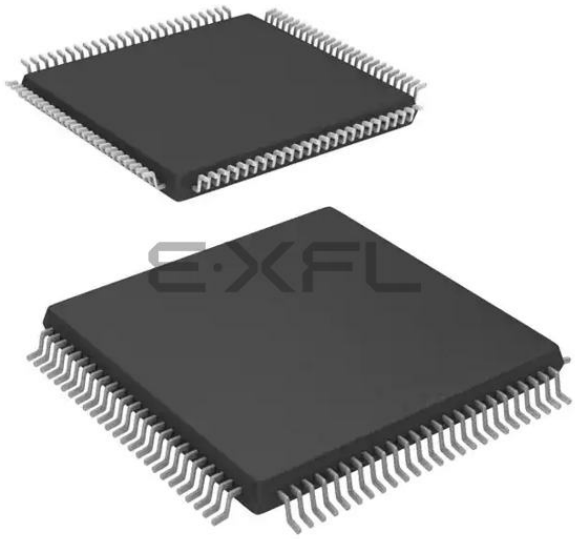

Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

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

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	8051
Core Size	8-Bit
Speed	50MHz
Connectivity	EBI/EMI, SMBus (2-Wire/I²C), SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, Temp Sensor, WDT
Number of I/O	64
Program Memory Size	128KB (128K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	8.25K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	A/D 8x8b, 8x10b; D/A 2x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	100-TQFP
Supplier Device Package	100-TQFP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f126r

1.1.3. Additional Features

Several key enhancements are implemented in the CIP-51 core and peripherals to improve overall performance and ease of use in end applications.

The extended interrupt handler provides 20 interrupt sources into the CIP-51 (as opposed to 7 for the standard 8051), allowing the 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.

There are up to seven reset sources for the MCU: an on-board V_{DD} monitor, a Watchdog Timer, a missing clock detector, a voltage level detection from Comparator0, a forced software reset, the CNVSTR0 input pin, and the $\overline{RST}$ pin. The $\overline{RST}$ pin is bi-directional, accommodating an external reset, or allowing the internally generated POR to be output on the $\overline{RST}$ pin. Each reset source except for the V_{DD} monitor and Reset Input pin may be disabled by the user in software; the V_{DD} monitor is enabled/disabled via the MONEN pin. The Watchdog Timer may be permanently enabled in software after a power-on reset during MCU initialization.

The MCU has an internal, stand alone clock generator which is used by default as the system clock after any reset. If desired, the clock source may be switched on the fly to the external oscillator, which can use a crystal, ceramic resonator, capacitor, RC, or external clock source to generate the system clock. This can be extremely useful in low power applications, allowing the MCU to run from a slow (power saving) external crystal source, while periodically switching to the 24.5 MHz internal oscillator as needed. Additionally, an on-chip PLL is provided to achieve higher system clock speeds for increased throughput.

