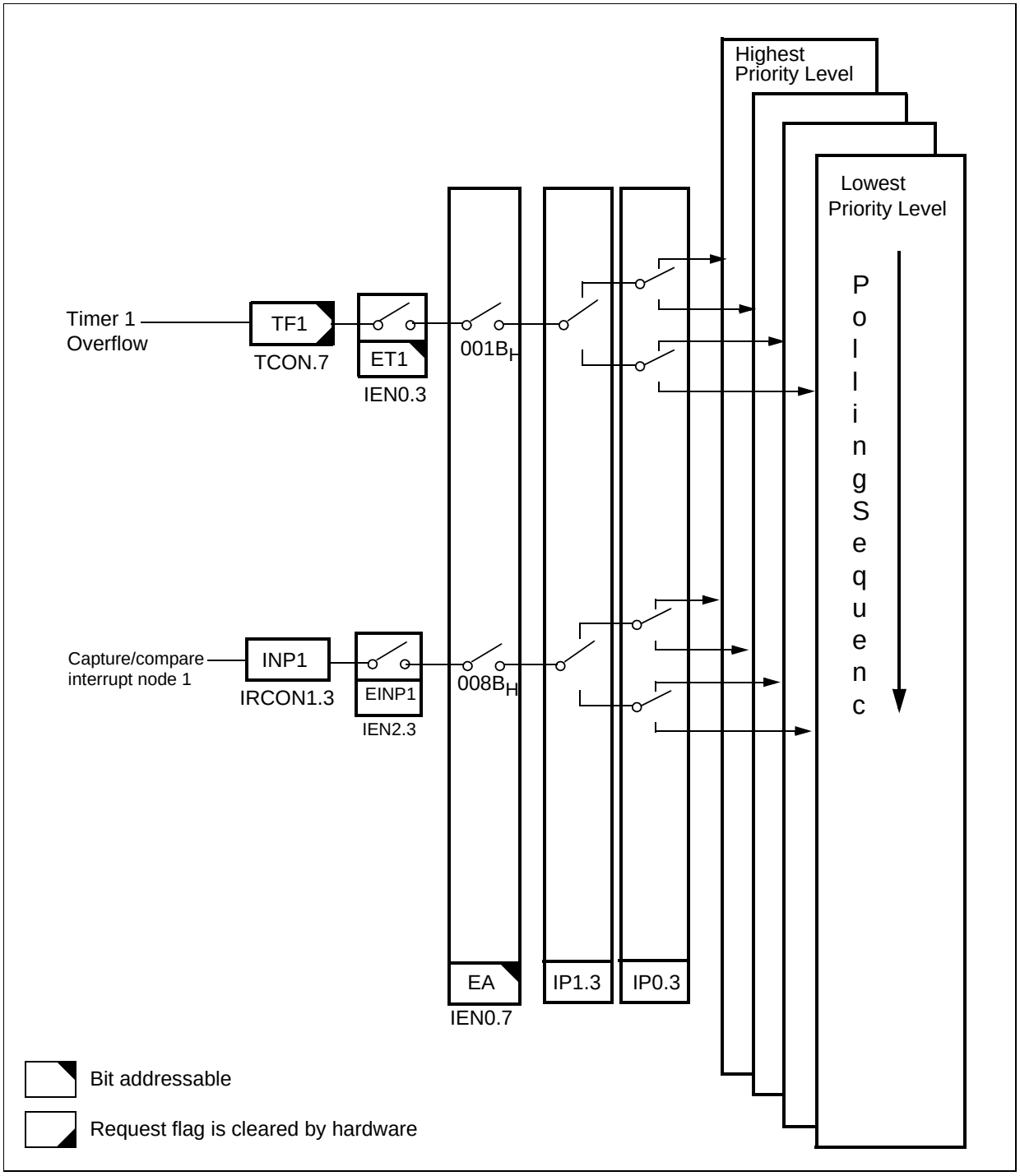


Figure 7-4 Interrupt Structure, Overview Part 3

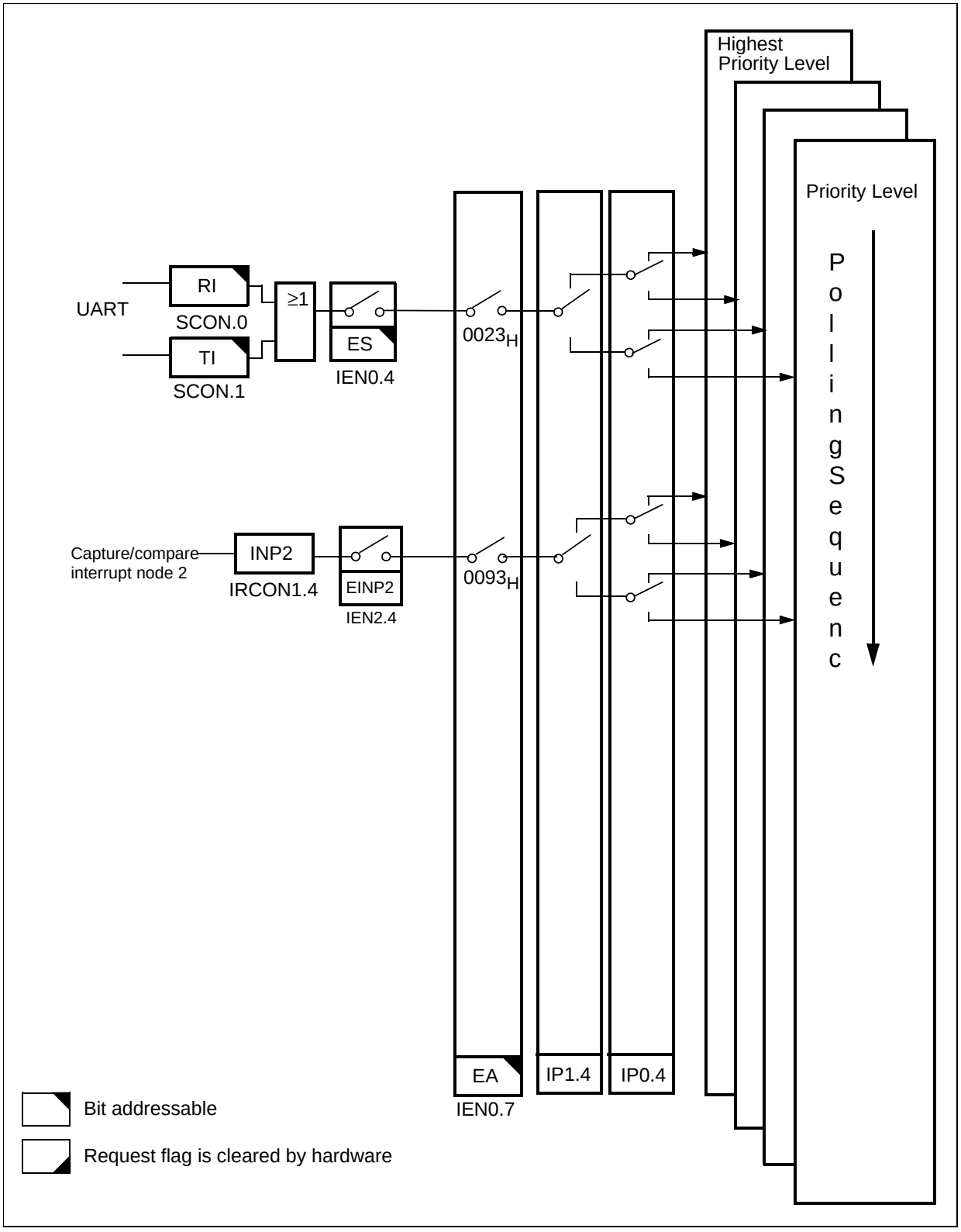


Figure 7-5 Interrupt Structure, Overview Part 4

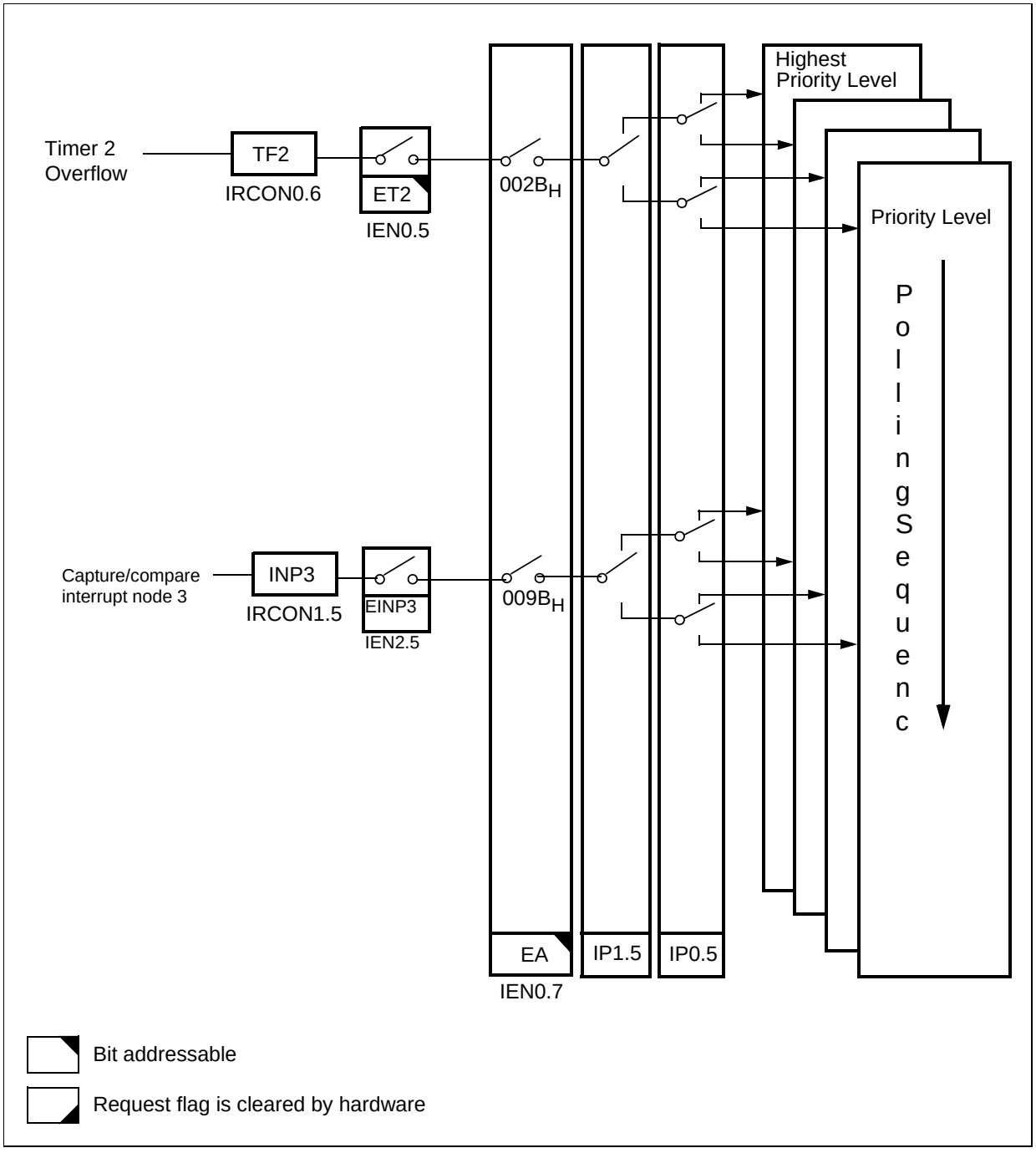


Figure 7-6 Interrupt Structure, Overview Part 5

7.2 Interrupt Registers

7.2.1 Interrupt Enable Registers

Each interrupt vector can be individually enabled or disabled by setting or clearing the corresponding bit in the interrupt enable registers IEN0, IEN1, IEN2. Register IEN0 also contains the global disable bit (EA), which can be cleared to disable all interrupts at once. Generally, after reset all interrupt enable bits are set to 0. That means that the corresponding interrupts are disabled.

The SFR IEN0 contains the enable bits for the external interrupts 0 and 1, the timer interrupts, and the UART interrupt.

IEN0
Interrupt Enable Register 0 **[Reset value: 0X000000$_B$]**

AF_H	AE_H	AD_H	AC_H	AB_H	AA_H	$A9_H$	$A8_H$
EA	**-**	**ET2**	**ES**	**ET1**	**EX1**	**ET0**	**EX0**
rw	r	rw	rw	rw	rw	rw	rw

Field	Bits	Typ	Description
EX0	0	rw	**External interrupt 0 enable.** If EX0 = 0, the external interrupt 0 is disabled. If EX0 = 1, the external interrupt 0 is enabled.
ET0	1	rw	**Timer 0 overflow interrupt enable.** If ET0 = 0, the timer 0 interrupt is disabled. If ET0 = 1, the timer 0 interrupt is enabled.
EX1	2	rw	**External interrupt 1 enable.** If EX1 = 0, the external interrupt 1 is disabled. If EX1 = 1, the external interrupt 1 is enabled.
ET1	3	rw	**Timer 1 overflow interrupt enable.** If ET1 = 0, the timer 1 interrupt is disabled. If ET1 = 1, the timer 1 interrupt is enabled.
ES	4	rw	**Serial channel (UART) interrupt enable** If ES = 0, the serial channel interrupt 0 is disabled. If ES = 1, the serial channel interrupt 0 is enabled.
ET2	5	rw	**Timer 2 overflow / external reload interrupt enable.** If ET2 = 0, the timer 2 interrupt is disabled. If ET2 = 1, the timer 2 interrupt is enabled.

Field	Bits	Typ	Description
EA	7	rw	**Enable/disable all interrupts.** If EA=0, no interrupt will be acknowledged. If EA=1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.
-	6	r	**reserved;** returns ’0’ if read; should be written with ’0’;

The SFR IEN1 contains the enable bits for the external interrupts 2 to 3, and the A/D converter interrupt.

IEN1
Interrupt Enable Register 1 **[Reset value: XXXXX000$_B$]**

7	6	5	4	3	2	1	0
-	-	-	-	-	EX3	EX2	EADC
r	r	r	r	r	rw	rw	rw

Field	Bits	Typ	Description
EADC	0	rw	**A/D converter interrupt enable** If EADC = 0, the A/D converter interrupt is disabled. If EADC = 1, the A/D converter interrupt is enabled.
EX2	1	rw	**External interrupt 2 enable** If EX2 = 0, external interrupt 2 is disabled. If EX2 = 1, external interrupt 2 is enabled.
EX3	2	rw	**External interrupt 3 / Timer 2 capture/compare** interrupt 0 enable If EX3 = 0, external interrupt 3 is disabled. If EX3 = 1, external interrupt 3 is enabled.
-	[7:3]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

IEN2
Interrupt Enable Register 2 **[Reset value: XX0000XX$_B$]**

7	6	5	4	3	2	1	0
-	-	EINP3	EINP2	EINP1	EINP0	-	-
r	r	rw	rw	rw	rw	r	r

Field	Bits	Typ	Description
EINP0	2	rw	**Capture/compare interrupt node 0 enable** If EINP0 = 0, the interrupt node 0 is disabled. If EINP0 = 1, the interrupt node 0 is enabled.
EINP1	3	rw	**Capture/compare interrupt node 1 enable** If EINP0 = 0, the interrupt node 1 is disabled. If EINP0 = 1, the interrupt node 1 is enabled.
EINP2	4	rw	**Capture/compare interrupt node 2 enable** If EINP0 = 0, the interrupt node 2 is disabled. If EINP0 = 1, the interrupt node 2 is enabled.
EINP3	5	rw	**Capture/compare interrupt node 3 enable** If EINP0 = 0, the interrupt node 3 is disabled. If EINP0 = 1, the interrupt node 3 is enabled.
-	[7:6], [1:0]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

7.2.2 Interrupt Request Flags

The request flags for the different interrupt sources are located in several special function registers. This section describes the locations and meanings of these interrupt request flags in detail. .

