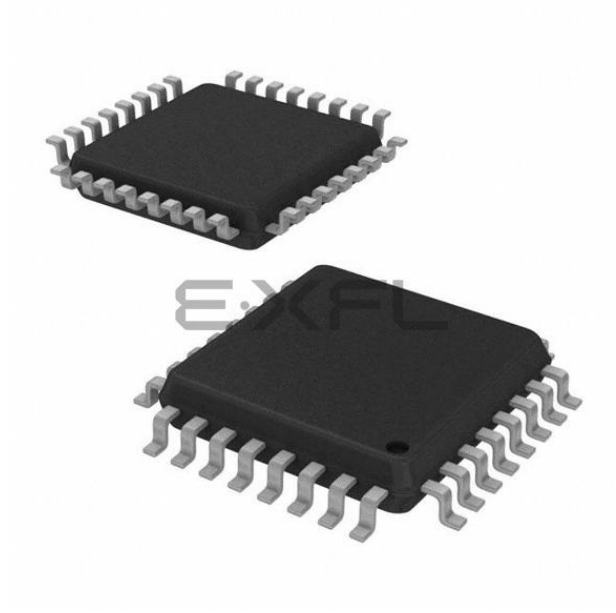


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

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

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Not For New Designs
Core Processor	8051
Core Size	8-Bit
Speed	50MHz
Connectivity	SMBus (2-Wire/I²C), SPI, UART/USART
Peripherals	POR, PWM, Temp Sensor, WDT
Number of I/O	29
Program Memory Size	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	A/D 21x10b; D/A 1x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	32-LQFP
Supplier Device Package	32-LQFP (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f368-c-gq

1.1.3. Additional Features

The C8051F36x SoC family includes several key enhancements to the CIP-51 core and peripherals to improve performance and ease of use in end applications.

The extended interrupt handler provides 16 interrupt sources into the CIP-51 (as opposed to 7 for the standard 8051), allowing numerous analog and digital peripherals to interrupt the controller. An interrupt driven system requires less intervention by the MCU, giving it more effective throughput. The extra interrupt sources are very useful when building multi-tasking, real-time systems.

Eight reset sources are available: power-on reset circuitry (POR), an on-chip V_{DD} Monitor (forces reset when power supply voltage drops below V_{RST} as given in Table 12.1 on page 134), a Watchdog Timer, a Missing Clock Detector, a voltage level detection from Comparator0, a forced software reset, an external reset pin, and an illegal Flash access protection circuit. Each reset source except for the POR, Reset Input Pin, or Flash error may be disabled by the user in software. The WDT may be permanently enabled in software after a power-on reset during MCU initialization.

The internal oscillator factory calibrated to 24.5 MHz ±2%. This internal oscillator period may be user programmed in ~0.5% increments. An additional low-frequency oscillator is also available which facilitates low-power operation. An external oscillator drive circuit is included, allowing an external crystal, ceramic resonator, capacitor, RC, or CMOS clock source to generate the system clock. If desired, the system clock source may be switched on-the-fly between both internal and external oscillator circuits. An external oscillator can also be extremely useful in low power applications, allowing the MCU to run from a slow (power saving) source, while periodically switching to the fast (up to 25 MHz) internal oscillator as needed. Additionally, an on-chip PLL is provided to achieve higher system clock speeds for increased throughput.

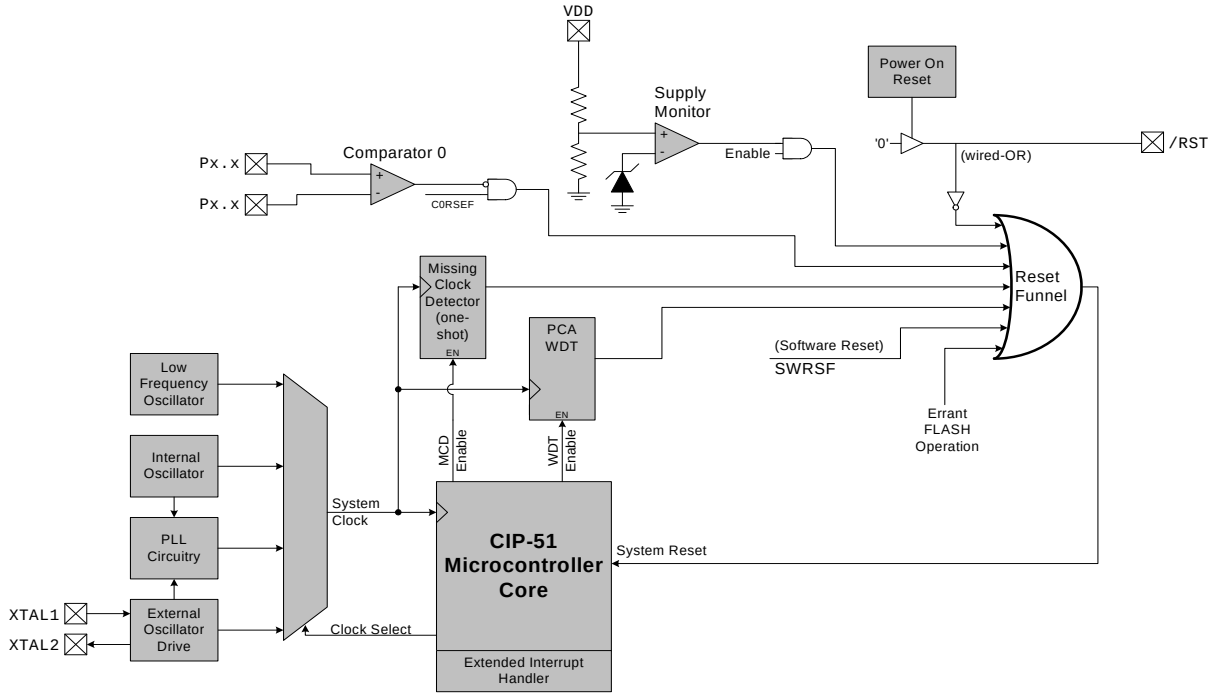


Figure 1.5. On-Chip Clock and Reset

4. Pinout and Package Definitions

Table 4.1. Pin Definitions for the C8051F36x

Name	Pin 'F360/3 (48-pin)	Pin 'F361/4/6/8 (32-pin)	Pin 'F362/5/7/9 (28-pin)	Type	Description
V_{DD}	19, 31, 43	4	4		Power Supply Voltage.
GND	18, 30, 42	3	3		Ground.
AGND	6	—	—		Analog Ground.
AV+	7	—	—		Analog Supply Voltage. Must be tied to +2.7 to +3.6 V.
$\overline{RST}$/	8	5	5	D I/O	Device Reset. Open-drain output of internal POR or V_{DD} Monitor. An external source can initiate a system reset by driving this pin low for at least 10 µs.
C2CK				D I/O	Clock signal for the C2 Debug Interface.
P4.6/	9	—	—	D I/O or A In	Port 4.6. See Section 17 for a complete description.
C2D				D I/O	Bi-directional data signal for the C2 Debug Interface.
P3.0/	—	6	6	D I/O or A In	Port 3.0. See Section 17 for a complete description.
C2D				D I/O	Bi-directional data signal for the C2 Debug Interface.
P0.0	5	2	2	D I/O or A In	Port 0.0. See Section 17 for a complete description.
P0.1	4	1	1	D I/O or A In	Port 0.1. See Section 17 for a complete description.
P0.2	3	32	28	D I/O or A In	Port 0.2. See Section 17 for a complete description.
P0.3	2	31	27	D I/O or A In	Port 0.3. See Section 17 for a complete description.
P0.4	1	30	26	D I/O or A In	Port 0.4. See Section 17 for a complete description.
P0.5	48	29	25	D I/O or A In	Port 0.5. See Section 17 for a complete description.
P0.6	47	28	24	D I/O or A In	Port 0.6. See Section 17 for a complete description.
P0.7	46	27	23	D I/O or A In	Port 0.7. See Section 17 for a complete description.