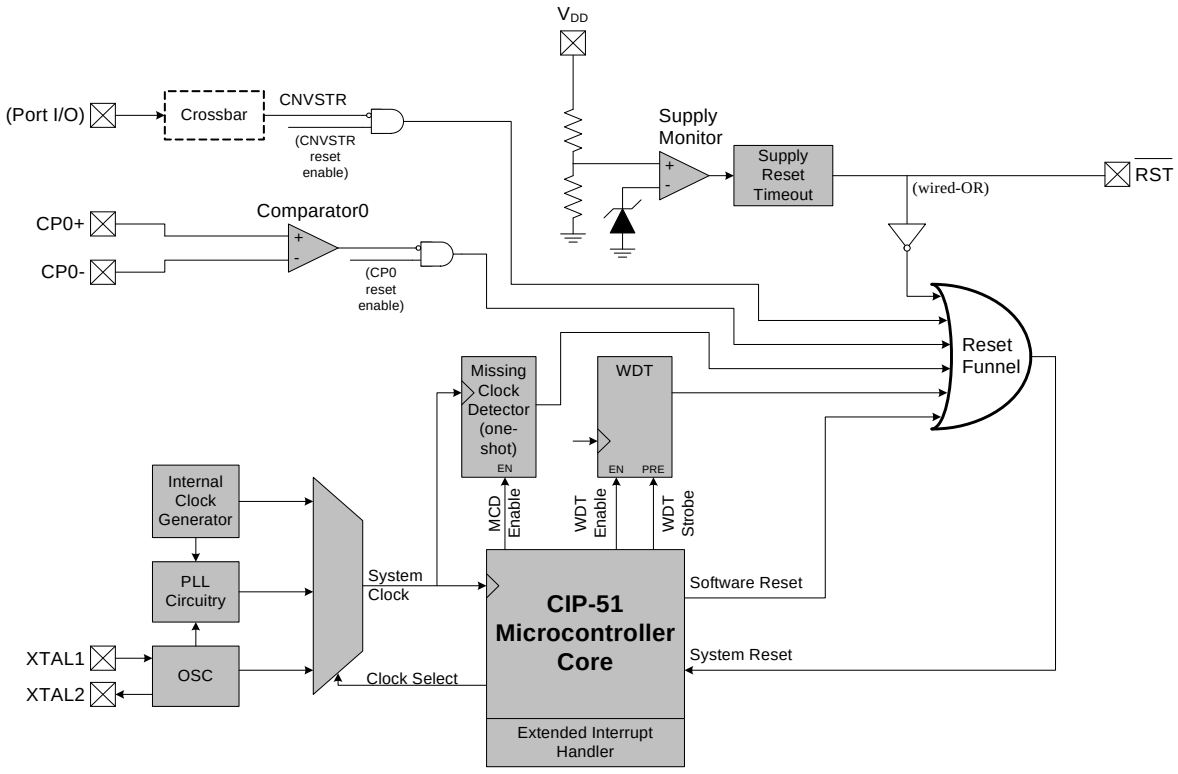

Figure 1.7. On-Board Clock and Reset

1.6. Programmable Counter Array

An on-board Programmable Counter/Timer Array (PCA) is included in addition to the five 16-bit general purpose counter/timers. The PCA consists of a dedicated 16-bit counter/timer time base with 6 programmable capture/compare modules. The timebase is clocked from one of six sources: the system clock divided by 12, the system clock divided by 4, Timer 0 overflow, an External Clock Input (ECI pin), the system clock, or the external oscillator source divided by 8.

Each capture/compare module can be configured to operate in one of six modes: Edge-Triggered Capture, Software Timer, High Speed Output, Frequency Output, 8-Bit Pulse Width Modulator, or 16-Bit Pulse Width Modulator. The PCA Capture/Compare Module I/O and External Clock Input are routed to the MCU Port I/O via the Digital Crossbar.