TCON
Timer 0/1 Control Register **[Reset value: 00_H]**

$8F_H$	$8E_H$	$8D_H$	$8C_H$	$8B_H$	$8A_H$	89_H	88_H
TF1	**TR1**	**TF0**	**TR0**	**IE1**	**IT1**	**IE0**	**IT0**
rwh	rw	rwh	rw	rwh	rw	rwh	rw

The functions of the shaded bits are not described here

Field	Bits	Typ	Description
IT0	0	rw	**External interrupt 0 level/edge trigger control flag** If IT0 = 0, low level triggered external interrupt 0 is selected. If IT0 = 1, falling edge triggered external interrupt 0 is selected.
IE0	1	rwh	**External interrupt 0 request flag** Set by hardware when external interrupt 0 edge is detected. Cleared by hardware when processor vectors to interrupt routine.
IT1	2	rw	**External interrupt 1 level/edge trigger control flag** If IT1 = 0, low level triggered external interrupt 1 is selected. If IT1 = 1, falling edge triggered external interrupt 1 is selected.
IE1	3	rwh	**External interrupt 1 request flag** Set by hardware when external interrupt 1 edge is detected. Cleared by hardware when processor vectors to interrupt routine.

Field	Bits	Typ	Description
TF0	5	rwh	**Timer 0 overflow flag** Set by hardware on timer/counter 0 overflow. Cleared by hardware when processor vectors to interrupt routine.
TF1	7	rwh	**Timer 1 overflow flag** Set by hardware on timer/counter 1 overflow. Cleared by hardware when processor vectors to interrupt routine.

The **timer 0 and timer 1 interrupts** are generated by TF0 and TF1 in register TCON, which are set by a rollover in their respective timer/counter registers. When a timer interrupt is generated, the flag that generated it is cleared by the on-chip hardware when the service routine is vectored to.

The **external interrupts 0 and 1** (P1.5/CCPOS0/T2/$\overline{\text{INT0}}$/AN0 ,P1.6/CCPOS1/T2EX/INT1/AN1) can each be either level-activated or negative transition-activated, depending on bits IT0 and IT1 in register TCON. The flags that actually generate these interrupts are bits IE0 and IE1 in TCON. When an external interrupt is generated, the flag that generated this interrupt is cleared by the hardware when the service routine is vectored to, but only if the interrupt was transition-activated. If the interrupt was level-activated, then the requesting external source directly controls the request flag, rather than the on-chip hardware.

IRCON0
External Interrupt Control Register 0 **[Reset value: XXXXXX00$_B$]**

7	6	5	4	3	2	1	0
-	-	-	-	-	-	**EXINT3**	**EXINT2**
r	r	r	r	r	r	rwh	rwh

Field	Bits	Typ	Description
EXINT2	0	rwh	**Interrupt Request Flag for External Interrupt 2** 0 : Interrupt request is not active, cleared by software. (default) 1 : Interrupt request is active, set by hardware.

Field	Bits	Typ	Description
EXINT3	1	rwh	**Interrupt Request Flag for External Interrupt 3** 0 : Interrupt request is not active, cleared by software. (default) 1 : Interrupt request is active, set by hardware.
-	[7:2]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

The external interrupt 2 and 3(P1.7/CCPOS2/$\overline{\text{INT2}}$/AN2, P1.3/$\overline{\text{INT3}}$) can be either positive or negative transition-activated, depending on the bits register EXICON. The flags that actually generates this interrupt are bits EXINT2 and EXINT3 in register IRCON0. When processing the external interrupts, flags must be cleared by software.

EXICON
External Interrupt Control Register **[Reset value: XXXXXX00$_B$]**

7	6	5	4	3	2	1	0
-	-	-	-	-	-	**ESEL3**	**ESEL2**
r	r	r	r	r	r	rw	rw

Field	Bits	Typ	Description
ESEL2	0	rw	**External Interrupt 2 Edge Trigger Select.** 0 : Interrupt on falling edge. (default) 1 : Interrupt on rising edge.
ESEL3	1	rw	**External Interrupt 3 Edge Trigger Select.** 0 : Interrupt on falling edge. (default) 1 : Interrupt on rising edge.
-	[7:2]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

All external interrupts (P1.7/CCPOS2/$\overline{\text{INT2}}$/AN2, P1.3/$\overline{\text{INT3}}$) can be either positive or negative transition-activated, depending on the bits register EXICON. The flags that actually generates this interrupt are bits EXINT2 and EXINT3 in register IRCON0. The flags must be cleared by software.

.

T2CON
Timer 2 Control Register **[Reset value: 00_H]**

CF_H	CE_H	CD_H	CC_H	CB_H	CA_H	$C9_H$	$C8_H$
TF2	**EXF2**	**RCLK**	**TCLK**	**EXEN2**	**TR2**	**C/$\overline{T2}$**	**CP/$\overline{RL2}$**
rwh	rwh	rw	rw	rw	rw	rw	rw

The functions of the shaded bits are not described here

Field	Bits	Typ	Description
EXF2	6	rwh	**Timer/counter 2 External Flag** This bit is set by hardware when a capture/reload occurred upon a negative transition at pin T2EX, if bit EXEN2=1. An interrupt request to the core is generated, unless bit DCEN=1. This bit must be cleared by software.
TF2	7	rw	Timer 2 Overflow Flag Set by a timer 2 overflow. Must be cleared by software. TF2 will not be set when either RCLK=1 or TCLK=1.

The **timer 2 interrupt** is generated by bit TF2 or EXF2 in register T2CON. This flags are not cleared by hardware when the service routine is vectored to. They should be cleared by software.

IRCON1
External Interrupt Request Register 1 **[Reset value: XX0000X0$_B$]**

7	6	5	4	3	2	1	0
-	**-**	**INP3**	**INP2**	**INP1**	**INP0**	**-**	**IADC**
r	r	rwh	rwh	rwh	rwh	r	rwh

Field	Bits	Typ	Description
IADC	0	rwh	**Interrupt Request Flag for ADC** 0 : Interrupt request is not active, cleared by software. (default) 1 : Interrupt request is active, set by hardware.
INP0	2	rwh	**Interrupt Request Flag for CCU6 interrupt node 0** 0 : Interrupt request is not active, cleared by software. (default) 1 : Interrupt request is active, set by hardware.
INP1	3	rwh	**Interrupt Request Flag for CCU6 interrupt node 1** 0 : Interrupt request is not active, cleared by software. (default) 1 : Interrupt request is active, set by hardware.
INP2	4	rwh	**Interrupt Request Flag for CCU6 interrupt node 2** 0 : Interrupt request is not active, cleared by software. (default) 1 : Interrupt request is active, set by hardware.
INP3	5	rwh	**Interrupt Request Flag for CCU6 interrupt node 3** 0 : Interrupt request is not active, cleared by software. (default) 1 : Interrupt request is active, set by hardware.
-	[7:1]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

The **A/D converter interrupt** is generated by IADC bit in register IRCON1. If an interrupt is generated, in any case the converted result in ADDATH is valid on the first instruction of the interrupt service routine. If continuous conversion is established, IADC is set once during each conversion. If an A/D converter interrupt is generated, flag IADC will have to be cleared by software.

SCON
Serial Channel Control Register [Reset value: 00_H]

$9F_H$	$9E_H$	$9D_H$	$9C_H$	$9B_H$	$9A_H$	99_H	98_H
SM0	**SM1**	**SM2**	**REN**	**TB8**	**RB8**	**TI**	**RI**
rwh	rwh	rw	rw	rw	rw	rwh	rwh

The functions of the shaded bits are not described here

Field	Bits	Typ	Description
RI	0	rwh	**Serial interface receiver interrupt flag** Set by hardware if a serial data byte has been received. Must be cleared by software.
TI	1	rwh	**Serial interface transmitter interrupt flag** Set by hardware at the end of a serial data transmission. Must be cleared by software.

The **serial interface interrupt** is generated by a logical OR of flag RI and TI in SFR SCON. Neither of these flags is cleared by hardware when the service routine is vectored to. In fact, the service routine will normally have to determine whether it was the receive interrupt flag or the transmission interrupt flag that generated the interrupt, and the corresonding bit will have to be cleared by software.

7.2.3 Interrupt Control Registers for CCU6

Register IS contains the individual interrupt request bits. This register can only be read, write actions have no impact on the contents of this register. The SW can set or reset the bits individually by writing to the registers ISS (to set the bits) or to register ISR (to reset the bits). .

ISL
Capture/Compare Interrupt Register ,Low Byte **[Reset value: 00_H]**

7	6	5	4	3	2	1	0
T12PM	**T12OM**	**ICC62F**	**ICC62R**	**ICC61F**	**ICC61R**	**ICC60F**	**ICC60R**
rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ICC60R, ICC61R, ICC62R	0, 2, 4	rh	**Capture, Compare-Match Rising Edge Flag** In compare mode, a compare-match has been detected while T12 was counting up. In capture mode, a rising edge has been detected at the input CC6x (x=0, 1, 2). 0 The event has not yet occurred since this bit has been reset for the last time. 1 The event described above has been detected.

Field	Bits	Type	Description
ICC60F, ICC61F, ICC62F	1, 3, 5	rh	**Capture, Compare-Match Falling Edge Flag** In compare mode, a compare-match has been detected while T12 was counting down. In capture mode, a falling edge has been detected at the input CC6x (x=0, 1, 2). 0 The event has not yet occurred since this bit has been reset for the last time. 1 The event described above has been detected.
T12OM	6	rh	**Timer T12 One-Match Flag** 0 A timer T12 one-match (while counting down) has not yet been detected since this bit has been reset for the last time. 1 A timer T12 one-match (while counting down) has been detected.
T12PM	7	rh	**Timer T12 Period-Match Flag** 0 A timer T12 period-match (while counting up) has not yet been detected since this bit has been reset for the last time. 1 A timer T12 period-match (while counting up) has been detected.

ISH
Capture/Compare Interrupt Register ,High Byte **[Reset value: 00_H]**

7	6	5	4	3	2	1	0
-	CCU_IDLE	WHE	CHE	TRPS	TRPF	T13PM	T13CM
r	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
T13CM	0	rh	**Timer T13 Compare-Match Flag** 0 A timer T13 compare-match has not yet been detected since this bit has been reset for the last time. 1 A timer T13 compare-match has been detected.
T13PM	1	rh	**Timer T13 Period-Match Flag** 0 A timer T13 period-match has not yet been detected since this bit has been reset for the last time. 1 A timer T13 period-match has been detected.
TRPF	2	rh	**Trap Flag** The trap flag TRPF will be set by HW if TRPPEN='1' and CTRAP='0' or by SW. If TRPM2='0', bit TRPF is reset by HW if the input CTRAP becomes inactive (TRPPEN='1'). If TRPM2='1', bit TRPF has to be reset by SW in order to leave the trap state. 0 The trap condition has not been detected. 1 The trap condition has been detected (input CTRAP has been '0' or by SW).
TRPS[1)]	3	rh	**Trap State** 0 The trap state is not active. 1 The trap state is active. Bit TRPS is set while bit TRPF='1'. It is reset according to the mode selected in register TRPCTR.

Field	Bits	Type	Description
CHE[2]	4	rh	**Correct Hall Event** 0 A transition to a correct (=expected) hall event has not yet been detected since this bit has been reset for the last time. 1 A transition to a correct (=expected) hall event has not yet been detected.
WHE[3]	5	rh	**Wrong Hall Event** 0 A transition to a wrong hall event (not the expected one) has not yet been detected since this bit has been reset for the last time. 1 A transition to a wrong hall event (not the expected one) has been detected.
CCU_IDLE[4]	6	rh	**IDLE State** This bit is set together with bit WHE (wrong hall event) and it has to be reset by SW. 0 No action. 1 Bitfield MCMP is cleared, the selected outputs are set to passive state.
-	7	r	**reserved;** returns ’0’ if read; should be written with ’0’;

1) During the trap state, the selected outputs are set to the passive state. The logic level driven during the passive state is defined by the corresponding bit in register CCMCON. Bit TRPS=’1’ and TRPF=’0’ can occur if the trap condition is no longer active but the selected synchronization has not yet taken place.

2) On every valid hall edge the contents of CURH is compared with the pattern on pin CCPOSx and if equal bit CHE is set.

3) On every valid hall edge the contents of EXPH is compared with the pattern on pin CCPOSx. If both compares (CURH and EXPH with CCPOSx) are not true, bit WHE (wrong hall event) is set.

4) Bit field MCMP is hold to ’0’ by hardware as long as CCU_IDLE = ’1’.

Note: Not all bits in register IS can generate an interrupt. Other status bits have been added, which have a similar structure for their set and reset actions.

Note: The interrupt generation is independent from the value of the bits in register IS, e.g. the interrupt will be generated (if enabled) even if the corresponding bit is already set. The trigger for an interrupt generation is the detection of a set condition (by HW or SW) for the corresponding bit in register IS.

Note: In compare mode (and hall mode), the timer-related interrupts are only generated while the timer is running (TxR=1). In capture mode, the capture interrupts are also generated while the timer T12 is stopped.

Register ISS contains the individual interrupt request set bits to generate a CCU6 interrupt request by software.

ISSL
Capture/Compare Interrupt Status Set Register ,Low Byte [Reset value: 00_H]

7	6	5	4	3	2	1	0
ST12PM	**ST12OM**	**SCC62F**	**SCC62R**	**SCC61F**	**SCC61R**	**SCC60F**	**SCC60R**
w	w	w	w	w	w	w	w

Field	Bits	Type	Description
SCC60R	0	w	**Set Capture, Compare-Match Rising Edge Flag** 0 No action 1 Bit CC60R in register IS will be set.
SCC60F	1	w	**Set Capture, Compare-Match Falling Edge Flag** 0 No action 1 Bit CC60F in register IS will be set.
SCC61R	2	w	**Set Capture, Compare-Match Rising Edge Flag** 0 No action 1 Bit CC61R in register IS will be set.
SCC61F	3	w	**Set Capture, Compare-Match Falling Edge Flag** 0 No action 1 Bit CC61F in register IS will be set.
SCC62R	4	w	**Set Capture, Compare-Match Rising Edge Flag** 0 No action 1 Bit CC62R in register IS will be set.
SCC62F	5	w	**Set Capture, Compare-Match Falling Edge Flag** 0 No action 1 Bit CC62F in register IS will be set.
ST12OM	6	w	**Set Timer T12 One-Match Flag** 0 No action 1 Bit T12OM in register IS will be set.
ST12PM	7	w	**Set Timer T12 Period-Match Flag** 0 No action 1 Bit T12PM in register IS will be set.

ISSH
Capture/Compare Interrupt Status Set Register ,High Byte [Reset value: 00_H]

7	6	5	4	3	2	1	0
-	SIDLE	SWHE	SCHE	-	STRPF	ST13PM	ST13CM
r	w	w	w	r	w	w	w

Field	Bits	Type	Description
ST13CM	0	w	**Set Timer T13 Compare-Match Flag** 0 No action 1 Bit T13CM in register IS will be set.
ST13PM	1	w	**Set Timer T13 Period-Match Flag** 0 No action 1 Bit T13PM in register IS will be set.
STRPF	2	w	**Set Trap Flag** 0 No action 1 Bit TRPF in register IS will be set (not taken into account while input CTRAP=’0’ and TRPPEN=’1’.
SCHE	3	w	**Set Correct Hall Event Flag** 0 No action 1 Bit CHE in register IS will be set.
SWHE	5	w	**Set Wrong Hall Event Flag** 0 No action 1 Bit WHE in register IS will be set.
SIDLE	6	w	**Set IDLE Flag** 0 No action 1 Bit CCU_IDLE in register IS will be set.
0	3, 7	r	**reserved;** returns ’0’ if read; should be written with ’0’;

Note: If the setting by HW of the corresponding flags can lead to an interrupt, the setting by SW has the same effect.