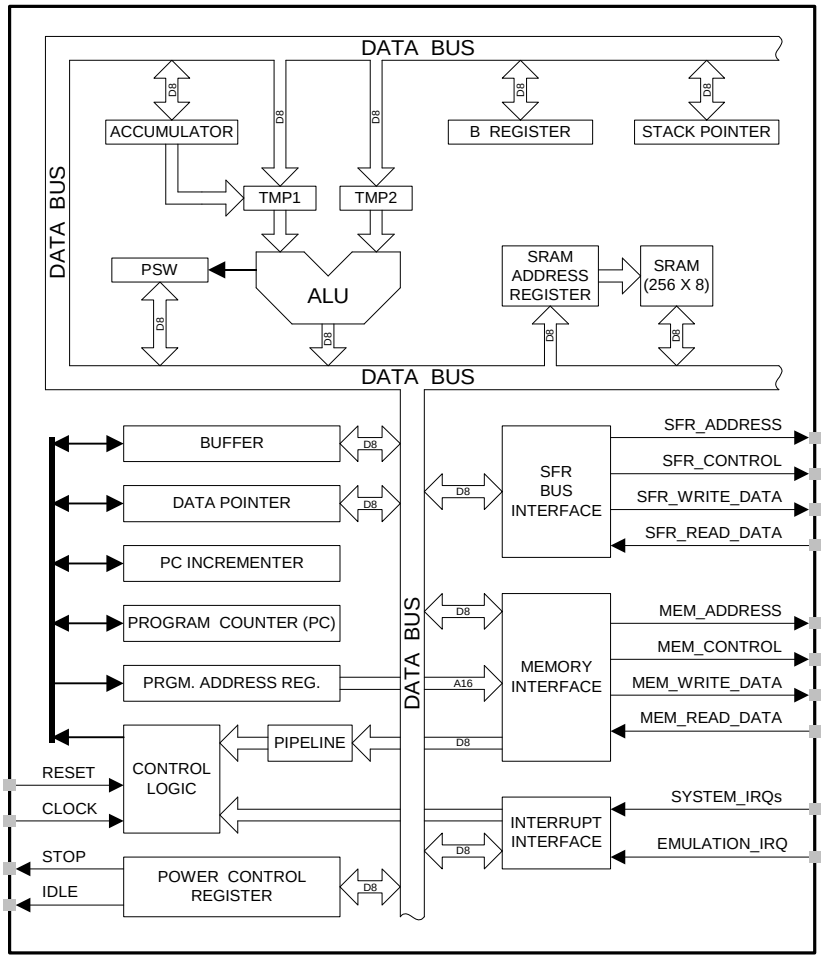


Figure 9.1. CIP-51 Block Diagram

9.2. Programming and Debugging Support

A C2-based serial interface is provided for in-system programming of the Flash program memory and communication with on-chip debug support logic. The re-programmable Flash can also be read and changed by the application software using the MOVC and MOVX instructions. This feature allows program memory to be used for non-volatile data storage as well as updating program code under software control.

The on-chip debug support logic facilitates full speed in-circuit debugging, allowing the setting of hardware breakpoints and watch points, starting, stopping and single stepping through program execution (including interrupt service routines), examination of the program's call stack, and reading/writing the contents of registers and memory. This method of on-chip debug is completely non-intrusive and non-invasive, requiring no RAM, Stack, timers, or other on-chip resources.

The CIP-51 is supported by development tools from Silicon Labs and third party vendors. Silicon Labs provides an integrated development environment (IDE) including editor, macro assembler, debugger and programmer. The IDE's debugger and programmer interface to the CIP-51 via its C2 interface to provide fast and efficient in-system device programming and debugging. Third party macro assemblers and C compilers are also available.

Table 9.2. Special Function Register (SFR) Memory Map

ADDRESS	SFR Page	0(8)	1(9)	2(A)	3(B)	4(C)	5(D)	6(E)	7(F)
F8	0 F	SPI0CN	PCA0L	PCA0H	PCA0CPL0	PCA0CPH0	PCA0CPL4	PCA0CPH4	VDM0CN
F0	0 F	B	MAC0BL P0MDIN	MAC0BH P1MDIN	P0MAT P2MDIN	P0MASK P3MDIN	PCA0CPL5	PCA0CPH5	- EMI0TC
E8	0 F	ADC0CN	PCA0CPL1	PCA0CPH1	PCA0CPL2	PCA0CPH2	PCA0CPL3	PCA0CPH3	RSTSRC
E0	0 F	ACC	P1MAT XBR0	P1MASK XBR1	- -	IT01CF	- SFR0CN	EIE1	EIE2
D8	0 F	PCA0CN	PCA0MD	PCA0CPM0	PCA0CPM1	PCA0CPM2	PCA0CPM3	PCA0CPM4	PCA0CPM5
D0	0 F	PSW	REF0CN	MAC0ACC0 CCH0LC	MAC0ACC1 CCH0MA	MAC0ACC2 P0SKIP	MAC0ACC3 P1SKIP	MAC0OVR P2SKIP	MAC0CF P3SKIP
C8	0 F	TMR2CN	- CCH0TN	TMR2RLL	TMR2RLH	TMR2L	TMR2H	- EIP1	MAC0STA EIP2
C0	0 F	SMB0CN	SMB0CF	SMB0DAT	ADC0GTL	ADC0GTH	ADC0LTL	ADC0LTH	- EMI0CF
B8	0 F	IP	IDA0CN	AMX0N	AMX0P	ADC0CF	ADC0L	ADC0H	- OSCICL
B0	0 F	P3	P2MAT PLL0MUL	P2MASK PLL0FLT	- PLL0CN	- -	P4	FLSCL OSCXCN	FLKEY OSCICN
A8	0 F	IE	- PLL0DIV	EMI0CN	- -	- FLSTAT	- OSCLCN	MAC0RNDL P4MDOUT	MAC0RNDH P3MDOUT
A0	0 F	P2	SPI0CFG	SPI0CKR	SPI0DAT	MAC0AL P0MDOUT	MAC0AH P1MDOUT	- P2MDOUT	SFRPAGE
98	0 F	SCON0	SBUF0	CPT1CN	CPT0CN	CPT1MD	CPT0MD	CPT1MX	CPT0MX
90	0 F	P1	TMR3CN	TMR3RLL	TMR3RLH	TMR3L	TMR3H	IDA0L	IDA0H
88	0 F	TCON	TMOD	TL0	TL1	TH0	TH1	CKCON	PSCTL CLKSEL
80	0 F	P0	SP	DPL	DPH	- CCH0CN	SFRNEXT	SFRLAST	PCON
		0(8)	1(9)	2(A)	3(B)	4(C)	5(D)	6(E)	7(F)

bit-addressable shaded SFRs are accessible on all SFR Pages regardless of the contents of SFRPAGE

9.5.1. Idle Mode

Setting the Idle Mode Select bit (PCON.0) causes the CIP-51 to halt the CPU and enter Idle mode as soon as the instruction that sets the bit completes. All internal registers and memory maintain their original data. All analog and digital peripherals can remain active during Idle mode.