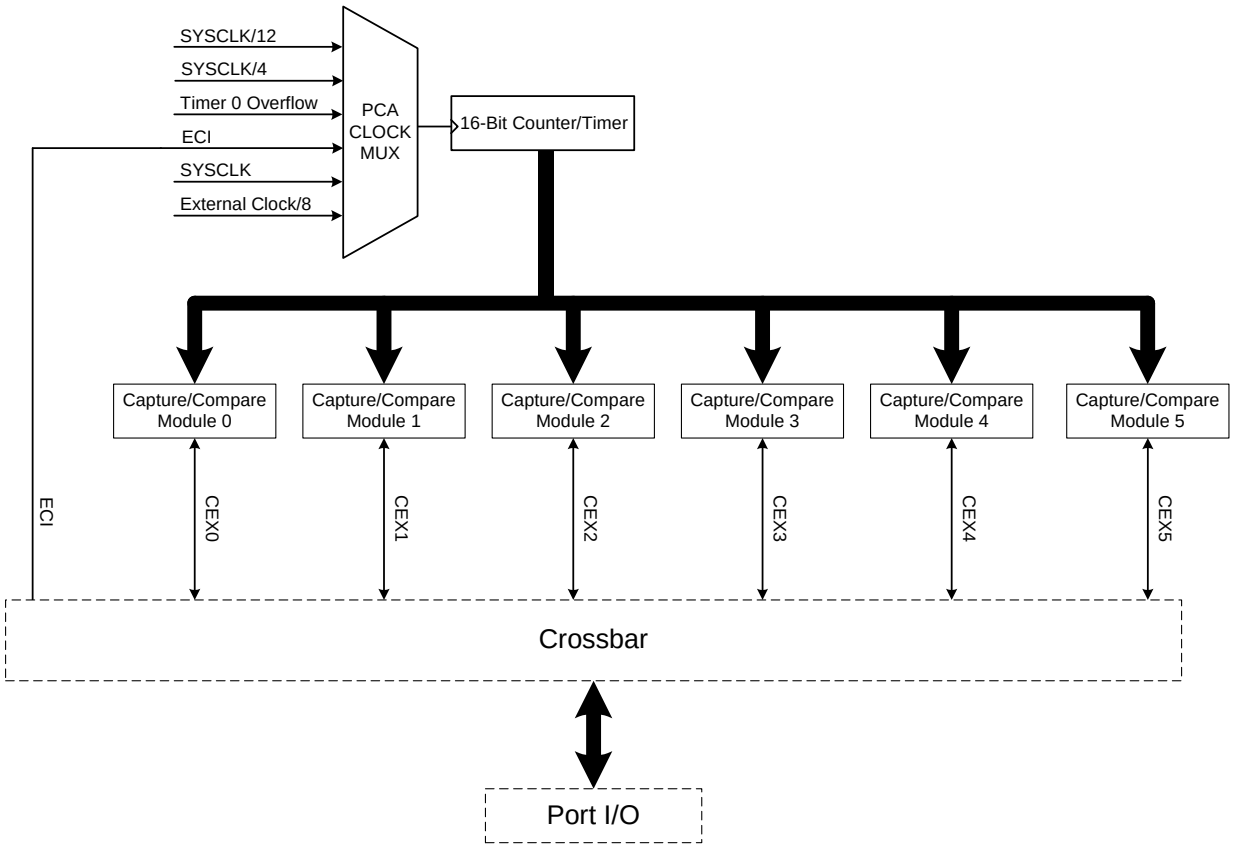

Figure 1.12. PCA Block Diagram

1.7. Serial Ports

Serial peripherals included on the devices are two Enhanced Full-Duplex UARTs, SPI Bus, and SMBus/I2C. Each of the serial buses is fully implemented in hardware and makes extensive use of the CIP-51's interrupts, thus requiring very little intervention by the CPU. The serial buses do not "share" resources such as timers, interrupts, or Port I/O, so any or all of the serial buses may be used together with any other.

1.9. 8-Bit Analog to Digital Converter

The C8051F12x devices have an on-board 8-bit SAR ADC (ADC2) with an 8-channel input multiplexer and programmable gain amplifier. This ADC features a 500 ksps maximum throughput and true 8-bit linearity with an INL of ±1LSB. Eight input pins are available for measurement. The ADC is under full control of the CIP-51 microcontroller via the Special Function Registers. The ADC2 voltage reference is selected between the analog power supply (AV+) and an external VREF pin. On the 100-pin TQFP devices, ADC2 has its own dedicated Voltage Reference input pin; on the 64-pin TQFP devices, ADC2 shares a Voltage Reference input pin with ADC0. User software may put ADC2 into shutdown mode to save power.

A programmable gain amplifier follows the analog multiplexer. The gain stage can be especially useful when different ADC input channels have widely varied input voltage signals, or when it is necessary to "zoom in" on a signal with a large DC offset (in differential mode, a DAC could be used to provide the DC offset). The PGA gain can be set in software to 0.5, 1, 2, or 4.

A flexible conversion scheduling system allows ADC2 conversions to be initiated by software commands, timer overflows, or an external input signal. ADC2 conversions may also be synchronized with ADC0 software-commanded conversions. Conversion completions are indicated by a status bit and an interrupt (if enabled), and the resulting 8-bit data word is latched into an SFR upon completion.