Register ISR contains the individual interrupt request reset the corresponding flags by software.

ISRL
Capture/Compare Interrupt Status Reset Register ,Low Byte [Reset value: 00_H]

7	6	5	4	3	2	1	0
RT12PM	**RT12OM**	**RCC62F**	**RCC62R**	**RCC61F**	**RCC61R**	**RCC60F**	**RCC60R**
w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RCC60R	0	w	**Reset Capture, Compare-Match Rising Edge Flag** 0 No action 1 Bit CC60R in register IS will be reset.
RCC60F	1	w	**Reset Capture, Compare-Match Falling Edge Flag** 0 No action 1 Bit CC60F in register IS will be reset.
RCC61R	2	w	**Reset Capture, Compare-Match Rising Edge Flag** 0 No action 1 Bit CC61R in register IS will be reset.
RCC61F	3	w	**Reset Capture, Compare-Match Falling Edge Flag** 0 No action 1 Bit CC61F in register IS will be reset.
RCC62R	4	w	**Reset Capture, Compare-Match Rising Edge Flag** 0 No action 1 Bit CC62R in register IS will be reset.
RCC62F	5	w	**Reset Capture, Compare-Match Falling Edge Flag** 0 No action 1 Bit CC62F in register IS will be reset.
RT12OM	6	w	**Reset Timer T12 One-Match Flag** 0 No action 1 Bit T12OM in register IS will be reset.
RT12PM	7	w	**Reset Timer T12 Period-Match Flag** 0 No action 1 Bit T12PM in register IS will be reset.

ISRH

Capture/Compare Interrupt Status Reset Register ,High Byte [Reset value: 00_H]

7	6	5	4	3	2	1	0
-	RIDLE	RWHE	RCHE	-	RTRPF	RT13PM	RT13CM
r	w	w	w	r	w	w	w

Field	Bits	Type	Description
RT13CM	0	w	**Reset Timer T13 Compare-Match Flag** 0 No action 1 Bit T13CM in register IS will be reset.
RT13PM	1	w	**Reset Timer T13 Period-Match Flag** 0 No action 1 Bit T13PM in register IS will be reset.
RTRPF	2	w	**Reset Trap Flag** 0 No action 1 Bit TRPF in register IS will be reset (not taken into account while input CTRAP=’0’ and TRPPEN=’1’.
RCHE	4	w	**Reset Correct Hall Event Flag** 0 No action 1 Bit CHE in register IS will be reset.
RWHE	5	w	**Reset Wrong Hall Event Flag** 0 No action 1 Bit WHE in register IS will be reset.
RIDLE	6	w	**Reset IDLE Flag** 0 No action 1 Bit CCU_IDLE in register IS will be reset.
-	3, 7	r	**reserved;** returns ’0’ if read; should be written with ’0’;

Register IEN contains the interrupt enable bits and a control bit to enable the automatic idle function in the case of a wrong hall pattern.

IENL
Capture/Compare Interrupt Enable Register ,Low Byte **[Reset value: 00_H]**

7	6	5	4	3	2	1	0
ENT12PM	**ENT12OM**	**ENCC62F**	**ENCC62R**	**ENCC61F**	**ENCC61R**	**ENCC60F**	**ENCC60R**
rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENCC60R	0	rw	**Capture, Compare-Match Rising Edge Interrupt Enable for Channel 0** 0 No interrupt will be generated if the set condition for bit CC60R in register IS occurs. 1 An interrupt will be generated if the set condition for bit CC60R in register IS occurs. The interrupt line, which will be activated is selected by bitfield INPCC60.
ENCC60F	1	rw	**Capture, Compare-Match Falling Edge Interrupt Enable for Channel 0** 0 No interrupt will be generated if the set condition for bit CC60F in register IS occurs. 1 An interrupt will be generated if the set condition for bit CC60F in register IS occurs. The interrupt line, which will be activated is selected by bitfield INPCC60.
ENCC61R	2	rw	**Capture, Compare-Match Rising Edge Interrupt Enable for Channel 1** 0 No interrupt will be generated if the set condition for bit CC61R in register IS occurs. 1 An interrupt will be generated if the set condition for bit CC61R in register IS occurs. The interrupt line, which will be activated is selected by bitfield INPCC61.

Field	Bits	Type	Description
ENCC61F	3	rw	**Capture, Compare-Match Falling Edge Interrupt Enable for Channel 1** 0 No interrupt will be generated if the set condition for bit CC61F in register IS occurs. 1 An interrupt will be generated if the set condition for bit CC61F in register IS occurs. The interrupt line, which will be activated is selected by bitfield INPCC61.
ENCC62R	4	rw	**Capture, Compare-Match Rising Edge Interrupt Enable for Channel 2** 0 No interrupt will be generated if the set condition for bit CC62R in register IS occurs. 1 An interrupt will be generated if the set condition for bit CC62R in register IS occurs. The interrupt line, which will be activated is selected by bitfield INPCC62.
ENCC62F	5	rw	**Capture, Compare-Match Falling Edge Interrupt Enable for Channel 2** 0 No interrupt will be generated if the set condition for bit CC62F in register IS occurs. 1 An interrupt will be generated if the set condition for bit CC62F in register IS occurs. The interrupt line, which will be activated is selected by bitfield INPCC62.
ENT12OM	6	rw	**Enable Interrupt for T12 One-Match** 0 No interrupt will be generated if the set condition for bit T12OM in register IS occurs. 1 An interrupt will be generated if the set condition for bit T12OM in register IS occurs. The interrupt line, which will be activated is selected by bitfield INPT12.
ENT12PM	7	rw	**Enable Interrupt for T12 Period-Match** 0 No interrupt will be generated if the set condition for bit T12PM in register IS occurs. 1 An interrupt will be generated if the set condition for bit T12PM in register IS occurs. The interrupt line, which will be activated is selected by bitfield INPT12.

IENH

Capture/Compare Interrupt Enable Register ,High Byte **[Reset value: 00_H]**

7	6	5	4	3	2	1	0
-	ENIDLE	ENWHE	ENCHE	-	ENTRPF	ENT13PM	ENT13CM
r	rw	rw	rw	r	rw	rw	rw

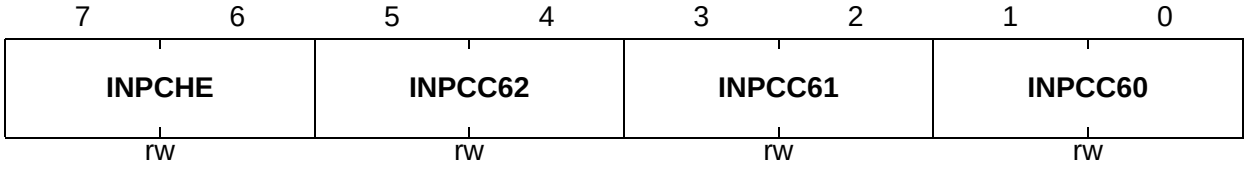
Field	Bits	Type	Description
ENT13CM	0	rw	**Enable Interrupt for T13 Compare-Match** 0 No interrupt will be generated if the set condition for bit T13CM in register IS occurs. 1 An interrupt will be generated if the set condition for bit T13CM in register IS occurs. The interrupt line, which will be activated is selected by bitfield INPT13.
ENT13PM	1	rw	**Enable Interrupt for T13 Period-Match** 0 No interrupt will be generated if the set condition for bit T13PM in register IS occurs. 1 An interrupt will be generated if the set condition for bit T13PM in register IS occurs. The interrupt line, which will be activated is selected by bitfield INPT13.
ENTRPF	2	rw	**Enable Interrupt for Trap Flag** 0 No interrupt will be generated if the set condition for bit TRPF in register IS occurs. 1 An interrupt will be generated if the set condition for bit TRPF in register IS occurs. The interrupt line, which will be activated is selected by bitfield INPERR.
ENCHE	4	rw	**Enable Interrupt for Correct Hall Event** 0 No interrupt will be generated if the set condition for bit CHE in register IS occurs. 1 An interrupt will be generated if the set condition for bit CHE in register IS occurs. The interrupt line, which will be activated is selected by bitfield INPCHE.

Field	Bits	Type	Description
ENWHE	5	rw	**Enable Interrupt for Wrong Hall Event** 0 No interrupt will be generated if the set condition for bit WHE in register IS occurs. 1 An interrupt will be generated if the set condition for bit WHE in register IS occurs. The interrupt line, which will be activated is selected by bitfield INPERR.
ENIDLE	6	rw	**Enable Idle** This bit enables the automatic entering of the idle state (bit CCU_IDLE will be set) after a wrong hall event has been detected (bit WHE is set). During the idle state, the bitfield MCMP is automatically cleared. 0 The bit CCU_IDLE is not automatically set when a wrong hall event is detected. 1 The bit CCU_IDLE is automatically set when a wrong hall event is detected.
-	3, 7	r	**reserved;** returns ’0’ if read; should be written with ’0’;

Registers INPL and INPH contains the interrupt node pointer bits, allowing for flexible interrupt handling.

INPL

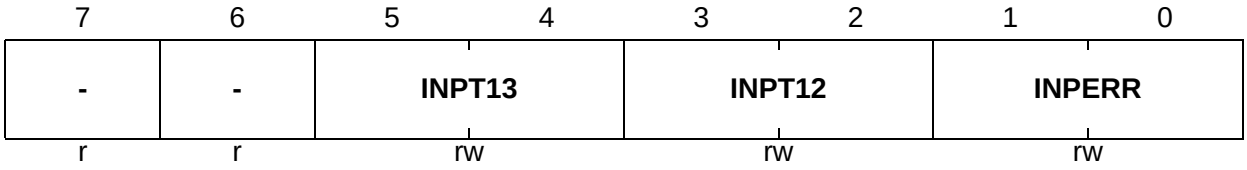
Capture/Compare Interrupt Node Pointer Register ,Low Byte [Reset value: 40_H]



Field	Bits	Type	Description
INPCC60	[1:0]	rw	**Interrupt Node Pointer for Channel 0 Interrupts** This bitfield defines the interrupt node, which is activated due to a set condition for bit CC60R (if enabled by bit ENCC60R) or for bit CC60F (if enabled by bit ENCC60F). 00 Interrupt node I0 is selected. 01 Interrupt node I1 is selected. 10 Interrupt node I2 is selected. 11 Interrupt node I3 is selected.
INPCC61	[3:2]	rw	**Interrupt Node Pointer for Channel 1 Interrupts** This bitfield defines the interrupt node, which is activated due to a set condition for bit CC61R (if enabled by bit ENCC61R) or for bit CC61F (if enabled by bit ENCC61F). 00 Interrupt node I0 is selected. 01 Interrupt node I1 is selected. 10 Interrupt node I2 is selected. 11 Interrupt node I3 is selected.
INPCC62	[5:4]	rw	**Interrupt Node Pointer for Channel 2 Interrupts** This bitfield defines the interrupt node, which is activated due to a set condition for bit CC62R (if enabled by bit ENCC62R) or for bit CC62F (if enabled by bit ENCC62F). 00 Interrupt node I0 is selected. 01 Interrupt node I1 is selected. 10 Interrupt node I2 is selected. 11 Interrupt node I3 is selected.
INPCHE	[7:6]	rw	**Interrupt Node Pointer for the CHE Interrupt** This bitfield defines the interrupt node, which is activated due to a set condition for bit CHE (if enabled by bit ENCHE). 00 Interrupt node I0 is selected. 01 Interrupt node I1 is selected. 10 Interrupt node I2 is selected. 11 Interrupt node I3 is selected.

INPH

Capture/Compare Interrupt Node Pointer Register ,High Byte [Reset value: 39_H]

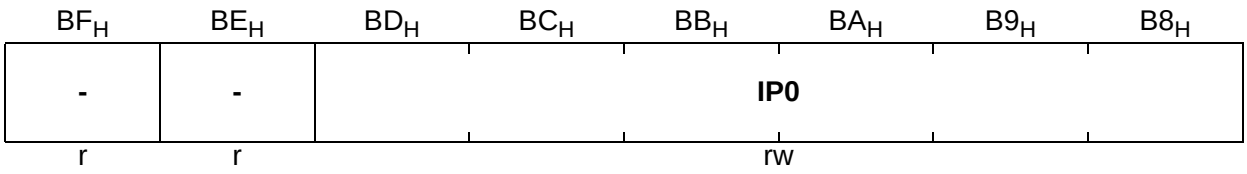


Field	Bits	Type	Description
INPERR	[1:0]	rw	**Interrupt Node Pointer for Error Interrupts** This bitfield defines the interrupt node, which is activated due to a set condition for bit TRPF (if enabled by bit ENTRPF) or for bit WHE (if enabled by bit ENWHE). 00 Interrupt node I0 is selected. 01 Interrupt node I1 is selected. 10 Interrupt node I2 is selected. 11 Interrupt node I3 is selected.
INPT12	[3:2]	rw	**Interrupt Node Pointer for Timer12 Interrupts** This bitfield defines the interrupt node, which is activated due to a set condition for bit T12OM (if enabled by bit ENT12OM) or for bit T12PM (if enabled by bit ENT12PM). 00 Interrupt node I0 is selected. 01 Interrupt node I1 is selected. 10 Interrupt node I2 is selected. 11 Interrupt node I3 is selected.
INPT13	[5:4]	rw	**Interrupt Node Pointer for Timer13 Interrupt** This bitfield defines the interrupt node, which is activated due to a set condition for bit T13CM (if enabled by bit ENT13CM) or for bit T13PM (if enabled by bit ENT13PM). 00 Interrupt node I0 is selected. 01 Interrupt node I1 is selected. 10 Interrupt node I2 is selected. 11 Interrupt node I3 is selected.
-	[7:6	r	**reserved;** returns ’0’ if read; should be written with ’0’;

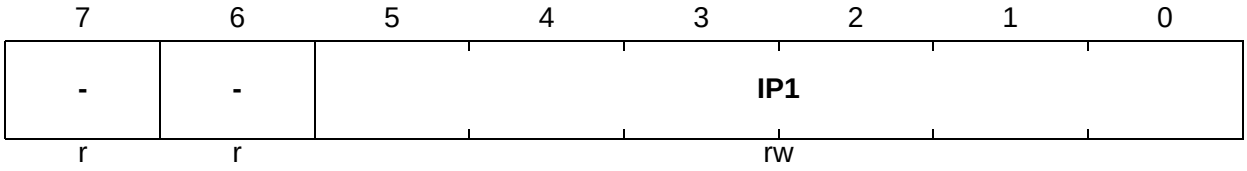
7.2.4 Interrupt Priority Register

The lower six bits of these two registers are used to define the interrupt priority level of the interrupt groups as they are defined in **Table 7-2** in the next section.

IP0
Interrupt Priority Register 0 **[Reset value: $XX000000_B$]**



IP1
Interrupt Priority Register 1 **[Reset value: $XX000000_B$]**



Field	Bits	Typ	Description
[IP1,IP0x] **x=0..5**	[5..0]	rw	**Interrupt group priority level bits** 00 Interrupt group x is set to priority level 0 (lowest) 01 Interrupt group x is set to priority level 1 10 Interrupt group x is set to priority level 2 11 Interrupt group x is set to priority level 3 (highest)
-	[7:6]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

7.3 Interrupt Priority Level Structure

The 13 interrupt sources of C868 are grouped according to the listing in **Table 7-1**.