Idle mode is terminated when an enabled interrupt or $\overline{\text{RST}}$ is asserted. The assertion of an enabled interrupt will cause the Idle Mode Selection bit (PCON.0) to be cleared and the CPU to resume operation. The pending interrupt will be serviced and the next instruction to be executed after the return from interrupt (RETI) will be the instruction immediately following the one that set the Idle Mode Select bit. If Idle mode is terminated by an internal or external reset, the CIP-51 performs a normal reset sequence and begins program execution at address 0x0000.

If enabled, the WDT will eventually cause an internal watchdog reset and thereby terminate the Idle mode. This feature protects the system from an unintended permanent shutdown in the event of an inadvertent write to the PCON register. If this behavior is not desired, the WDT may be disabled by software prior to entering the Idle mode if the WDT was initially configured to allow this operation. This provides the opportunity for additional power savings, allowing the system to remain in the Idle mode indefinitely, waiting for an external stimulus to wake up the system. Refer to Section 22.3 for more information on the use and configuration of the WDT.

Note: Any instruction which sets the IDLE bit should be immediately followed by an instruction which has two or more opcode bytes. For example:

```
// in 'C':
PCON |= 0x01;    // Set IDLE bit
PCON = PCON;     // ... Followed by a 3-cycle Dummy Instruction

; in assembly:
ORL PCON, #01h   ; Set IDLE bit
MOV PCON, PCON   ; ... Followed by a 3-cycle Dummy Instruction
```

If the instruction following the write to the IDLE bit is a single-byte instruction and an interrupt occurs during the execution of the instruction of the instruction which sets the IDLE bit, the CPU may not wake from IDLE mode when a future interrupt occurs.

9.5.2. Stop Mode

Setting the Stop Mode Select bit (PCON.1) causes the CIP-51 to enter Stop mode as soon as the instruction that sets the bit completes. In Stop mode, the CPU and oscillators are stopped, effectively shutting down all digital peripherals. Each analog peripheral must be shut down individually prior to entering Stop Mode. Stop mode can only be terminated by an internal or external reset. On reset, the CIP-51 performs the normal reset sequence and begins program execution at address 0x0000.

If enabled, the Missing Clock Detector will cause an internal reset and thereby terminate the Stop mode. The Missing Clock Detector should be disabled if the CPU is to be put to sleep for longer than the MCD timeout of 100 µs.

9.5.3. Suspend Mode

The C8051F36x devices feature a low-power SUSPEND mode, which stops the internal oscillator until an awakening event occurs. See Section “16.1.1. Internal Oscillator Suspend Mode” on page 169.

SFR Definition 9.11. PCON: Power Control

SFR Page: all pages
SFR Address: 0x87

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	STOP	IDLE	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

Bits 7–3: RESERVED. Read = 000000b. Must Write 000000b.
Bit 1: STOP: STOP Mode Select.
Writing a ‘1’ to this bit will place the CIP-51 into STOP mode. This bit will always read ‘0’.
1: CIP-51 forced into power-down mode. (Turns off oscillator).
Bit 0: IDLE: IDLE Mode Select.
Writing a ‘1’ to this bit will place the CIP-51 into IDLE mode. This bit will always read ‘0’.
1: CIP-51 forced into IDLE mode. (Shuts off clock to CPU, but clock to Timers, Interrupts, and all peripherals remain active.)

SFR Definition 10.3. EIE1: Extended Interrupt Enable 1

SFR Page: all pages
SFR Address: 0xE6

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
ET3	ECP1	ECP0	EPCA0	EADC0	EWADC0	–	ESMB0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

Bit 7: ET3: Enable Timer 3 Interrupt.
This bit sets the masking of the Timer 3 interrupt.
0: Disable Timer 3 interrupts.
1: Enable interrupt requests generated by the TF3L or TF3H flags.

Bit 6: ECP1: Enable Comparator1 (CP1) Interrupt.
This bit sets the masking of the CP1 interrupt.
0: Disable CP1 interrupts.
1: Enable interrupt requests generated by the CP1RIF or CP1FIF flags.

Bit 5: ECP0: Enable Comparator0 (CP0) Interrupt.
This bit sets the masking of the CP0 interrupt.
0: Disable CP0 interrupts.
1: Enable interrupt requests generated by the CP0RIF or CP0FIF flags.

Bit 4: EPCA0: Enable Programmable Counter Array (PCA0) Interrupt.
This bit sets the masking of the PCA0 interrupts.
0: Disable all PCA0 interrupts.
1: Enable interrupt requests generated by PCA0.

Bit 3: EADC0: Enable ADC0 Conversion Complete Interrupt.
This bit sets the masking of the ADC0 Conversion Complete interrupt.
0: Disable ADC0 Conversion Complete interrupt.
1: Enable interrupt requests generated by the AD0INT flag.

Bit 2: EWADC0: Enable ADC0 Window Comparison Interrupt.
This bit sets the masking of the ADC0 Window Comparison interrupt.
0: Disable ADC0 Window Comparison interrupt.
1: Enable interrupt requests generated by the AD0WINT flag.

Bit 1: UNUSED. Read = 0b. Write = don’t care.

Bit 0: ESMB0: Enable SMBus (SMB0) Interrupt.
This bit sets the masking of the SMB0 interrupt.
0: Disable all SMB0 interrupts.
1: Enable interrupt requests generated by SMB0.

SFR Definition 10.5. EIE2: Extended Interrupt Enable 2

SFR Page: all pages
SFR Address: 0xE7

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
–	–	–	–	–	–	EMAT	–	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

Bits 7–2: UNUSED. Read = 000000b. Write = don't care.
Bit 1: EMAT: Enable Port Match Interrupt.
This bit sets the masking of the Port Match interrupt.
0: Disable the Port Match interrupt.
1: Enable the Port Match interrupt.
Bit 0: UNUSED. Read = 0b. Write = don't care.

SFR Definition 10.6. EIP2: Extended Interrupt Priority 2

SFR Page: F
SFR Address: 0xCF

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
–	–	–	–	–	–	PMAT	–	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

Bits 7–2: UNUSED. Read = 000000b. Write = don't care.
Bit 1: PMAT: Port Match Interrupt Priority Control.
This bit sets the priority of the Port Match interrupt.
0: Port Match interrupt set to low priority level.
1: Port Match interrupt set to high priority level.
Bit 0: UNUSED. Read = 0b. Write = don't care.