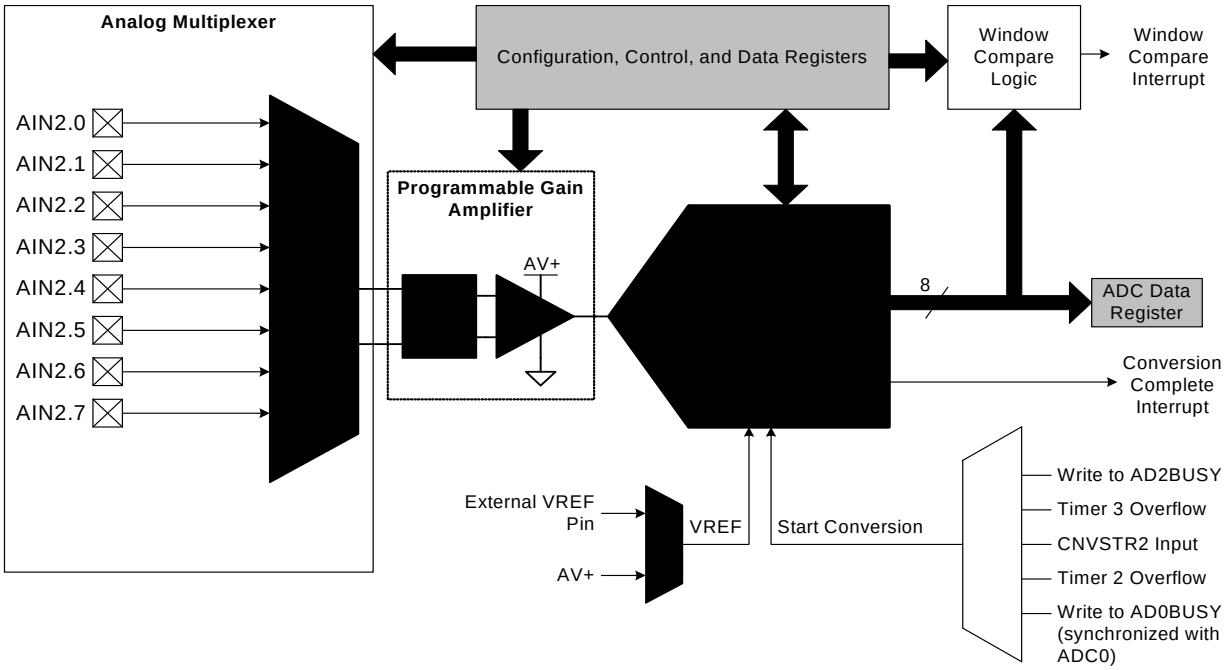

Figure 1.14. 8-Bit ADC Diagram

6.2. ADC Modes of Operation

ADC0 has a maximum conversion speed of 100 ksps. The ADC0 conversion clock is derived from the system clock divided by the value held in the ADCSC bits of register ADC0CF.

6.2.1. Starting a Conversion

A conversion can be initiated in one of four ways, depending on the programmed states of the ADC0 Start of Conversion Mode bits (AD0CM1, AD0CM0) in ADC0CN. Conversions may be initiated by:

1. Writing a ‘1’ to the AD0BUSY bit of ADC0CN;
2. A Timer 3 overflow (i.e. timed continuous conversions);
3. A rising edge detected on the external ADC convert start signal, CNVSTR0;
4. A Timer 2 overflow (i.e. timed continuous conversions).

The AD0BUSY bit is set to logic 1 during conversion and restored to logic 0 when conversion is complete. The falling edge of AD0BUSY triggers an interrupt (when enabled) and sets the AD0INT interrupt flag (ADC0CN.5). Converted data is available in the ADC0 data word MSB and LSB registers, ADC0H, ADC0L. Converted data can be either left or right justified in the ADC0H:ADC0L register pair (see example in Figure 6.5) depending on the programmed state of the AD0LJST bit in the ADC0CN register.

When initiating conversions by writing a ‘1’ to AD0BUSY, the AD0INT bit should be polled to determine when a conversion has completed (ADC0 interrupts may also be used). The recommended polling procedure is shown below.