Table 7-1 Interrupt Source Structure

Interrupt Group	Associated Interrupts			
0	External interrupt 0	A/D converter interrupt		
1	Timer 0 Overflow	External interrupt 2		
2	External interrupt 1	External interrupt 3	Capture/compare interrupt node 0	
3	Timer 1 overflow		Capture/compare interrupt node 1	
4	Serial interrupt		Capture/compare interrupt node 2	
5	Timer 2 overflow		Capture/compare interrupt node 3	

Each group of interrupt sources can be programmed individually to one of four priority levels by setting or clearing the bits in the special function registers IP1 and IP0. A low-priority interrupt can be interrupted by a high-priority interrupt, but not by another interrupt of the same or a lower priority. An interrupt of the highest priority level cannot be interrupted by another interrupt source.

If two or more requests of different priority levels are received simultaneously, the request of the highest priority is serviced first. If requests of the same priority level are received simultaneously, an internal polling sequence determines which request is to be serviced first. Thus, within each priority level there is a second priority structure determined by the polling sequence. This is illustrated in **Table 7-2**.

Table 7-2 Interrupt Source Structure

Interrupt Group	Priority Bits of Interrupt Group	Interrupt Source Priority High Priority → Low Priority				Priority
0	IP1.0, IP0.0	EXINT0	IADC			High ↓
1	IP1.1, IP0.1	TF0	EXINT2			
2	IP1.2, IP0.2	EXINT1	EXINT3	INP0[1)]		
3	IP1.3, IP0.3	TF1		INP1[1)]		
4	IP1.4, IP0.4	RI + TI		INP2[1)]		Low
5	IP1.5, IP0.5	TF2+EXF2		INP3[1)]		

1) Capture/compare has 10 interrupt sources channeled to the 4 interrupt nodes INP0..3. The 3 capture/compare ports has 3 pairs of interrupt request flags, ICC60R, ICC60F, ICC61R, ICC61F, ICC62R, ICC62F. The other flags are T12OM, T12PM, T13CM, T13PM, TRPF, WHE, CHE.

Within a column, the topmost interrupt is serviced first, then the second and the third, when available. The interrupt groups are serviced from left to right of the table. A low-priority interrupt can itself be interrupted by a higher-priority interrupt, but not by another interrupt of the same or a lower priority. An interrupt of the highest priority level cannot be interrupted by another interrupt source.

If two or more requests of different priority levels are received simultaneously, the request of the highest priority is serviced first. If requests of the same priority level are received simultaneously, an internal polling sequence determines which request is to be serviced first. Thus, within each priority level there is a second priority structure which is illustrated in table 7-10.

The “priority-within-level” structure is only used to resolve simultaneous requests of the same priority level.

7.4 How Interrupts are Handled

The interrupt flags are sampled at S5P2 in each machine cycle. The sampled flags are polled during the following machine cycle. If one of the flags was in a set condition at S5P2 of the preceeding cycle, the polling cycle will find it and the interrupt system will generate a LCALL to the appropriate service routine, provided this hardware-generated LCALL is not blocked by any of the following conditions:

An interrupt of equal or higher priority is already in progress.

The current (polling) cycle is not in the final cycle of the instruction in progress.

The instruction in progress is RETI or any write access to registers IEN0/IEN1 or IP0.

Any of these three conditions will block the generation of the LCALL to the interrupt service routine. Condition 2 ensures that the instruction in progress is completed before vectoring to any service routine. Condition 3 ensures that if the instruction in progress is RETI or any write access to registers IEN0/IEN1 or IP0, then at least one more instruction will be executed before any interrupt is vectored to; this delay guarantees that changes of the interrupt status can be observed by the CPU.

The polling cycle is repeated with each machine cycle, and the values polled are the values that were present at S5P2 of the previous machine cycle. Note that if any interrupt flag is active but not being responded to for one of the conditions already mentioned, or if the flag is no longer active when the blocking condition is removed, the denied interrupt will not be serviced. In other words, the fact that the interrupt flag was once active but not serviced is not remembered. Every polling cycle interrogates only the pending interrupt requests.

The polling cycle/LCALL sequence is illustrated in **Table 7-7**.

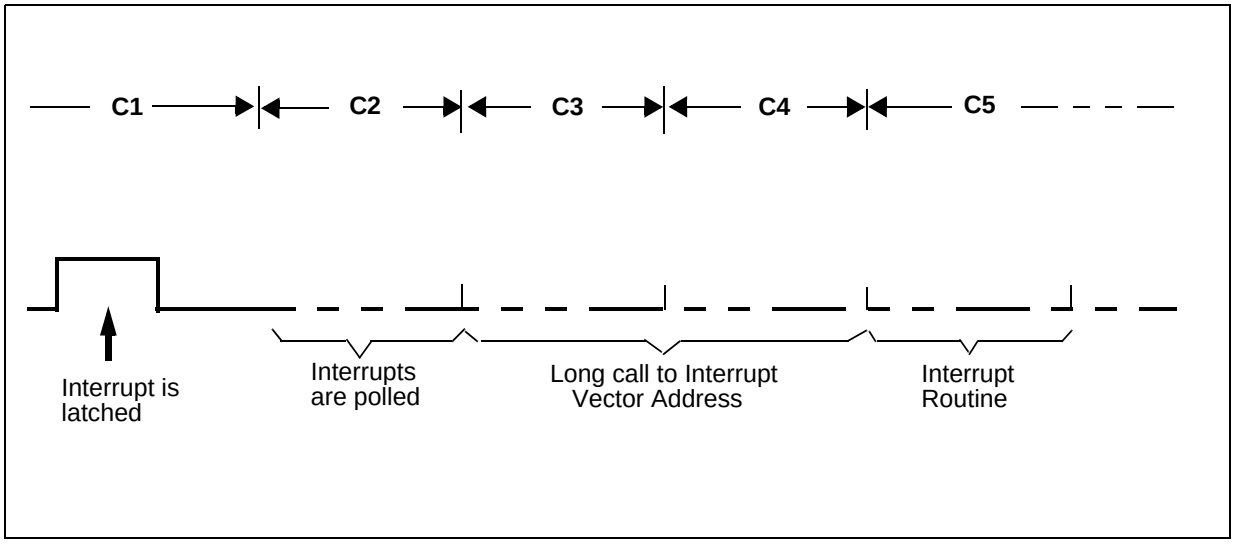


Figure 7-7 Interrupt Response Timing Diagram

Note that if an interrupt of a higher priority level goes active prior to S5P2 in the machine cycle labeled C3 in **Figure 7-7** then, in accordance with the above rules, it will be vectored to during C5 and C6 without any instruction for the lower priority routine to be executed.

Thus, the processor acknowledges an interrupt request by executing a hardware-generated LCALL to the appropriate servicing routine. In some cases it also clears the flag that generated the interrupt, while in other cases it does not; then this has to be done by the user's software. The hardware clears the external interrupt flags IE0 and IE1 only if they were transition-activated. The hardware-generated LCALL pushes the contents of the program counter onto the stack (but it does not save the PSW) and reloads the program counter with an address that depends on the source of the interrupt being vectored to, as shown in the following **Table 7-3**.

Table 7-3 Interrupt Source and Vectors

Interrupt Source	Interrupt Vector Address(core connections)	Interrupt Request Flags
External Interrupt 0	0003_H(EX0)	IE0
Timer 0 Overflow	$000B_H$(ET0)	TF0
External Interrupt 1	0013_H(EX1)	IE1
Timer 1 Overflow	$001B_H$(ET1)	TF1
Serial Channel	0023_H(ES)	RI / TI
Timer 2 Overflow	$002B_H$(EX5)	TF2+EXF2
A/D Converter	0033_H(EX6)	IADC
External Interrupt 2	$003B_H$(EX7)	IEX2
External Interrupt 3	0043_H(EX8)	IEX3
	$004B_H$(EX9)	
	0053_H(EX10)	
	$005B_H$(EX11)	
	0063_H(EX12)	
	$006B_H$(EX13)	
CCU6 interrupt node 0	0083_H(EX14)	INP0[1)]
CCU6 interrupt node 1	$008B_H$(EX15)	INP1[1)]
CCU6 interrupt node 2	0093_H(EX16)	INP2[1)]
CCU6 interrupt node3	$009B_H$(EX17)	INP3[1)]
	$00A3_H$(EX18)	

Table 7-3 Interrupt Source and Vectors

	$00AB_H$(EX19)	
	$00D3_H$(EX20)	
	$00DB_H$(EX21)	
	$00E3_H$(EX22)	
Wake-up from power-down mode	$007B_H$	–

1) Capture/compare has 10 interrupt sources channeled to the 4 interrupt nodes INP0..3. The 3 capture/compare ports has 3 pairs of interrupt request flags, ICC60R, ICC60F, ICC61R, ICC61F, ICC62R, ICC62F. The other flags are T12OM, T12PM, T13CM, T13PM, TRPF, WHE, CHE.

Execution proceeds from that location until the RETI instruction is encountered. The RETI instruction informs the processor that the interrupt routine is no longer in progress, then pops the two top bytes from the stack and reloads the program counter. Execution of the interrupted program continues from the point where it was stopped. Note that the RETI instruction is very important because it informs the processor that the program left the current interrupt priority level. A simple RET instruction would also have returned execution to the interrupted program, but it would have left the interrupt control system thinking an interrupt was still in progress. In this case no interrupt of the same or lower priority level would be acknowledged.

External Interrupts

The external interrupts 0 and 1 can be programmed to be level-activated or negative-transition activated by setting or clearing bit ITx (x = 0 or 1), respectively in register TCON. If ITx = 0, external interrupt x is triggered by a detected low level at the INTx pin. If ITx = 1, external interrupt x is negative edge-triggered. In this mode, if successive samples of the INTx pin show a high in one cycle and a low in the next cycle, interrupt request flag IEx in TCON is set. Flag bit IEx=1 then requests the interrupt.

If the external interrupt 0 or 1 is level-activated, the external source has to hold the request active until the requested interrupt is actually generated. Then it has to deactivate the request before the interrupt service routine is completed, or else another interrupt will be generated.

The external interrupts 2 and 3 can be programmed to be negative or positive transition-activated by setting or clearing bits I2FR or I3FR in register T2CON. If IxFR = 0 (x = 2 or 3) then the external interrupt x is negative transition-activated. If IxFR = 1, external interrupt is triggered by a positive transition.

Since the external interrupt pins are sampled once in each machine cycle, an input high or low should be held for at least 3 oscillator periods to ensure sampling. If the external interrupt is positive (negative) transition-activated, the external source has to hold the request pin low (high) for at least one cycle, and then hold it high (low) for at least one

cycle to ensure that the transition is recognized so that the corresponding interrupt request flag will be set (see **Figure 7-8**). The external interrupt request flags will automatically be cleared by the CPU when the service routine is called.

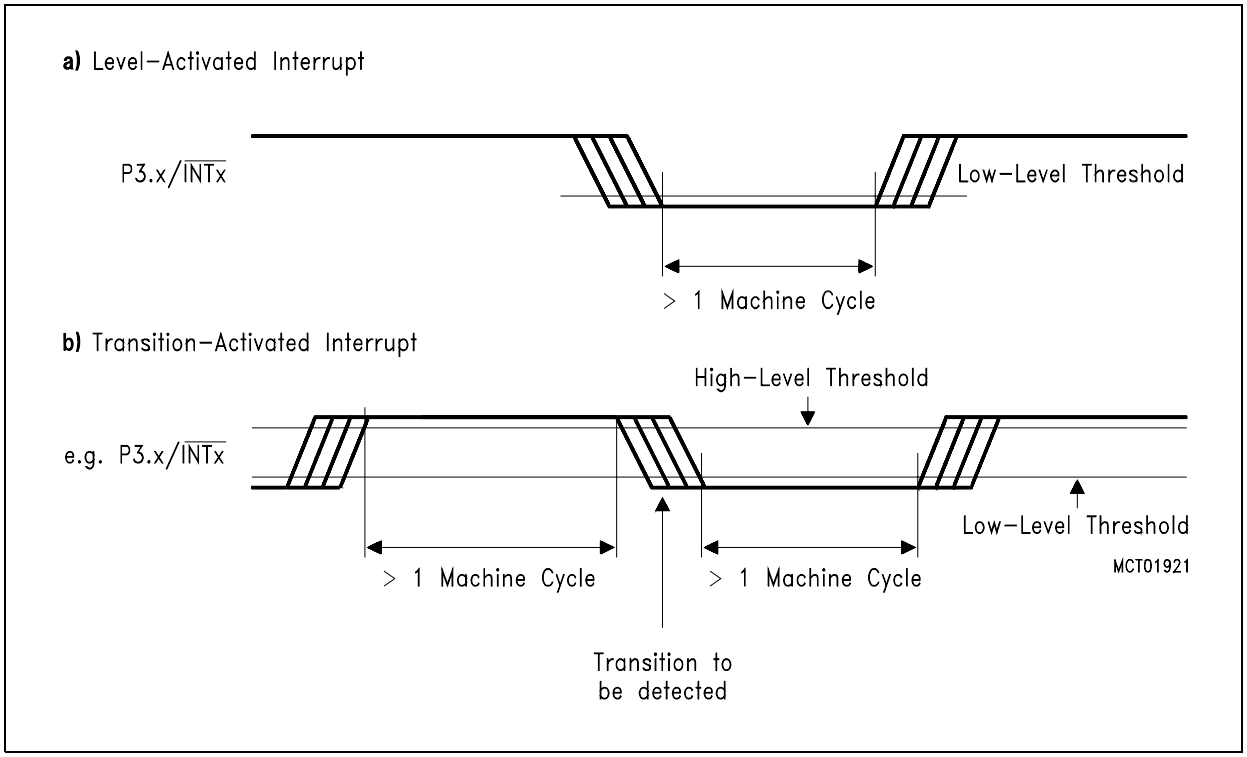


Figure 7-8 External Interrupt Detection

7.5 Interrupt Response Time

If an external interrupt is recognized, its corresponding request flag is set at S5P2 in every machine cycle. The value is not polled by the circuitry until the next machine cycle. If the request is active and conditions are right for it to be acknowledged, a hardware subroutine call to the requested service routine will be the next instruction to be executed. The call itself takes two cycles. Thus a minimum of three complete machine cycles will elapse between activation and external interrupt request and the beginning of execution of the first instruction of the service routine.

A longer response time would be obtained if the request was blocked by one of the three previously listed conditions. If an interrupt of equal or higer priority is already in progress, the additional wait time obviously depends on the nature of the other interrupt's service routine. If the instruction in progress is not in its final cycle, the additional wait time cannot be more than 3 cycles since the longest instructions (MUL and DIV) are only 4 cycles long; and, if the instruction in progress is RETI or a write access to registers IEN0, IEN1 or IP0 the additional wait time cannot be more than 5 cycles (a maximum of one more

cycle to complete the instruction in progress, plus 4 cycles to complete the next instruction, if the instruction is MUL or DIV).

Thus a single interrupt system, the response time is always more than 3 cycles and less than 9 cycles.

8 Power Saving Modes

The C868 provides two basic power saving modes, the idle mode and the power down mode. Additionally, a slow down mode is available. This power saving mode reduces the internal clock rate in normal operating mode and it can also be used for further power reduction in idle mode. Further power saving is possible in the normal, idle and slow down modes by disabling unutilized peripherals. The peripherals that has the power-down capability are the timer/counter2, capture/compare unit and the A/D converter.

8.1 Power Saving Mode Control Registers

The functions of the power saving modes are controlled by bits which are located in the special function registers PCON and PMCON0.

The bits PDE and IDLE located in SFR PCON select the power down mode or the idle mode, respectively. If the power down mode and the idle mode are set at the same time, power down mode takes precedence. Furthermore, register PCON contains two general purpose flags. For example, the flag bits GF0 and GF1 can be used to give an indication if an interrupt occurred during normal operation or during idle mode. For this, an instruction that activates idle mode can also set one or both flag bits. When idle mode is terminated by an interrupt, the interrupt service routine can examine the flag bits.