Steps 3–8 must be repeated for each byte to be written

For block Flash writes, the Flash write procedure is only performed after the last byte of each block is written with the MOVX write instruction. When writing to addresses located in any of the four code banks, a Flash write block is four bytes long, from addresses ending in 00b to addresses ending in 11b. Writes must be performed sequentially (i.e. addresses ending in 00b, 01b, 10b, and 11b must be written in order). The Flash write will be performed following the MOVX write that targets the address ending in 11b. The Flash write will be performed following the MOVX write that targets the address ending in 1b. If any bytes in the block do not need to be updated in Flash, they should be written to 0xFF. The recommended procedure for writing Flash in blocks is as follows:

Step 1. Disable interrupts.
Step 2. Set CHBLKW (register CCH0CN) to select block write mode.
Step 3. Write the first key code to FLKEY: 0xA5.
Step 4. Write the second key code to FLKEY: 0xF1.
Step 5. Set PSWE (register PSCTL) to redirect MOVX commands to write to Flash.
Step 6. Clear the PSEE bit (register PSCTL).
Step 7. Using the MOVX instruction, write the first data byte to the first block location (ending in 00b).
Step 8. Clear the PSWE bit to redirect MOVX commands to the XRAM data space.
Step 9. Write the first key code to FLKEY: 0xA5.
Step 10. Write the second key code to FLKEY: 0xF1.
Step 11. Set PSWE (register PSCTL) to redirect MOVX commands to write to Flash.
Step 12. Clear the PSEE bit (register PSCTL).
Step 13. Using the MOVX instruction, write the second data byte to the second block location (ending in 01b).
Step 14. Clear the PSWE bit to redirect MOVX commands to the XRAM data space.
Step 15. Write the first key code to FLKEY: 0xA5.
Step 16. Write the second key code to FLKEY: 0xF1.
Step 17. Set PSWE (register PSCTL) to redirect MOVX commands to write to Flash.
Step 18. Clear the PSEE bit (register PSCTL).
Step 19. Using the MOVX instruction, write the third data byte to the third block location (ending in 10b).
Step 20. Clear the PSWE bit to redirect MOVX commands to the XRAM data space.
Step 21. Write the first key code to FLKEY: 0xA5.
Step 22. Write the second key code to FLKEY: 0xF1.
Step 23. Set PSWE (register PSCTL) to redirect MOVX commands to write to Flash.
Step 24. Clear the PSEE bit (register PSCTL).
Step 25. Using the MOVX instruction, write the fourth data byte to the last block location (ending in 11b).
Step 26. Clear the PSWE bit to redirect MOVX commands to the XRAM data space.
Step 27. Re-enable interrupts.

Steps 3-26 must be repeated for each block to be written.

13.1.4. Non-volatile Data Storage

The Flash memory can be used for non-volatile data storage as well as program code. This allows data such as calibration coefficients to be calculated and stored at run time. Data is written and erased using the MOVX write instruction (as described in Section 13.1.2 and Section 13.1.3) and read using the MOVC instruction. Note: MOVX read instructions always target XRAM.

16. Oscillators

The C8051F36x devices include a programmable internal high-frequency oscillator, a programmable internal low-frequency oscillator, and an external oscillator drive circuit. The internal high-frequency oscillator can be enabled, disabled, and calibrated using the OSCICN and OSCICL registers, as shown in Figure 16.1. The internal low-frequency oscillator can be enabled/disabled and calibrated using the OSCLCN register, as shown in SFR Definition 16.3. Both internal oscillators offer a selectable post-scaling feature. The system clock can be sourced by the external oscillator circuit, either internal oscillator, or the on-chip phase-locked loop (PLL). The internal oscillator's electrical specifications are given in Table 16.1 on page 170 and Table 16.2 on page 171.

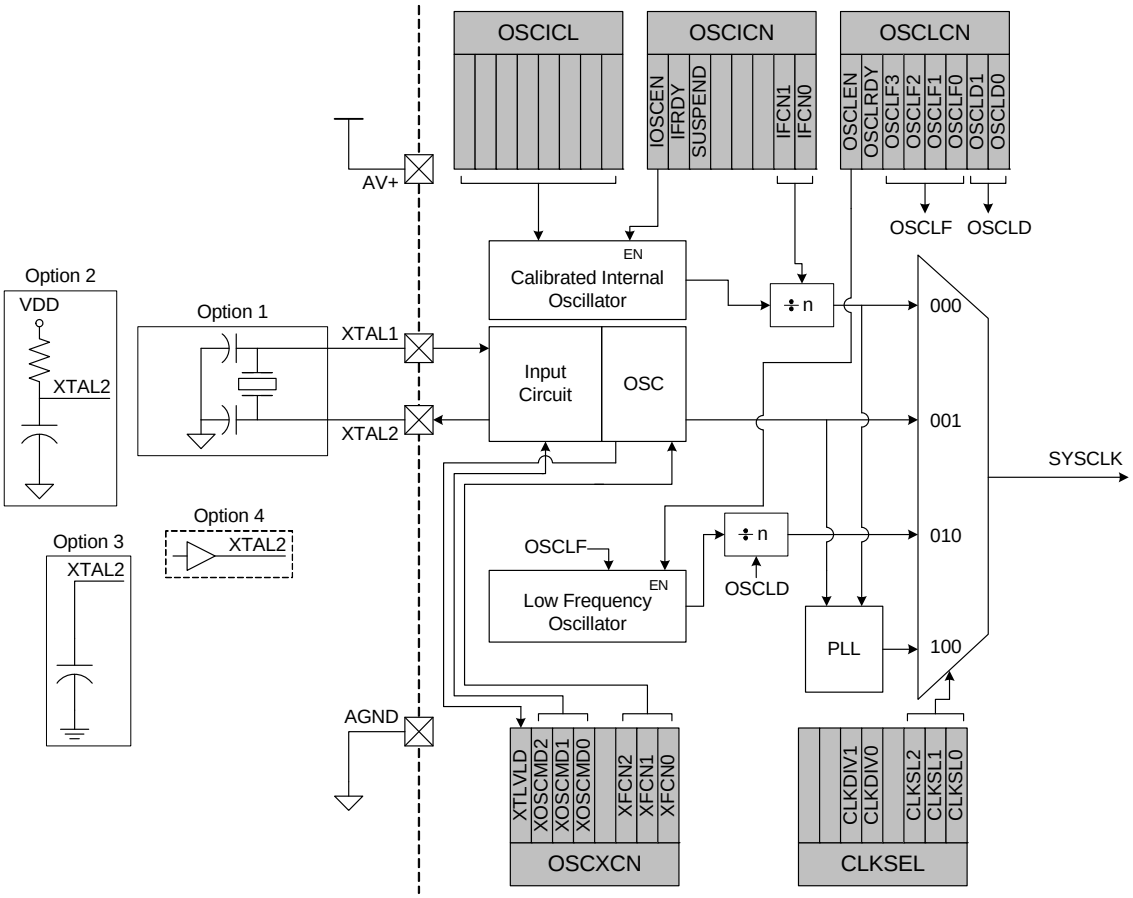


Figure 16.1. Oscillator Diagram

16.1. Programmable Internal High-Frequency (H-F) Oscillator

All devices include a calibrated internal high-frequency oscillator that defaults as the system clock after a system reset. The internal oscillator period can be adjusted via the OSCICL register as defined by SFR Definition 16.1. OSCICL is factory calibrated to obtain a 24.5 MHz frequency.

Electrical specifications for the precision internal oscillator are given in Table 16.1 on page 170 and Table 16.2 on page 171. Note that the system clock may be derived from the programmed internal oscillator divided by 1, 2, 4, or 8, as defined by the IFCN bits in register OSCICN. The divide value defaults to 8 following a reset.