Step 1. Write a ‘0’ to AD0INT;
Step 2. Write a ‘1’ to AD0BUSY;
Step 3. Poll AD0INT for ‘1’;
Step 4. Process ADC0 data.

When CNVSTR0 is used as a conversion start source, it must be enabled in the crossbar, and the corresponding pin must be set to open-drain, high-impedance mode (see **Section “18. Port Input/Output” on page 235** for more details on Port I/O configuration).

7.3. ADC2 Programmable Window Detector

The ADC2 Programmable Window Detector continuously compares the ADC2 output to user-programmed limits, and notifies the system when a desired condition is detected. This is especially effective in an interrupt-driven system, saving code space and CPU bandwidth while delivering faster system response times. The window detector interrupt flag (AD2WINT in register ADC2CN) can also be used in polled mode. The ADC2 Greater-Than (ADC2GT) and Less-Than (ADC2LT) registers hold the comparison values. Example comparisons for Differential and Single-ended modes are shown in Figure 7.6 and Figure 7.5, respectively. Notice that the window detector flag can be programmed to indicate when measured data is inside or outside of the user-programmed limits, depending on the contents of the ADC2LT and ADC2GT registers.

7.3.1. Window Detector In Single-Ended Mode

Figure 7.5 shows two example window comparisons for Single-ended mode, with ADC2LT = 0x20 and ADC2GT = 0x10. Notice that in Single-ended mode, the codes vary from 0 to VREF*(255/256) and are represented as 8-bit unsigned integers. In the left example, an AD2WINT interrupt will be generated if the ADC2 conversion word (ADC2) is within the range defined by ADC2GT and ADC2LT (if 0x10 < ADC2 < 0x20). In the right example, and AD2WINT interrupt will be generated if ADC2 is outside of the range defined by ADC2GT and ADC2LT (if ADC2 < 0x10 or ADC2 > 0x20).