8.2 Register Description

PCON
Power Control Register **[Reset value: $0XXX0000_B$]**

7	6	5	4	3	2	1	0
SMOD	**-**	**-**	**SD**	**GF1**	**GF0**	**PDE**	**IDLE**
rw	r	r	rw	rw	rw	rw	rw

The functions of the shaded bits are not described here

Field	Bits	Typ	Description
IDLE	0	rw	**Idle mode enable bit** When set, starting of the idle mode is enabled
PDE	1	rw	**Power down enable bit** When set, starting of the power down is enabled
GF0	2	rw	**General purpose flag**
GF1	3	rw	**General purpose flag**
SD	4	rw	**Slow down mode bit** When set, the slow down mode is enabled
-	[6:5]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

PMCON0
Wake-up Control Register **[Reset value: $XXX000000_B$]**

7	6	5	4	3	2	1	0
-	-	-	EBO	BO	SDSTAT	WS	EWPD
r	r	r	rw	rw	rh	rw	rw

The functions of the shaded bits are not described here

Field	Bits	Typ	Description
EWPD	0	rw	**Enable Wake Up from Power Down Mode.** When this bit is set, power down mode can be terminated either by an external active INT0 signal or by an external signal from a second source, RXD.
WS	1	rw	**Wake Up Source Select.** This is applicable only if EWPD is set. 0 : INT0 will terminate power down mode. (default) 1 : RXD will terminate power down mode.
SDSTAT	2	rh	**Slow Down Status Bit.** 0 : Slow Down Mode is not active, ie system clock is not slow down clock. 1 : Slow Down Mode is active, ie system clock is the slow down clock. This bit is set or cleared by hardware only.
-	[7:5]	r	**reserved;** returns ’0’ if read; should be written with ’0’;

8.3 Idle Mode

In the idle mode the oscillator of the C868 continues to run, but the CPU is gated off from the clock signal. However, the interrupt system, the serial port, the A/D converter, the capture/compare unit, and all timers with the exception of the watchdog timer are further provided with the clock. The CPU status is preserved in its entirety: the stack pointer, program counter, program status word, accumulator, and all other registers maintain their data during idle mode.

The reduction of power consumption, which can be achieved by this feature depends on the number of peripherals running. If all timers are stopped and the A/D converter, and the serial interfaces are not running, the maximum power reduction can be achieved. This state is also the test condition for the idle mode I_{DDP}.

Thus, the user has to take care which peripheral should continue to run and which has to be stopped during idle mode. Also the state of all port pins – either the pins controlled by their latches or controlled by their secondary functions – depends on the status of the controller when entering idle mode.

Normally, the port pins hold the logical state they had at the time when the idle mode was activated. If some pins are programmed to serve as alternate functions they still continue to output during idle mode if the assigned function is on. This especially applies to the serial interface in case it cannot finish reception or transmission during normal operation.

As in normal operation mode, the ports can be used as inputs during idle mode. Thus a capture or reload operation can be triggered, the timers can be used to count external events, and external interrupts will be detected.

The idle mode is a useful feature which makes it possible to "freeze" the processor's status - either for a predefined time, or until an external event reverts the controller to normal operation, as discussed below. The watchdog timer is the only peripheral which is automatically stopped during idle mode.

The idle mode is entered by setting the flag bit IDLE (PCON.0).

There are two ways to terminate the idle mode:

The idle mode can be terminated by activating any enabled interrupt. The CPU operation is resumed, the interrupt will be serviced and the next instruction to be executed after the RETI instruction will be the one following the instruction that had set the bit IDLE.

The other way to terminate the idle mode, is a hardware reset.

8.4 Slow Down Mode Operation

In some applications, where power consumption and dissipation are critical, the controller might run for a certain time at reduced speed (e.g. if the controller is waiting for an input signal). Since in CMOS devices there is an almost linear dependence of the operating frequency and the power supply current, a reduction of the operating frequency results in reduced power consumption.

The slow down mode is activated by setting the bit SD in SFR PCON. If the slow down mode is enabled, the clock signals for the CPU and the peripheral units can be reduced from 1/2 to 1/32 of the nominal system clock rate. The clock divider is described in the Reset and System Clock Operation chapter. The controller actually enters the slow down mode after a short synchronization period (max. two machine cycles). The slow down mode is terminated by clearing bit SD.

The slow down mode can be combined with the idle mode by setting the IDLE and SD bits in SFR PCON.

There are two ways to terminate the combined Idle and Slow Down Mode :

– The idle mode can be terminated by activation of any enabled interrupt. The CPU operation is resumed, the interrupt will be serviced and the next instruction to be executed after the RETI instruction will be the one following the instruction that had set the bits IDLE and SD. Nevertheless the slow down mode keeps enabled and if required has to be terminated by clearing the bit SD in the corresponding interrupt service routine or at any point in the program where the user no longer requires the slow-down mode power saving.
– The other possibility of terminating the combined idle and slow down mode is a hardware reset.

8.5 Software Power Down Mode

In the software power down mode, the on-chip oscillator which operates with the XTAL pins and the PLL are all stopped. Therefore, all functions of the microcontroller are stopped and only the contents of the on-chip RAM, XRAM and the SFR's are maintained. The port pins, which are controlled by their port latches, output the values that are held by their SFR's. The port pins which serve the alternate output functions show the values they had at the end of the last cycle of the instruction which initiated the power down mode. ALE is held at logic level high unless it is disabled.

In the power down mode of operation, V_{DDP} can be reduced to minimize power consumption. It must be ensured, however, that V_{DDP} is not reduced before the power down mode is invoked, and that V_{DDP} is restored to its normal operating level before the power down mode is terminated.

The software power down mode can be left either by an active reset signal or by a low signal at one of the wake-up source pins. Using reset to leave power down mode puts the microcontroller with its SFRs into the reset state. Using either the $\overline{\text{INT0}}$ pin or the RXD pin for power down mode exit starts the on-chip oscillator and the PLL and maintains the state of the SFRs, which have been frozen when power down mode is entered. Leaving power down mode should not be done before V_{DDC} and V_{DDP} is restored to its nominal operating level.

The software power down mode is entered by setting bit PDE (PCON.1).

Note: Before entering the power down mode, an A/D conversion in progress must be stopped.

8.5.1 Exit from Software Power Down Mode

If power down mode is exit via a hardware reset, the microcontroller with its SFRs is put into the hardware reset state and the content of RAM and XRAM are not changed. The reset signal that terminates the power down mode also restarts the on-chip oscillator and the PLL. The reset operation should not be activated before V_{DDP} is restored to its normal operating level.

Figure 8-1 shows the behaviour when power down mode is left via the $\overline{\text{INT0}}$ or the RXD wake-up capability.

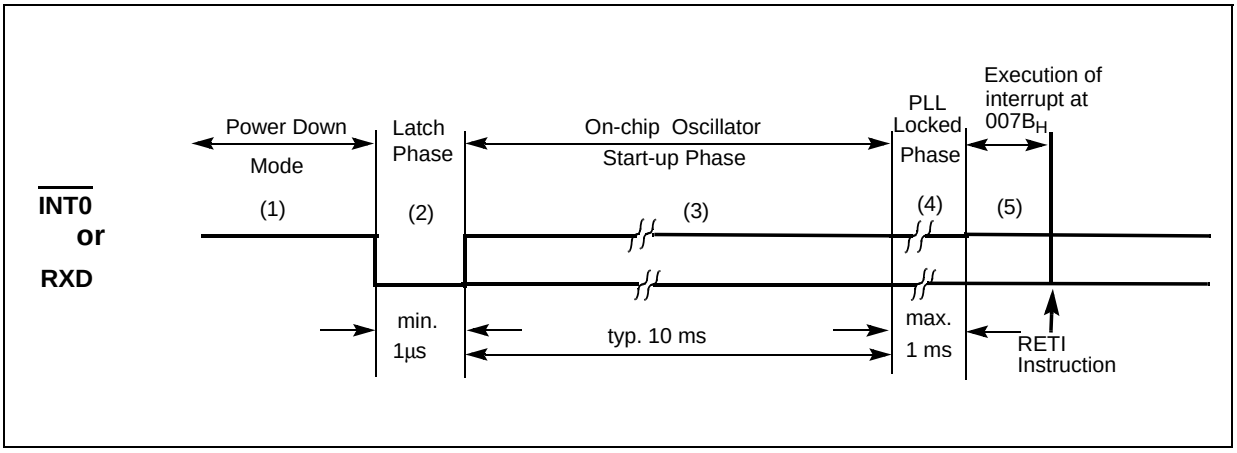


Figure 8-1 Wake-up from Power Down Mode Procedure

When the power down mode wake-up capability has been enabled (bit EWPD in SFR PMCON0 set) prior to entering power down mode and bit WS in SFR PMCON0 is cleared, the power down mode can be exit via $\overline{\text{INT0}}$ while executing the following procedure :

1. In power down mode pin $\overline{\text{INT0}}$ must be held at high level.
2. Power down mode is left when $\overline{\text{INT0}}$ goes low(latch phase). After this delay the on-chip oscillator and the PLL are started, the state of pin $\overline{\text{INT0}}$ is internally latched, and $\overline{\text{INT0}}$ can be set again to high level if required.
3. The on-chip oscillator takes about, typically, 10 ms to stabilize.
4. The PLL will be locked within 1 ms after the on-chip oscillator clock is detected for stable nominal frequency. Subsequently, the microcontroller starts again with its operation initiating the power down wake-up interrupt. The interrupt address of the first instruction to be executed after wake-up is 007B$_H$. Instruction fetches during the interrupt call are, however, discarded.
5. After the RETI instruction of the power down wake-up interrupt routine has been executed, the instruction which follows the initiating power down mode instruction sequence will be executed.

All interrupts of the C868 are disabled from phase 2 until the end of phase 5. Other Interrupts can be first handled after the RETI instruction of the wake-up interrupt routine.

The procedure to exit the software power down mode via the RXD pin is identical to the above procedure except that in this case pin RXD replaces pin $\overline{\text{INT0}}$, and bit WS in SFR PCON1 should be set prior to entering software power down mode.

9 The Bootstrap Loader

The C868, includes a bootstrap mode, which is activated by setting the ALE/BSL pin at logic low with a pulldown and TxD pin at logic high with a pullup at the rising edge of the RESET. Or it can be entered by software, that is by setting BSLEN bit and resetting SWAP bit in SFR SYSCON1 accompany by an unlock sequence, the details can be found in **Chapter 3.3.2**.

In the bootstrap mode, software routines of the bootstrap loader located in the boot ROM will be executed. Its purpose is to allow the easy and quick programming of the internal SRAM (0000_H to $1FFF_H$) or XRAM ($FF00_H$ to $FFFF_H$) via serial interface (UART) while the MCU is in-circuit. It also provides a way to program SRAM or XRAM through bootstrapping from an external SPI or I2C EEPROM.

The first action of the bootstrap loader is to detect the presence of EEPROM and its type, SPI or I2C, and check the first byte of the serial EEPROM. If the first byte is $0A5_H$, the MCU would enter Phase A to download from the EEPROM. Otherwise, it will enter Phase B to establish a serial communication with the connected host. Bootstrapping from the serial EEPROM can also be done in phase B if it is invoked by the host.

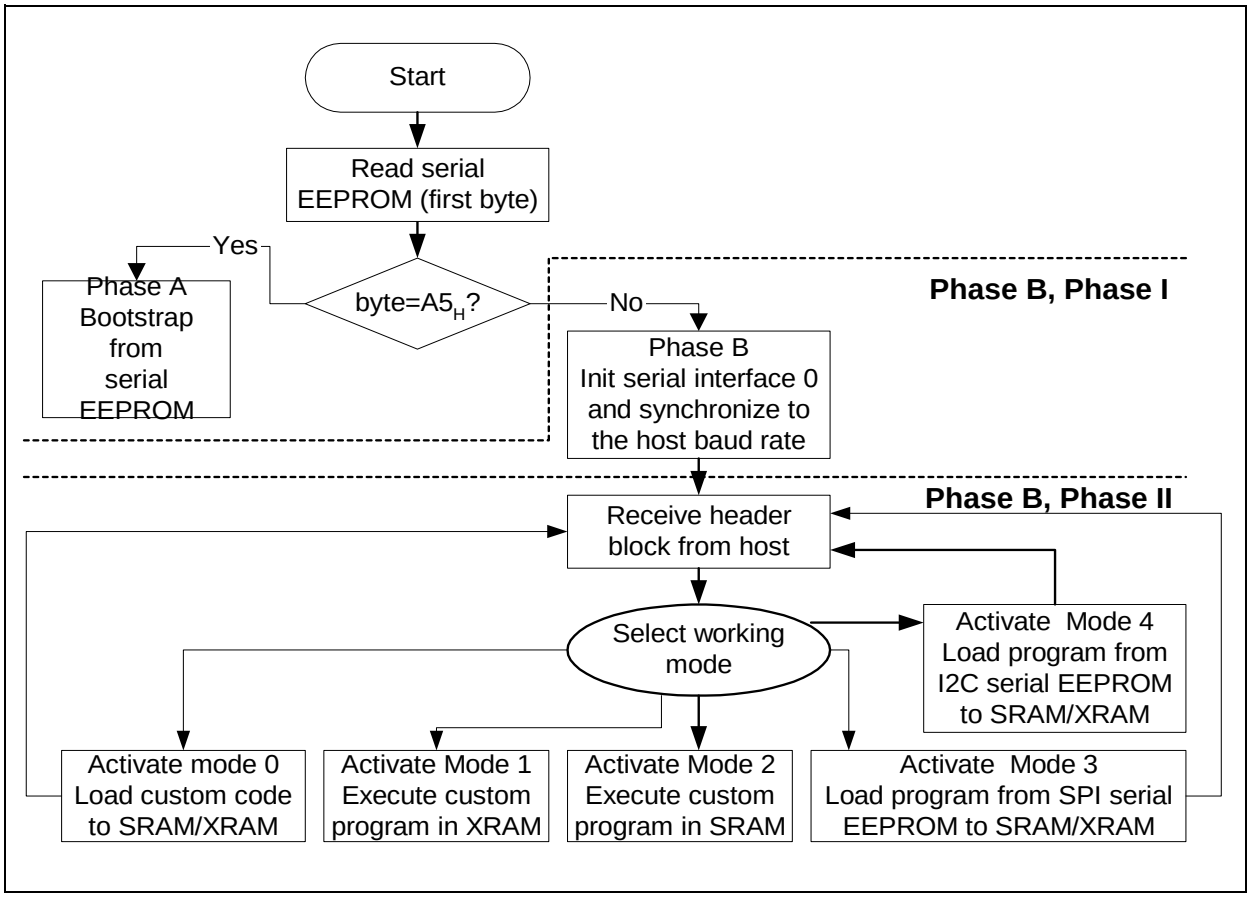


Figure 9-1 The phases of the Bootstrap Loader

9.1 Bootstrap from Serial EEPROM

Circuit connections for SPI and I2C EEPROM are shown in **Figure 9-2**.

9.1.1 Data Format of Serial EEPROM

The bootstrap loader accesses the serial EEPROM in a page of 32 bytes at a time. Take the SPI EEPROM AT25640 for example. The size is 8K bytes, so the total number of pages is 8K/32 = 256. The serial EEPROM may contain one or more transfer blocks of data, each containing up to 256 pages, and a download process may involve more than one transfer blocks. Each transfer block has an 8 byte header in the first page. The remainder of the first page and the rest of the pages in the transfer block contains the data. A transfer block has the following structure:

Table 9-1 First Page of a Transfer Block

Byte	Description
0	**Password** If the password is $0A5_H$, the MCU would enter Phase A (downloading from the EEPROM). Otherwise, it would enter Phase B (serial communication with the host). This password would only be checked for the first transfer block and would be ignored for the subsequent transfer blocks
1	**Page-count** This byte indicates the length of this transfer block. It ranged from 0 to 255.
2	**Last** If Last is 00_H, the current transfer block is the last transfer block to be downloaded. Otherwise, the current transfer block is not the last block to be downloaded.
3	**Download Address, high byte** The most significant byte of the start address of XRAM or SRAM where the data of EEPROM should be downloaded. Noted that we can determine whether the destination is XRAM or SRAM by comparing this byte with $0FF_H$.
4	**Download Address, low byte** The least significant byte of the start address of XRAM or SRAM where the data of EEPROM should be downloaded.