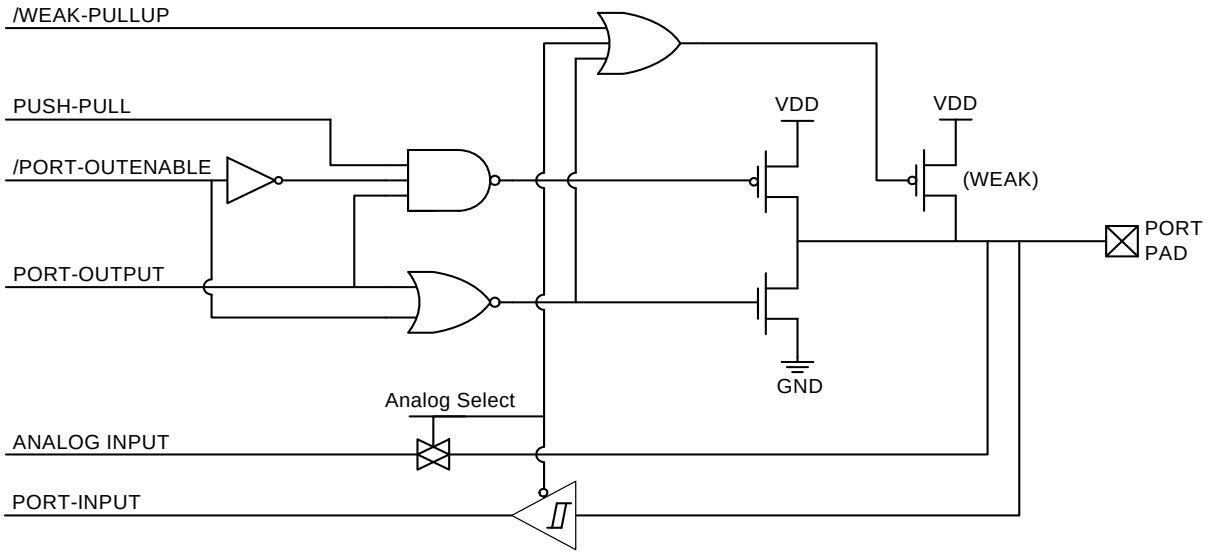


Figure 17.2. Port I/O Cell Block Diagram

SFR Definition 17.16. P2MDIN: Port2 Input Mode

SFR Page: F
SFR Address: 0xF3

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
								11111111
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

Bits 7–0: Analog Input Configuration Bits for P2.7-P2.0 (respectively).
Port pins configured as analog inputs have their weak pullup, digital driver, and digital receiver disabled.
0: Corresponding P2.n pin is configured as an analog input.
1: Corresponding P2.n pin is not configured as an analog input.

SFR Definition 17.17. P2MDOUT: Port2 Output Mode

SFR Page: F
SFR Address: 0xA6

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
								00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

Bits 7–0: Output Configuration Bits for P2.7-P2.0 (respectively): ignored if corresponding bit in register P2MDIN is logic ‘0’.
0: Corresponding P2.n Output is open-drain.
1: Corresponding P2.n Output is push-pull.

18. SMBus

The SMBus I/O interface is a two-wire, bi-directional serial bus. The SMBus is compliant with the System Management Bus Specification, version 1.1, and compatible with the I^2C serial bus. Reads and writes to the interface by the system controller are byte oriented with the SMBus interface autonomously controlling the serial transfer of the data. Data can be transferred at up to 1/10th of the system clock as a master or slave (this can be faster than allowed by the SMBus specification, depending on the system clock used). A method of extending the clock-low duration is available to accommodate devices with different speed capabilities on the same bus.

The SMBus interface may operate as a master and/or slave, and may function on a bus with multiple masters. The SMBus provides control of SDA (serial data), SCL (serial clock) generation and synchronization, arbitration logic, and START/STOP control and generation. Three SFRs are associated with the SMBus: SMB0CF configures the SMBus; SMB0CN controls the status of the SMBus; and SMB0DAT is the data register, used for both transmitting and receiving SMBus data and slave addresses.

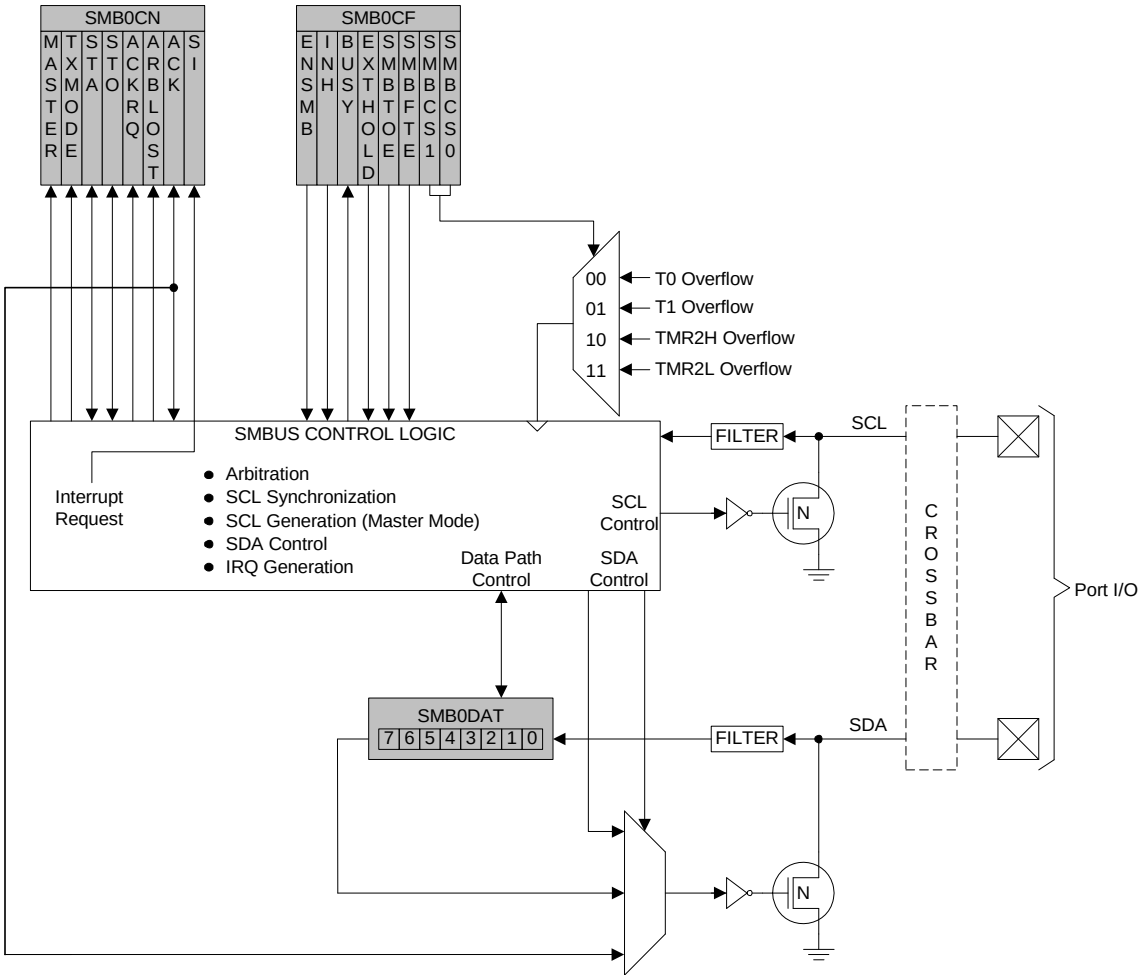


Figure 18.1. SMBus Block Diagram

SFR Definition 18.2. SMB0CN: SMBus Control

SFR Page: all pages
SFR Address: 0xC0 (bit addressable)

R	R	R/W	R/W	R	R	R/W	R/W	Reset Value
MASTER	TXMODE	STA	STO	ACKRQ	ARBLOST	ACK	SI	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

Bit 7: MASTER: SMBus Master/Slave Indicator.
This read-only bit indicates when the SMBus is operating as a master.
0: SMBus operating in Slave Mode.
1: SMBus operating in Master Mode.