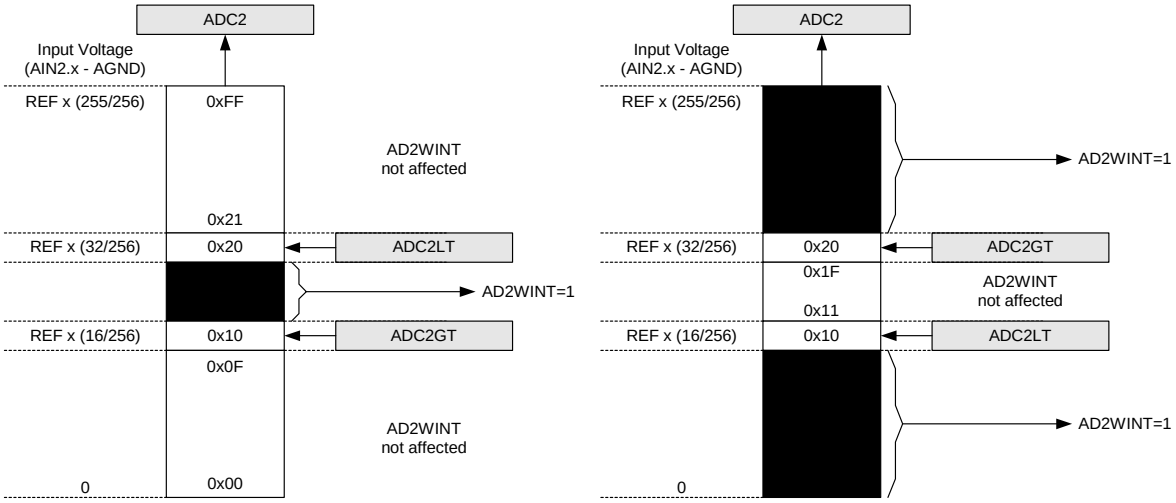

Figure 7.5. ADC2 Window Compare Examples, Single-Ended Mode

Table 9.1. Voltage Reference Electrical Characteristics

V_{DD} = 3.0 V, AV+ = 3.0 V, –40 to +85 °C unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Analog Bias Generator Power Supply Current	BIASE = 1	—	100	—	µA
Internal Reference (REFBE = 1)					
Output Voltage	25 °C ambient	2.36	2.43	2.48	V
VREF Short-Circuit Current		—	—	30	mA
VREF Temperature Coefficient		—	15	—	ppm/°C
Load Regulation	Load = 0 to 200 µA to AGND	—	0.5	—	ppm/µA
VREF Turn-on Time 1	4.7 µF tantalum, 0.1 µF ceramic bypass	—	2	—	ms
VREF Turn-on Time 2	0.1 µF ceramic bypass	—	20	—	µs
VREF Turn-on Time 3	no bypass cap	—	10	—	µs
Reference Buffer Power Supply Current		—	40	—	µA
Power Supply Rejection		—	140	—	ppm/V
External Reference (REFBE = 0)					
Input Voltage Range		1.00	—	(AV+) – 0.3	V
Input Current		—	0	1	µA

11.2.2. Data Memory

The CIP-51 implements 256 bytes of internal RAM mapped into the data memory space from 0x00 through 0xFF. The lower 128 bytes of data memory are used for general purpose registers and memory. Either direct or indirect addressing may be used to access the lower 128 bytes of data memory. Locations 0x00 through 0x1F are addressable as four banks of general purpose registers, each bank consisting of eight byte-wide registers. The next 16 bytes, locations 0x20 through 0x2F, may either be addressed as bytes or as 128 bit locations accessible with the direct addressing mode.

The upper 128 bytes of data memory are accessible only by indirect addressing. This region occupies the same address space as the Special Function Registers (SFR) but is physically separate from the SFR space. The addressing mode used by an instruction when accessing locations above 0x7F determines whether the CPU accesses the upper 128 bytes of data memory space or the SFR's. Instructions that use direct addressing will access the SFR space. Instructions using indirect addressing above 0x7F access the upper 128 bytes of data memory. Figure 11.2 illustrates the data memory organization of the CIP-51.

11.2.3. General Purpose Registers

The lower 32 bytes of data memory, locations 0x00 through 0x1F, may be addressed as four banks of general-purpose registers. Each bank consists of eight byte-wide registers designated R0 through R7. Only one of these banks may be enabled at a time. Two bits in the program status word, RS0 (PSW.3) and RS1 (PSW.4), select the active register bank (see description of the PSW in SFR Definition 11.9). This allows fast context switching when entering subroutines and interrupt service routines. Indirect addressing modes use registers R0 and R1 as index registers.

11.2.4. Bit Addressable Locations

In addition to direct access to data memory organized as bytes, the sixteen data memory locations at 0x20 through 0x2F are also accessible as 128 individually addressable bits. Each bit has a bit address from 0x00 to 0x7F. Bit 0 of the byte at 0x20 has bit address 0x00 while bit 7 of the byte at 0x20 has bit address 0x07. Bit 7 of the byte at 0x2F has bit address 0x7F. A bit access is distinguished from a full byte access by the type of instruction used (bit source or destination operands as opposed to a byte source or destination). The MCS-51™ assembly language allows an alternate notation for bit addressing of the form XX.B where XX is the byte address and B is the bit position within the byte.

For example, the instruction:

```
MOV    C, 22.3h
```

moves the Boolean value at 0x13 (bit 3 of the byte at location 0x22) into the Carry flag.

11.2.5. Stack

A programmer's stack can be located anywhere in the 256 byte data memory. The stack area is designated using the Stack Pointer (SP, address 0x81) SFR. The SP will point to the last location used. The next value pushed on the stack is placed at SP+1 and then SP is incremented. A reset initializes the stack pointer to location 0x07; therefore, the first value pushed on the stack is placed at location 0x08, which is also the first register (R0) of register bank 1. Thus, if more than one register bank is to be used, the SP should be initialized to a location in the data memory not being used for data storage. The stack depth can extend up to 256 bytes.