Table 9-1 First Page of a Transfer Block

Byte	Description
5	**Jump Address, high byte** The most significant byte of the address at which the C868 would jump to after the downloading process. This byte indicates whether to jump to XRAM or SRAM and the respective software unlock sequence and chip mode conversion should be invoked before jumping. Note that only the Jump Address of the last transfer block is effective.
6	**Jump Address, low byte** The least significant byte of the address at which the C868 would jump to after the downloading process. Note that only the Jump Address of the last transfer block is effective.
7	**Checksum** This byte is the checksum of the previous 7 bytes of the header block. It is obtained by XORING the previous 7 bytes.
8-31	**Data** Data to be written to the XRAM/SRAM. The header will not be written to XRAM/SRAM.

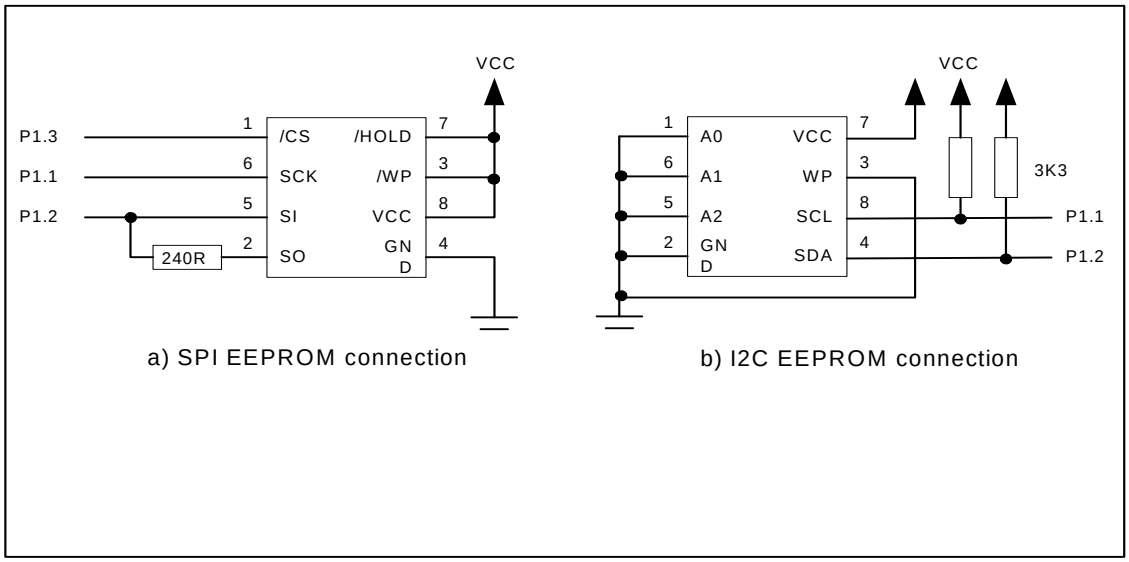


Figure 9-2 EEPROM connections for a) SPI and b) I2C

9.1.2 Download Process

Once phase A is entered, the process of data download from the serial EEPROM will commence. When a transfer block with Last = 00_H is encountered, it would jump to the Jump Address specified in this transfer block after downloading.

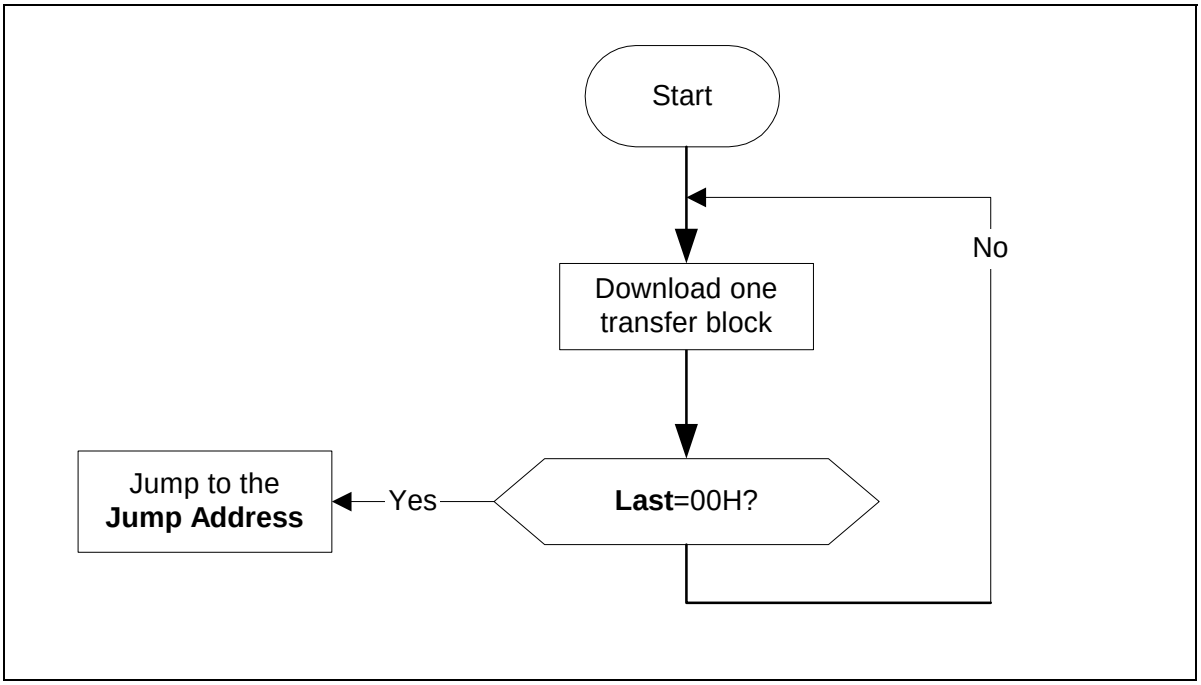


Figure 9-3 Flowchart for the download process

9.2 Serial Communication through the UART

Phase B consists of two functional parts that represent two phases:

- Phase I: Establish a serial connection and automatically synchronize to the transfer speed (baud rate) of the serial communication partner (host).
- Phase II: Perform the serial communication with the host. The host controls the communication by sending special header information, which select one of the working modes. These modes are:

Table 9-2 Serial Communication Modes of Phase B

Modes	Description
0	Transfer a customer program from the host to the SRAM (0000_H to $1FFF_H$) or XRAM ($FF00_H$ -$FFFF_H$). Then return to the beginning of phase II and wait for the next command from the host.
1	Execute a customer program in the XRAM at start address $FF00_H$.
2	Execute a customer program in the SRAM at start address 0000_H.
3	Transfer a customer program from the SPI EEPROM to the SRAM (0000_H to $1FFF_H$) or XRAM ($FF00_H$ -$FFFF_H$). Then return to the beginning of phase II and wait for the next command from the host.
4	Transfer a customer program from the I2C EEPROM to the SRAM (0000_H to $1FFF_H$) or XRAM ($FF00_H$ -$FFFF_H$). Then return to the beginning of phase II and wait for the next command from the host.
5-9	reserved

The serial communication, which is activated in phase II, is performed with the full-duplex serial interface (UART) of the C868. The MCU is connected to the serial port of the host via a serial cable (RS232).

The serial transfer is working in asynchronous mode with the serial parameters 8N1 (eight data bits, no parity and one stop bit). The host can vary the baud rate in a wide range because the MCU does an automatic synchronization with the host in phase I.

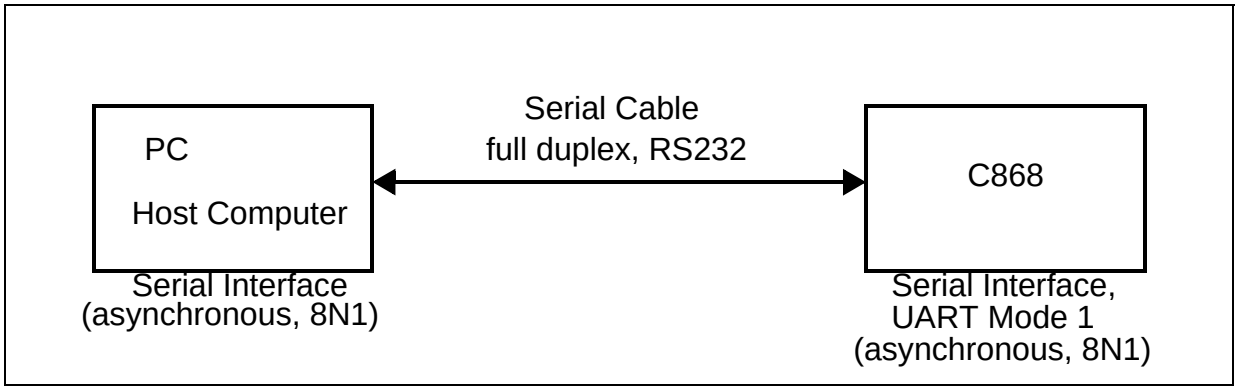


Figure 9-4 Bootstrap Loader Interface to the PC

9.2.1 Phase I: Automatic Serial Synchronization to the Host

The first action of the bootstrap loader is to synchronize the baud rate between the MCU and host. It is performed in the following steps.

STEP 1: Initialize serial interface for reception and Timer 2 for baud rate measurement.

STEP 2: Wait for test byte (80_H) from the host

STEP 3: Synchronize the baud rate to the host

STEP 4: Send Acknowledge byte (55_H) to the host

STEP 5: Enter Phase II.

The serial port of the bootstrap loader should be set to Mode 1 (8-bit UART, variable baud rate) for communication. Timer 2 is in auto-reload mode (16-bit timer) for baud rate measurement. A test byte (80_H) from the host is captured by starting the timer on reception of the start bit (0) and stopping it on reception of the last bit of the test byte (1). From the captured timer value, the bootstrap loader calculates the actual baud rate and activates Timer 2 as the baud rate generator of the serial interface. When the synchronization is done, the bootstrap loader sends back the Acknowledge byte (55_H) to the host and enter phase II. If the communication is not established due to difference in the actual and calculated baudrate, a reset to the C868 has to be activated to restart the device for a new synchronization attempt.

9.2.1.1 Calculation of Timer 2 Reload Value

By polling the receive port of the serial interface (P1.4/RxD Pin), the bootstrap loader measures the low period of the test byte by using timer 2 as shown in **Figure 9-5**.

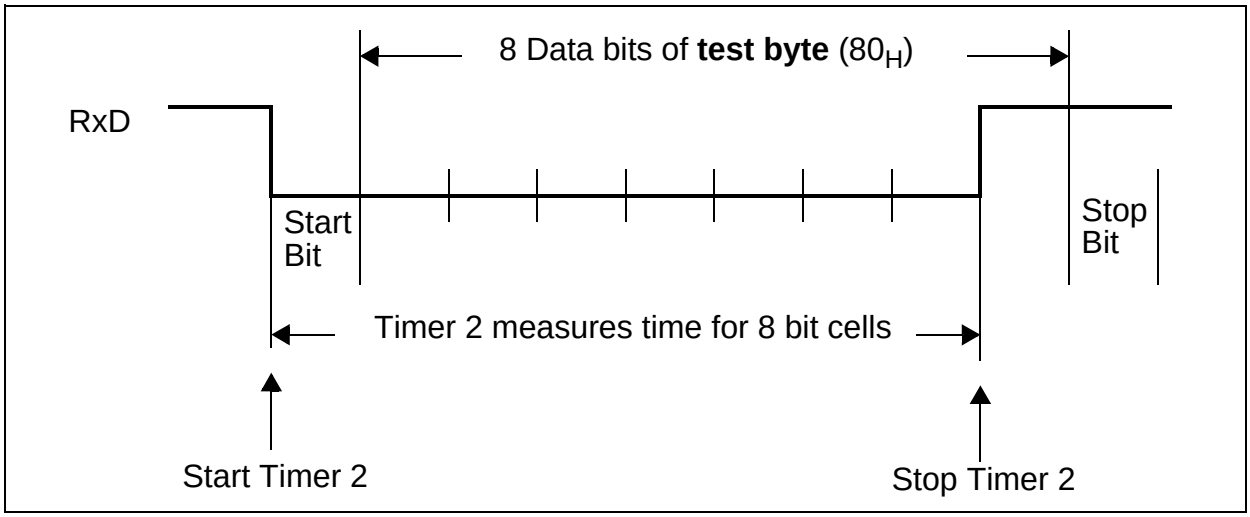


Figure 9-5 Measuring the received time of a testbyte by using timer 2

The time recorded is the receiving time of 8 bits (1 start bit plus 7 least significant bits of the test byte). The resulting timer value is 16-bit (T2). This value is used to calculate the 8-bit reload value (RC2H,L) for Timer 2 as a baud rate generator.

The correlation between the baud rate (baud) and the reload value (RC2H,L) depends on the internal MCU system frequency (fsys)

$$baud = \frac{2^{SMOD} * f_{SYS}}{64 * (2^{16}\text{-}RC2H,L)} \quad [9.1]$$

The relationship between the baud rate (baud) and the recording value of Timer 2 (T2) depends on the MCU system frequency (fsys) and the number of received bits (Nb)

$$baud = \frac{f_{SYS} * N_b}{12 * T2} \quad [9.2]$$

Combining **Equation [9.1]** and **Equation [9.2]**, with SMOD=1 and Nb=8, resolving the for RC2H,L leads to formula

$$RC2H,L = 2^{16} - T2*3/64 \quad [9.3]$$

The benefit of using the test byte 80H and recording 8 bits makes the formula easier for realization in assembly language. The division with 64 can be simply achieved by a 6-bit right shift operation. Additionally, the result of the division is rounded by a simple bit comparison of the last right shifted bit. After setting Timer 2 to baudrate generator mode,

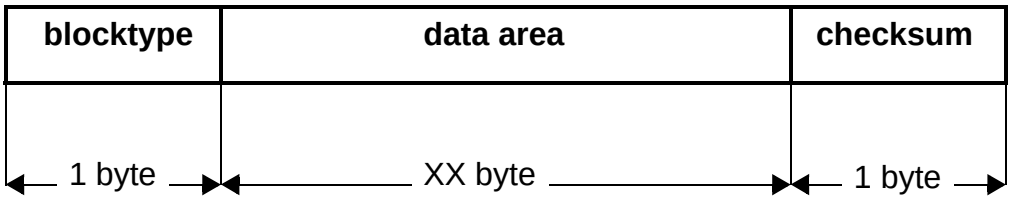
the bootstrap loader sends an Acknowledge byte (55_H) to the host. If this byte is received correctly, it will be guaranteed that both serial interfaces are working with the same baud rate.

9.2.2 Phase II: Serial communication protocol and the working modes

After the successful synchronization to the host, the bootstrap loader enters phase II, during which it communicates with the host to select the desired working modes. The detailed communication protocol is explained as follows:

9.2.2.1 Serial communication protocol

The communication between the host and the bootstrap loader is done by a simple transfer protocol. The information is sent from the host to the MCU in blocks. All the blocks follow the specified block structure. The communication is nearly unidirectional, that is, that the host is sending several transfer blocks and the bootstrap loader is just confirming them by sending back single acknowledge or error bytes. The MCU itself does not send any transfer blocks. **Figure 9-6** shows the format of the transfer block.