Bit 6: TXMODE: SMBus Transmit Mode Indicator.
This read-only bit indicates when the SMBus is operating as a transmitter.
0: SMBus in Receiver Mode.
1: SMBus in Transmitter Mode.

Bit 5: STA: SMBus Start Flag.
Write:
0: No Start generated.
1: When operating as a master, a START condition is transmitted if the bus is free (If the bus is not free, the START is transmitted after a STOP is received or a timeout is detected). If STA is set by software as an active Master, a repeated START will be generated after the next ACK cycle.
Read:
0: No Start or repeated Start detected.
1: Start or repeated Start detected.

Bit 4: STO: SMBus Stop Flag.
Write:
0: No STOP condition is transmitted.
1: Setting STO to logic ‘1’ causes a STOP condition to be transmitted after the next ACK cycle. When the STOP condition is generated, hardware clears STO to logic ‘0’. If both STA and STO are set, a STOP condition is transmitted followed by a START condition.
Read:
0: No Stop condition detected.
1: Stop condition detected (if in Slave Mode) or pending (if in Master Mode).

Bit 3: ACKRQ: SMBus Acknowledge Request
This read-only bit is set to logic ‘1’ when the SMBus has received a byte and needs the ACK bit to be written with the correct ACK response value.

Bit 2: ARBLOST: SMBus Arbitration Lost Indicator.
This read-only bit is set to logic ‘1’ when the SMBus loses arbitration while operating as a transmitter. A lost arbitration while a slave indicates a bus error condition.

Bit 1: ACK: SMBus Acknowledge Flag.
This bit defines the out-going ACK level and records incoming ACK levels. It should be written each time a byte is received (when ACKRQ=1), or read after each byte is transmitted.
0: A "not acknowledge" has been received (if in Transmitter Mode) OR will be transmitted (if in Receiver Mode).
1: An "acknowledge" has been received (if in Transmitter Mode) OR will be transmitted (if in Receiver Mode).

Bit 0: SI: SMBus Interrupt Flag.
This bit is set by hardware under the conditions listed in Table 18.3. SI must be cleared by software. While SI is set, SCL is held low and the SMBus is stalled.

19.2.2. 9-Bit UART

9-bit UART mode uses a total of eleven bits per data byte: a start bit, 8 data bits (LSB first), a programmable ninth data bit, and a stop bit. The state of the ninth transmit data bit is determined by the value in TB80 (SCON0.3), which is assigned by user software. It can be assigned the value of the parity flag (bit P in register PSW) for error detection, or used in multiprocessor communications. On receive, the ninth data bit goes into RB80 (SCON0.2) and the stop bit is ignored.

Data transmission begins when an instruction writes a data byte to the SBUF0 register. The TI0 Transmit Interrupt Flag (SCON0.1) is set at the end of the transmission (the beginning of the stop-bit time). Data reception can begin any time after the REN0 Receive Enable bit (SCON0.4) is set to '1'. After the stop bit is received, the data byte will be loaded into the SBUF0 receive register if the following conditions are met: (1) RI0 must be logic '0', and (2) if MCE0 is logic '1', the 9th bit must be logic '1' (when MCE0 is logic '0', the state of the ninth data bit is unimportant). If these conditions are met, the eight bits of data are stored in SBUF0, the ninth bit is stored in RB80, and the RI0 flag is set to '1'. If the above conditions are not met, SBUF0 and RB80 will not be loaded and the RI0 flag will not be set to '1'. A UART0 interrupt will occur if enabled when either TI0 or RI0 is set to '1'.

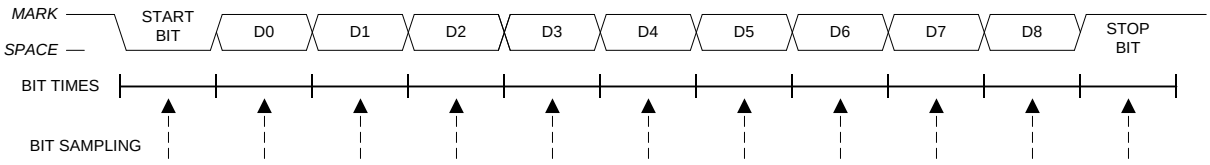


Figure 19.5. 9-Bit UART Timing Diagram

19.3. Multiprocessor Communications

9-Bit UART mode supports multiprocessor communication between a master processor and one or more slave processors by special use of the ninth data bit. When a master processor wants to transmit to one or more slaves, it first sends an address byte to select the target(s). An address byte differs from a data byte in that its ninth bit is logic ‘1’; in a data byte, the ninth bit is always set to logic ‘0’.

Setting the MCE0 bit (SCON0.5) of a slave processor configures its UART such that when a stop bit is received, the UART will generate an interrupt only if the ninth bit is logic ‘1’ (RB80 = 1) signifying an address byte has been received. In the UART interrupt handler, software will compare the received address with the slave's own assigned 8-bit address. If the addresses match, the slave will clear its MCE0 bit to enable interrupts on the reception of the following data byte(s). Slaves that weren't addressed leave their MCE0 bits set and do not generate interrupts on the reception of the following data bytes, thereby ignoring the data. Once the entire message is received, the addressed slave resets its MCE0 bit to ignore all transmissions until it receives the next address byte.

Multiple addresses can be assigned to a single slave and/or a single address can be assigned to multiple slaves, thereby enabling "broadcast" transmissions to more than one slave simultaneously. The master processor can be configured to receive all transmissions or a protocol can be implemented such that the master/slave role is temporarily reversed to enable half-duplex transmission between the original master and slave(s).