The MCUs also have built-in hardware for a stack record which is accessed by the debug logic. The stack record is a 32-bit shift register, where each PUSH or increment SP pushes one record bit onto the register, and each CALL pushes two record bits onto the register. (A POP or decrement SP pops one record bit,

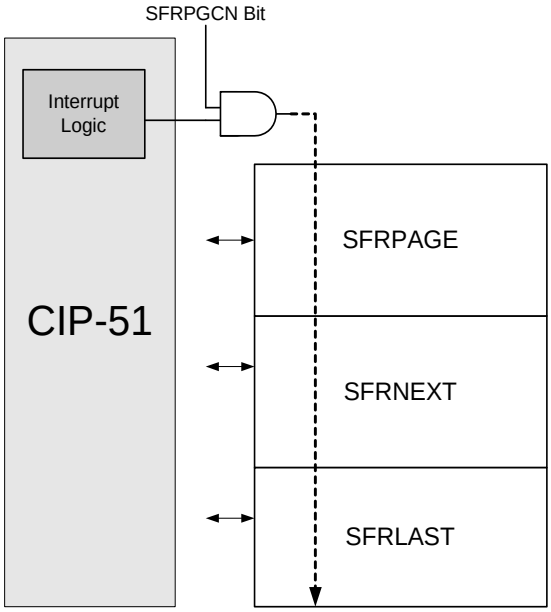

Figure 11.4. SFR Page Stack

Automatic hardware switching of the SFR Page on interrupts may be enabled or disabled as desired using the SFR Automatic Page Control Enable Bit located in the SFR Page Control Register (SFRPGCN). This function defaults to ‘enabled’ upon reset. In this way, the autoswitching function will be enabled unless disabled in software.

A summary of the SFR locations (address and SFR page) is provided in Table 11.2. in the form of an SFR memory map. Each memory location in the map has an SFR page row, denoting the page in which that SFR resides. Note that certain SFR’s are accessible from ALL SFR pages, and are denoted by the “**(ALL PAGES)**” designation. For example, the Port I/O registers P0, P1, P2, and P3 all have the “**(ALL PAGES)**” designation, indicating these SFR’s are accessible from all SFR pages regardless of the SFRPAGE register value.