Format Item	Description
blocktype	This byte determines how the data in the **data area** is interpreted. Implemented block types are: 00_H type "HEADER" 01_H type "DATA" 02_H type "END OF TRANSMISSION" (EOT)
data area	This area contains a number of bytes which represents the data of the block. The maximal length of **data area** cannot exceed 125 bytes.
checksum	the XOR checksum of **block byte** and **data area**, sent after the **data area**.

Figure 9-6 Basic Structure of a Bootstrap Loader Transfer Block

The host would decide the number of transfer blocks and their respective lengths during one serial communication process. For safety purpose the last byte of each transfer block is a simple checksum of the block type and data area. The host generates the checksum by XOR-ING all the bytes of block type and data area. Every time the bootstrap loader receives a transfer block, it recalculates the checksum of the received bytes (block type and data area) and compares it with the attached checksum.

There are three types of transfer blocks depending on the value of block type. The following table provides general information on these block types. **Table 9-3** gives an overview of these block types. Details would be described in the corresponding sections later.

Table 9-3 Types of Transfer Blocks

Block Name	block type	Description
Header Block	00_H (HEADER)	This block has a fixed length of 8 bytes. Special information is contained in the data area of the block, which is used to select different working modes.
Data Block	01_H (DATA)	This block length depends on the special information given in the previous header block. This block is used in working mode 0 to transfer a portion of program codes. The program codes are in the data area of the block.
EOT Block	02_H (EOT)	This block length depends on the special information given in the previous header block. This block is the last block in data transmission in working mode 0. The last program codes to be transferred are in the data area of the block.

The header block is always the first transfer block to be sent by the host during one data communication process. It contains the working mode number and special information on the related mode (referred to as "mode data"). **Figure 9-7** shows the general structure of this header block. In mode 0, data blocks will follow and the process ends with the EOT block. In other modes, only the header mode is sent. The structure of data and EOT blocks are described in 9.2.2.2.

00_H (1 byte)	Mode (1 byte)	Mode data (5 bytes)	Checksum (1 byte)

Format Item	Description
mode	The working mode. Refer to **Table 9-2** for description
mode data	Five bytes of special information, which are necessary to activate the corresponding working mode.
checksum	The checksum of the header block

Figure 9-7 Structure of a HEADER Block

The MCU would let the host know whether a block has been successfully received by sending out a response code. If a block is received correctly, an Acknowledge code (55_H) is sent. There are two kinds of errors. If a wrong block type is detected, the bootstrap loader would send back a block type error (FF_H) to the host. This kind of error is caused by two conditions. First condition is that the MCU receives a block type other than the three implemented ones. The other is that the MCU receives the transfer blocks in wrong sequences. For example, in working mode 0, immediately after the header block is received, if another header block instead of a data block is received, the MCU would consider this case be a wrong block type error. Besides wrong block type error, the other error is checksum error. If the checksum comparison fails, the bootstrap loader rejects the transfer block by sending back a checksum error code (FE_H) to the host. In both error cases the bootstrap loader awaits the actual block from the host again. **Table 9-4** gives a summary of the response codes to be sent back to the host by the MCU after it receives each transfer block.

Table 9-4 Types of Transfer Blocks

Confirmation status	Response code to host
Successful	55_H
Blocktype error	FF_H
Checksum error	FE_H

9.2.2.2 Working modes selection

When the bootstrap loader enters phase II, it first waits for an eight-byte long header block from the host. The header block contains the information for the selection of the working modes. Depending on this information, the bootstrap loader selects and activates the desired working mode. If the MCU receives an incorrect header block, the bootstrap loader sends, instead of an Acknowledge code, a checksum or block error code to the host and awaits the header block again. In this case the host may react by re-sending the header block.

Mode 0 is used to transfer a customer program from the host to the XRAM or SRAM of the MCU via serial interface. The block structures are described in **Figure 9-8**.

The Header Block

00_H (Header)	00H (Mode 0)	Mode data				Checksum
		Start Address (High)	**Start Address (Low)**	**Block-Length**	**Not used**	

Mode data item	Description
Start Address High/Low	16-bit start address, which determines where to copy the received program codes in the XRAM or SRAM.
Block Length	the length of the following data blocks and EOT block. Note: the **Block-Length** refers to the whole length (block type, data area and checksum) of the following transfer block (data block or EOT block).
Not used	2 bytes, these bytes are not used and can be set to any value. They will be ignored in Mode 0.

The Data Block

01_H (Data)	**Program Codes** (**Block Length** -2)	Checksum

Mode data item	Description
Program Codes	The program codes have a length of (**Block-length**-2), where Block-length is provided in the header block.

The EOT Block

02_H (EOT)	**Last codelength** (1 byte)	**Program Codes**	**Not used**	Checksum

Mode data item	Description
Last codelength	this byte indicates the length of the program codes in this EOT block.
Program Codes	The last program codes to be sent to the MCU
Not used	the length is (**Block-Length**-3-last_codelength).

Figure 9-8 Transfer Blocks for Mode 0

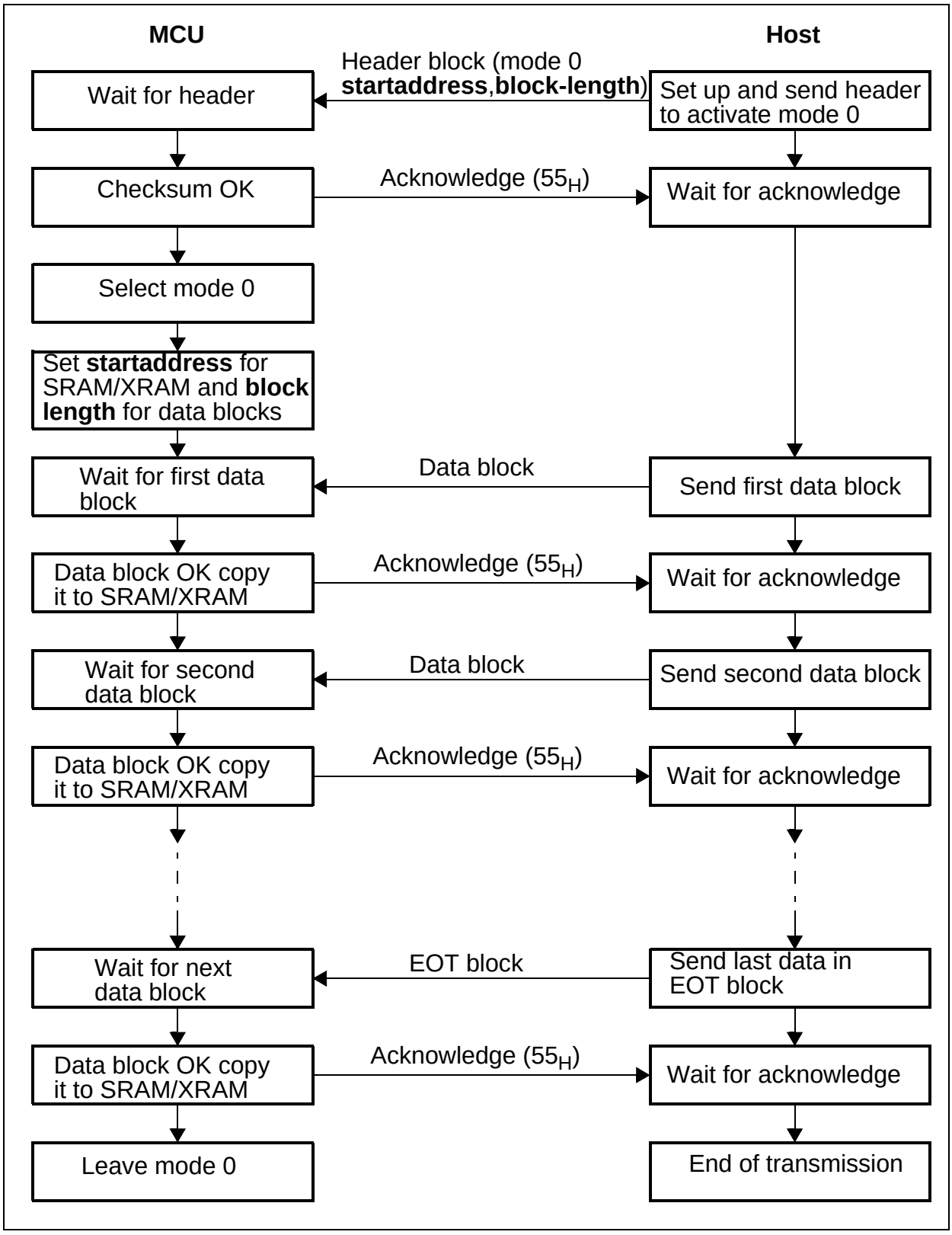


Figure 9-9 Mode 0 Communication Structure

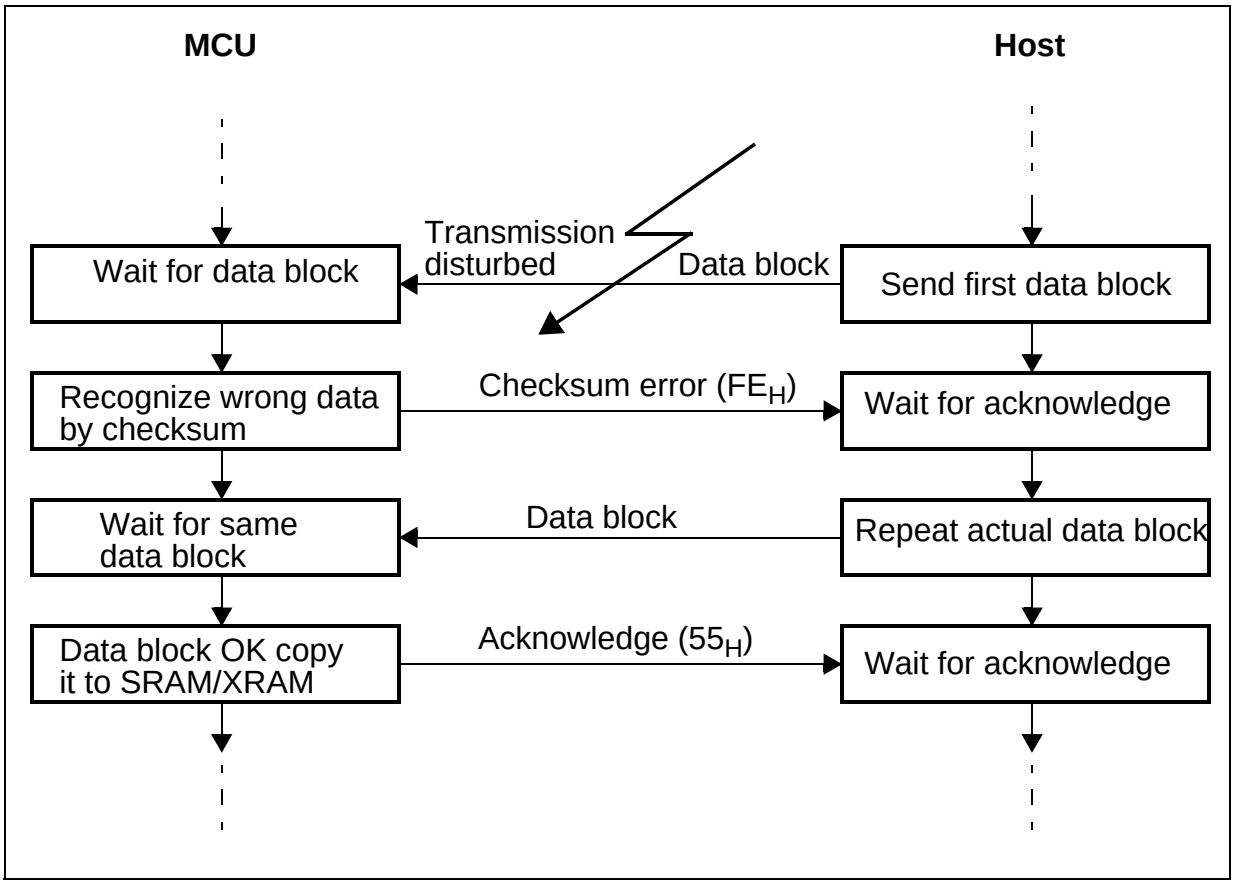


Figure 9-10 Handling Transmission Errors in Mode 0

If the host sends a header block or a block that is not implemented in the protocol (a block type number higher than 02_H), the bootstrap loader reacts in a similar way as described in **Figure 9-10** above, with the exception, that now a blocktype error (FF_H) is sent to the host. It is up to the host to handle this error properly.

The bootstrap loader flowchart of the complete transfer protocol of mode 0 is shown in **Figure 9-11**.

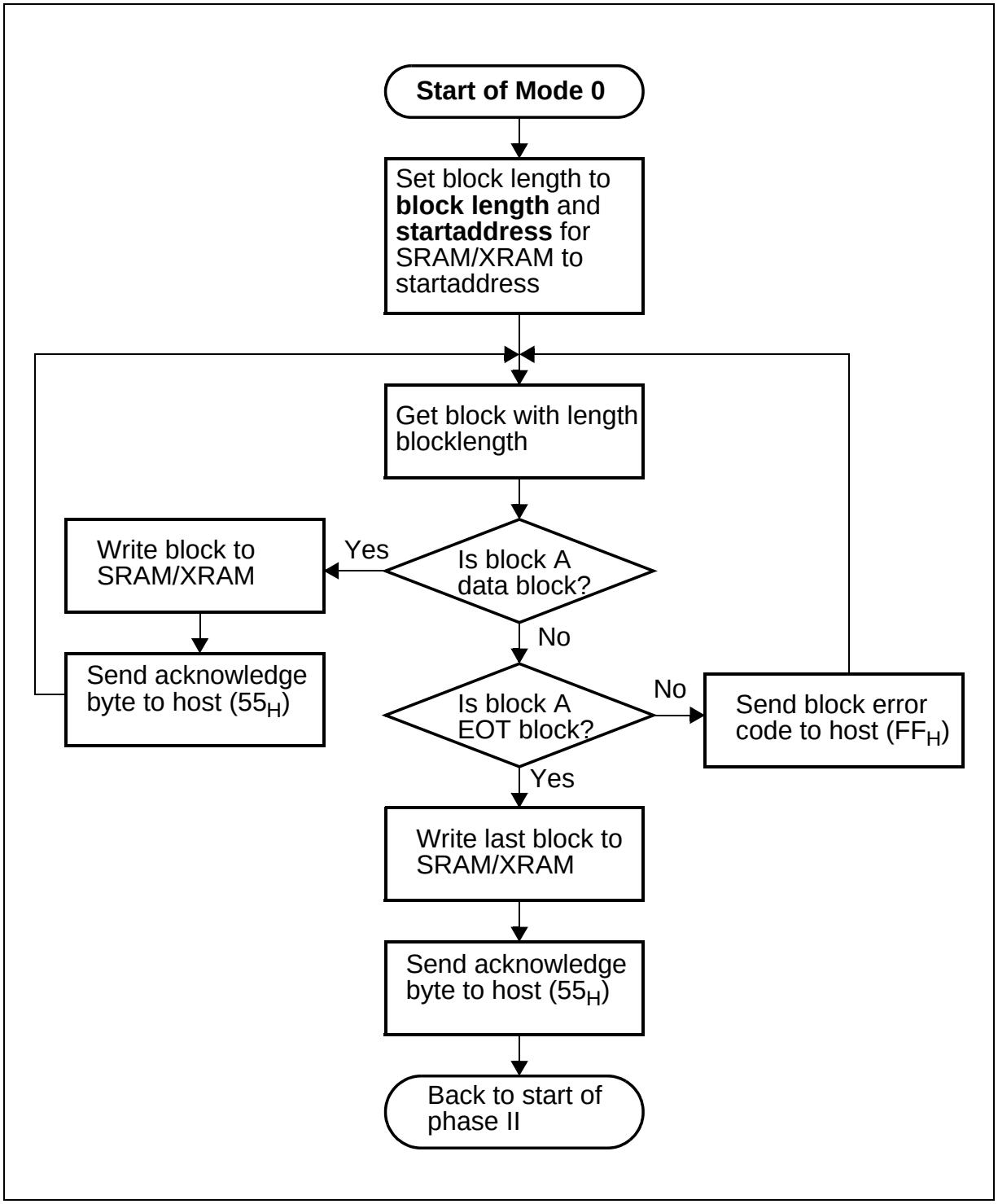


Figure 9-11 Bootstrap Loader Flowchart of Mode 0

Mode 1, 2, 3 and 4 has the same block structures.