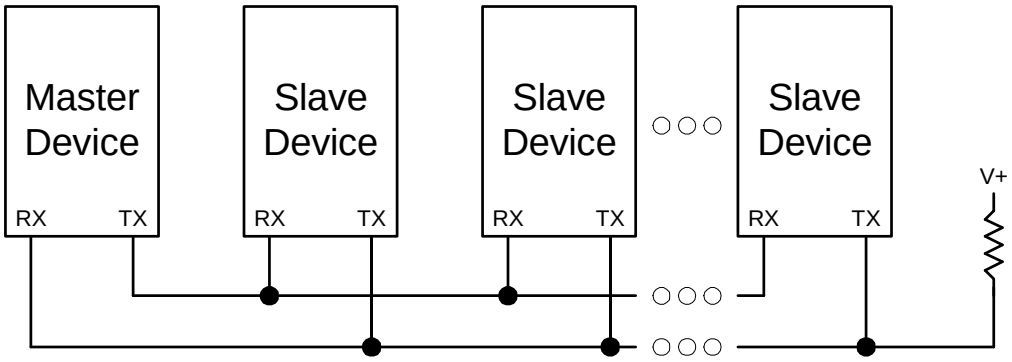


Figure 19.6. UART Multi-Processor Mode Interconnect Diagram

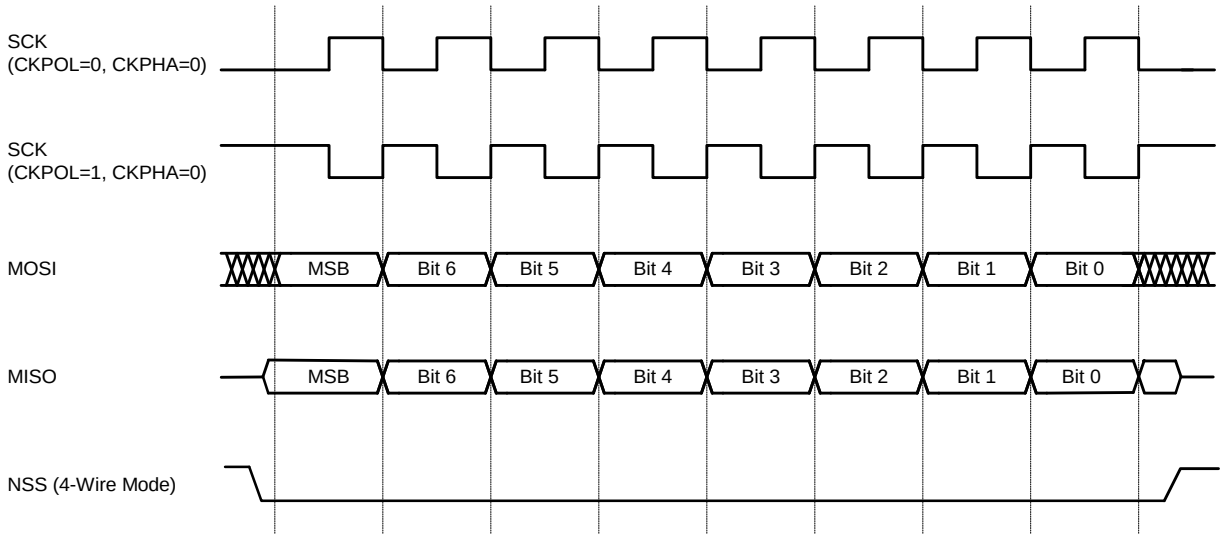


Figure 20.6. Slave Mode Data/Clock Timing (CKPHA = 0)

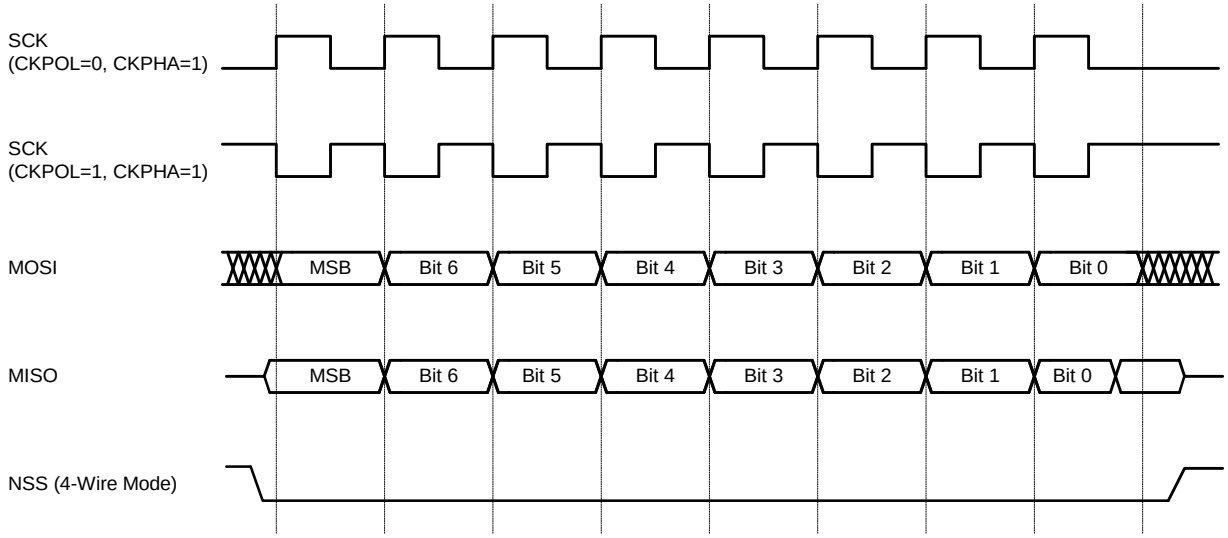
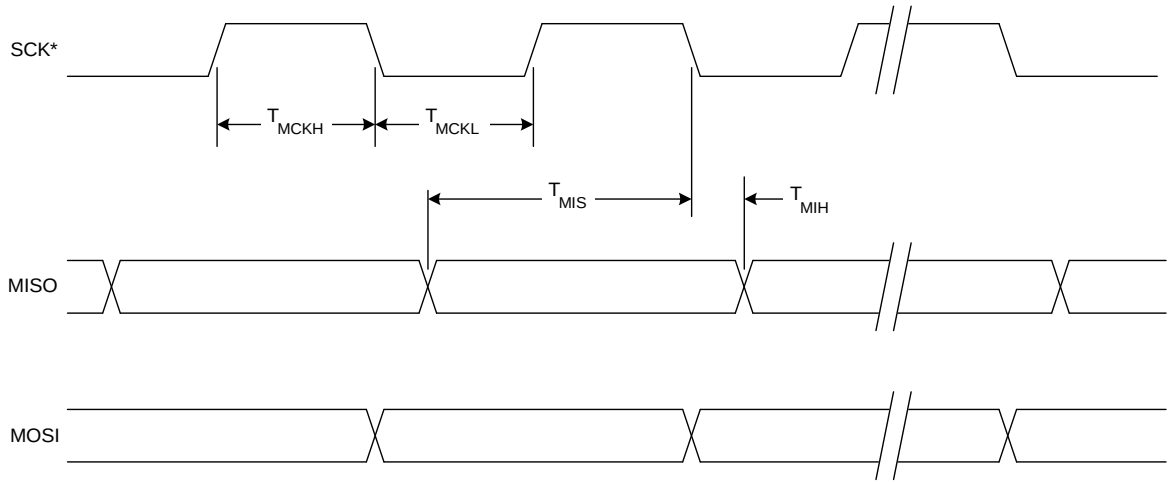
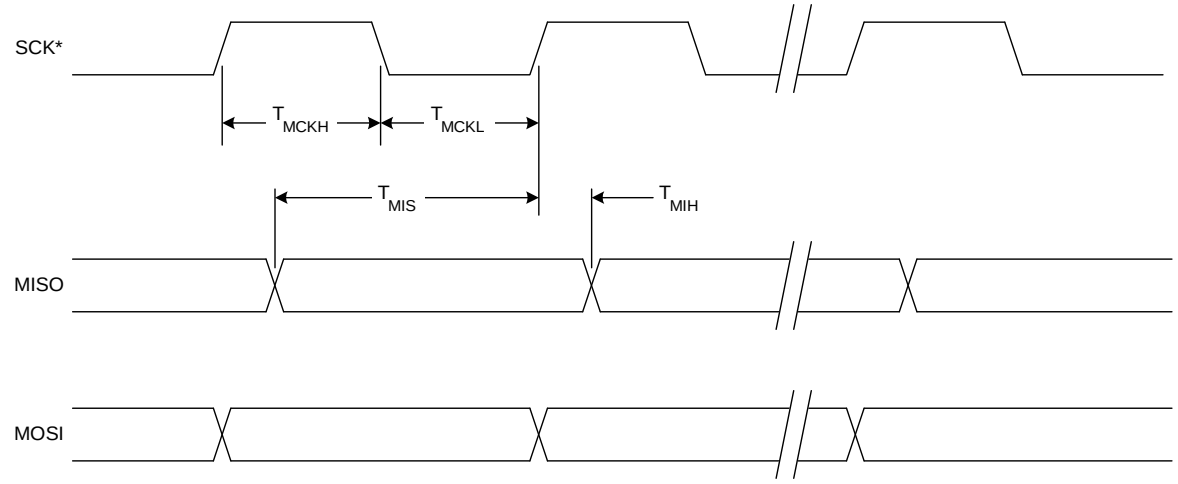