Mode 1 is used to execute program in the XRAM of the MCU at 0FF00H. The MCU would exit the bootstrap mode and enter the XRAM mode after the communication process is completed.

Mode 2 is used to execute program in the SRAM of the MCU at 0000H. . The MCU would exit the bootstrap mode and enter the normal mode after the communication process is completed.

Mode 3 is used to transfer program from the SPI serial EEPROM to the XRAM or SRAM of the MCU. Note that if the phase A (bootstrap from serial EEPROM) is invoked after reset, the MCU would jump to the Jump Address specified by the last transfer block after the downloading process. However, in mode 3 of phase II of phase B, the MCU will return to the beginning of Phase II after downloading. Jumping to XRAM/SRAM can be invoked by other host commands. This is to allow more control by the host.

Mode 4 is used to transfer program from the I2C serial EEPROM to the XRAM or SRAM of the MCU. Note that if the phase A (bootstrap from serial EEPROM) is invoked after reset, the MCU would jump to the Jump Address specified by the last transfer block after the downloading process. However, in mode 4 of phase II of phase B, the MCU will return to the beginning of Phase II after downloading. Jumping to XRAM/SRAM can be invoked by other host commands. This is to allow more control by the host.

The block structures are described in **Figure 9-12**. In these modes, the header block is the only transfer block to be sent by the host, no further serial communication is necessary. **Figure 9-13**, **Figure 9-14** shows the communication structure and the transfer protocol for mode 1. **Figure 9-15**, **Figure 9-16** shows the communication structure and the transfer protocol for mode 2. **Figure 9-17**, **Figure 9-18** shows the communication structure and transfer protocol for mode 3 and 4.

The Header Block

00_H (Header)	Mode	Mode data	Checksum
		Not used	

Mode data item	Description
Mode	01_H: mode 1 02_H: mode 2 03_H: mode 3 04_H: mode 4
Not used	2 bytes, these bytes are not used and can be set to any value. Note:In mode 3 and 4, the MCU gets information from the header block in EEPROM instead of the host's command. So the Mode data is left empty here.

Figure 9-12 Transfer Block for Mode 1, 2, 3 and 4

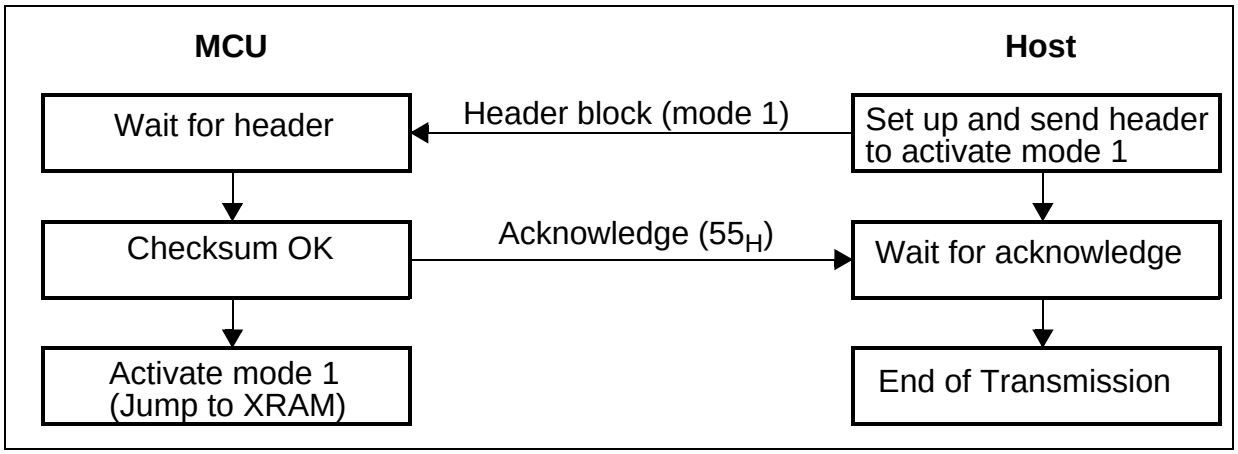


Figure 9-13 Mode 1 Communication Structure

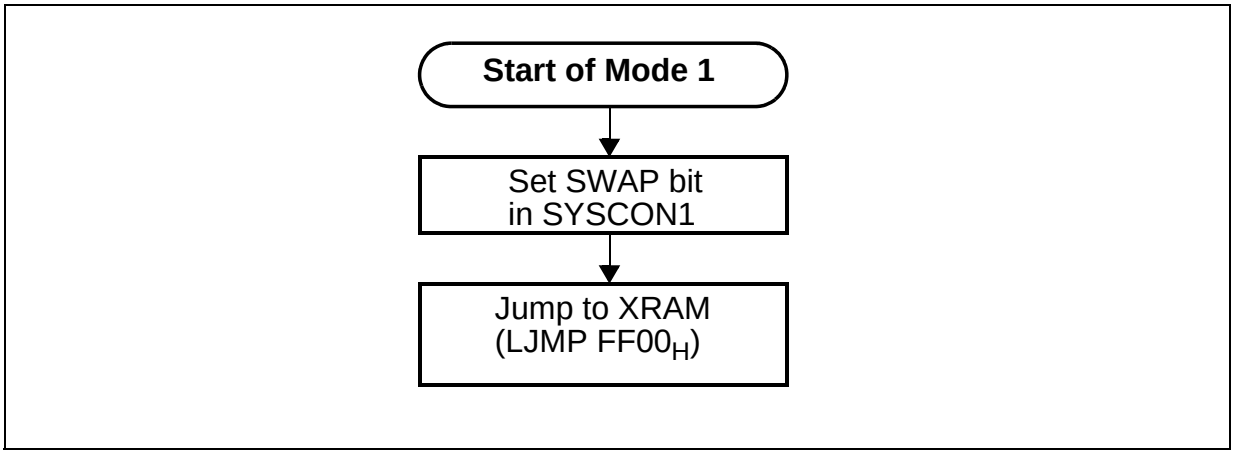


Figure 9-14 Bootstrap loader Flowchart of Mode 1

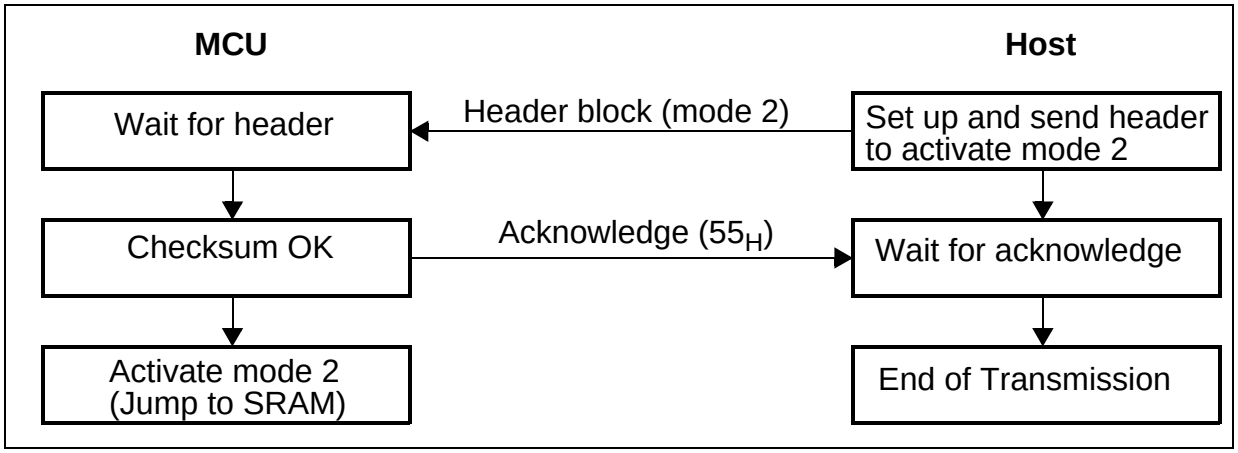


Figure 9-15 Mode 2 Communication Structure

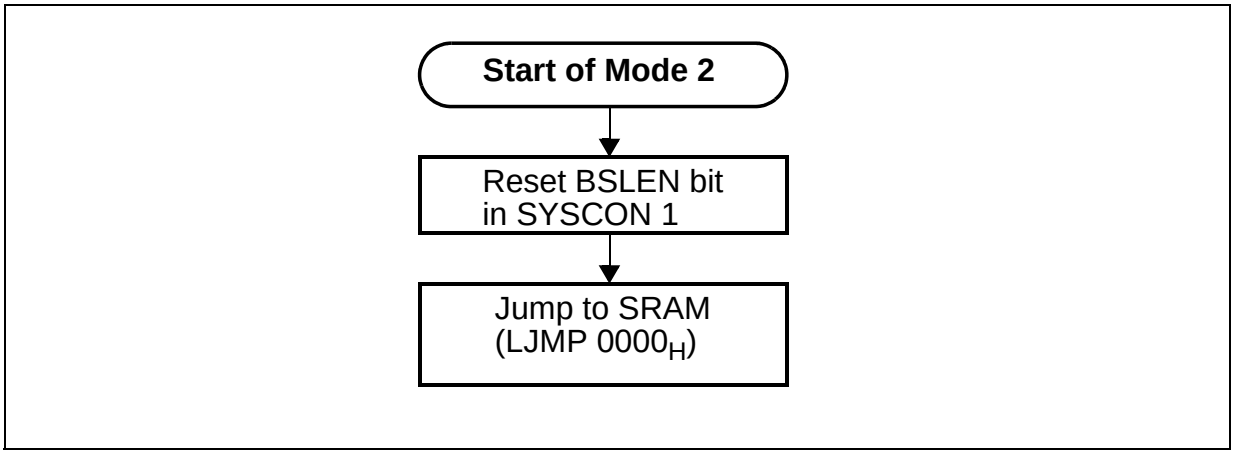


Figure 9-16 Bootstrap loader Flowchart of Mode 2

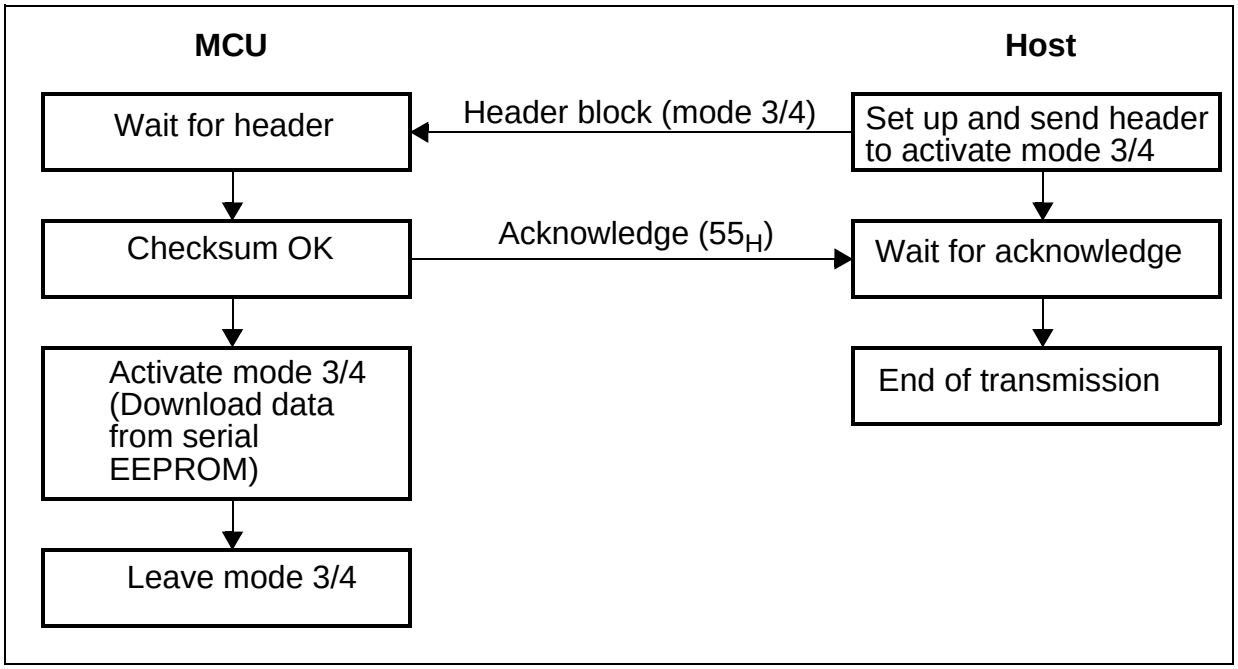


Figure 9-17 Communication Structure for Mode 3 or 4

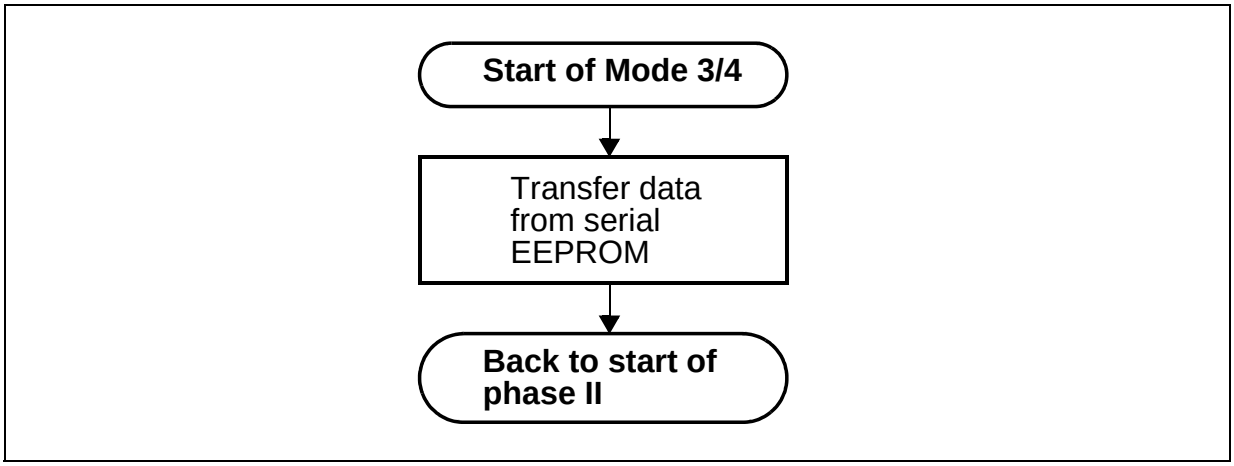


Figure 9-18 Bootstrap loader Flowchart of Mode 3 or 4

10 Index

A

B

C

D

E

F

G

H

I

K

M

O

P

R

S

T

U

V

W

X

((1))

Infineon goes for Business Excellence

“Business excellence means intelligent approaches and clearly defined processes, which are both constantly under review and ultimately lead to good operating results.
Better operating results and business excellence mean less idleness and wastefulness for all of us, more professional success, more accurate information, a better overview and, thereby, less frustration and more satisfaction.”

Dr. Ulrich Schumacher

http://www.infineon.com

Published by Infineon Technologies AG