Figure 20.7. Slave Mode Data/Clock Timing (CKPHA = 1)



* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

Figure 20.8. SPI Master Timing (CKPHA = 0)



* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

Figure 20.9. SPI Master Timing (CKPHA = 1)

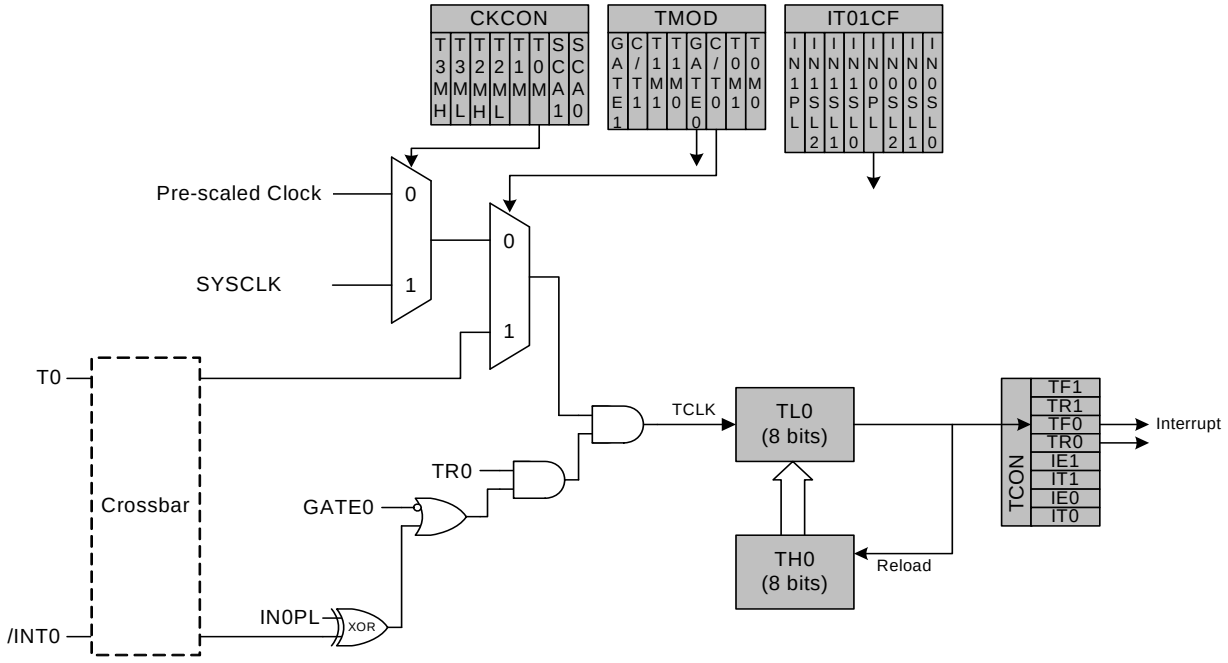


Figure 21.2. T0 Mode 2 Block Diagram

22.2.5. 8-Bit Pulse Width Modulator Mode

Each module can be used independently to generate pulse width modulated (PWM) outputs on its associated CEXn pin. The frequency of the output is dependent on the timebase for the PCA0 counter/timer. The duty cycle of the PWM output signal is varied using the module's PCA0CPLn capture/compare register. When the value in the low byte of the PCA0 counter/timer (PCA0L) is equal to the value in PCA0CPLn, the output on the CEXn pin will be high. When the count value in PCA0L overflows, the CEXn output will be low (see Figure 22.8). Also, when the counter/timer low byte (PCA0L) overflows from 0xFF to 0x00, PCA0CPLn is reloaded automatically with the value stored in the counter/timer's high byte (PCA0H) without software intervention. Setting the ECOMn and PWMn bits in the PCA0CPMn register enables 8-Bit Pulse Width Modulator mode. The duty cycle for 8-Bit PWM Mode is given by Equation 22.2.

Equation 22.2. 8-Bit PWM Duty Cycle

$$DutyCycle = \frac{(256 - PCA0CPHn)}{256}$$

Using Equation 22.2, the largest duty cycle is 100% (PCA0CPHn = 0), and the smallest duty cycle is 0.39% (PCA0CPHn = 0xFF). A 0% duty cycle may be generated by clearing the ECOMn bit to ‘0’.

Important Note About Capture/Compare Registers: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to ‘0’; writing to PCA0CPHn sets ECOMn to ‘1’.

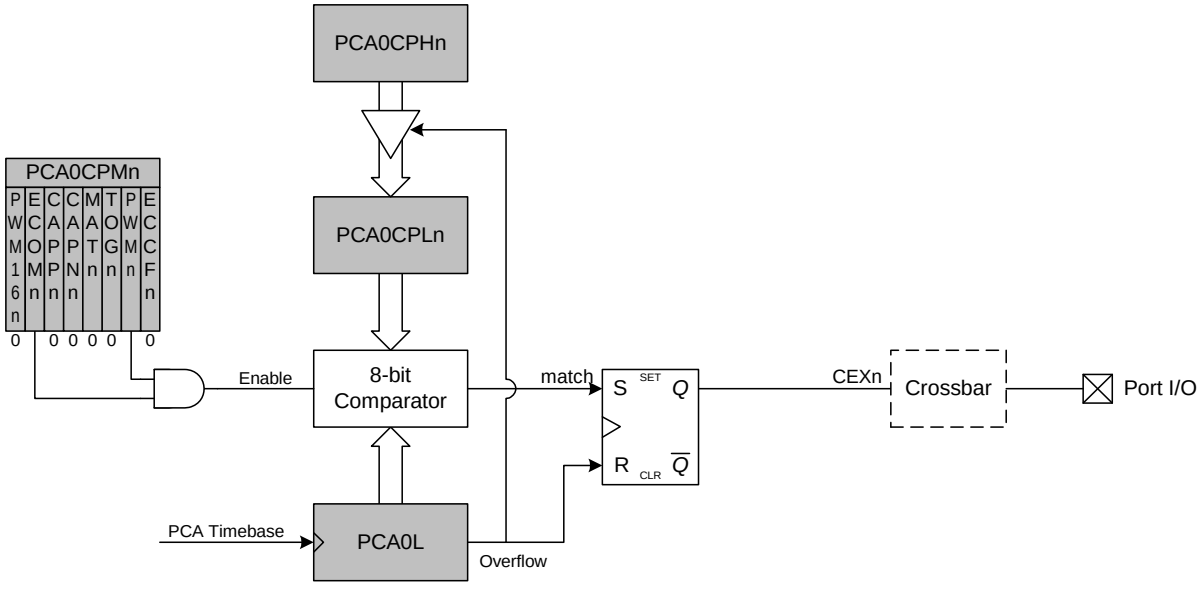


Figure 22.8. PCA 8-Bit PWM Mode Diagram