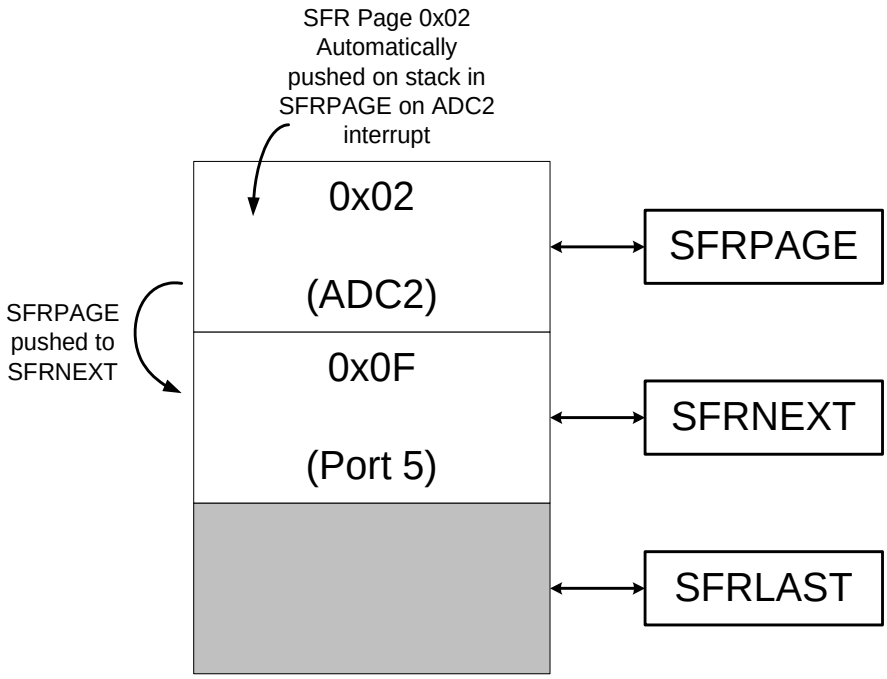

Figure 11.6. SFR Page Stack After ADC2 Window Comparator Interrupt Occurs

While in the ADC2 ISR, a PCA interrupt occurs. Recall the PCA interrupt is configured as a *high* priority interrupt, while the ADC2 interrupt is configured as a *low* priority interrupt. Thus, the CIP-51 will now vector to the high priority PCA ISR. Upon doing so, the CIP-51 will automatically place the SFR page needed to access the PCA's special function registers into the SFRPAGE register, SFR Page 0x00. The value that was in the SFRPAGE register before the PCA interrupt (SFR Page 2 for ADC2) is pushed down the stack into SFRNEXT. Likewise, the value that was in the SFRNEXT register before the PCA interrupt (in this case SFR Page 0x0F for Port 5) is pushed down to the SFRLAST register, the "bottom" of the stack. Note that a value stored in SFRLAST (via a previous software write to the SFRLAST register) will be overwritten. See Figure 11.7 below.

11.3. Interrupt Handler

The CIP-51 includes an extended interrupt system supporting a total of 20 interrupt sources with two priority levels. The allocation of interrupt sources between on-chip peripherals and external input pins varies according to the specific version of the device. Each interrupt source has one or more associated interrupt-pending flag(s) located in an SFR. When a peripheral or external source meets a valid interrupt condition, the associated interrupt-pending flag is set to logic 1.

If interrupts are enabled for the source, an interrupt request is generated when the interrupt-pending flag is set. As soon as execution of the current instruction is complete, the CPU generates an LCALL to a predetermined address to begin execution of an interrupt service routine (ISR). Each ISR must end with an RETI instruction, which returns program execution to the next instruction that would have been executed if the interrupt request had not occurred. If interrupts are not enabled, the interrupt-pending flag is ignored by the hardware and program execution continues as normal. (The interrupt-pending flag is set to logic 1 regardless of the interrupt's enable/disable state.)

Each interrupt source can be individually enabled or disabled through the use of an associated interrupt enable bit in an SFR (IE, EIE1, or EIE2). However, interrupts must first be globally enabled by setting the EA bit (IE.7) to logic 1 before the individual interrupt enables are recognized. Setting the EA bit to logic 0 disables all interrupt sources regardless of the individual interrupt-enable settings.

Note: Any instruction that clears the EA bit should be immediately followed by an instruction that has two or more opcode bytes. For example:

```
// in 'C':
EA = 0; // clear EA bit.
EA = 0; // this is a dummy instruction with two-byte opcode.

; in assembly:
CLR EA ; clear EA bit.
CLR EA ; this is a dummy instruction with two-byte opcode.
```

If an interrupt is posted during the execution phase of a "CLR EA" opcode (or any instruction which clears the EA bit), and the instruction is followed by a single-cycle instruction, the interrupt may be taken. However, a read of the EA bit will return a '0' inside the interrupt service routine. When the "CLR EA" opcode is followed by a multi-cycle instruction, the interrupt will not be taken.

Some interrupt-pending flags are automatically cleared by the hardware when the CPU vectors to the ISR. However, most are not cleared by the hardware and must be cleared by software before returning from the ISR. If an interrupt-pending flag remains set after the CPU completes the return-from-interrupt (RETI) instruction, a new interrupt request will be generated immediately and the CPU will re-enter the ISR after the completion of the next instruction.

11.3.1. MCU Interrupt Sources and Vectors

The MCUs support 20 interrupt sources. Software can simulate an interrupt event by setting any interrupt-pending flag to logic 1. If interrupts are enabled for the flag, an interrupt request will be generated and the CPU will vector to the ISR address associated with the interrupt-pending flag. MCU interrupt sources, associated vector addresses, priority order and control bits are summarized in Table 11.4. Refer to the datasheet section associated with a particular on-chip peripheral for information regarding valid interrupt conditions for the peripheral and the behavior of its interrupt-pending flag(s).

12. Multiply And Accumulate (MAC0)

The C8051F120/1/2/3 and C8051F130/1/2/3 devices include a multiply and accumulate engine which can be used to speed up many mathematical operations. MAC0 contains a 16-by-16 bit multiplier and a 40-bit adder, which can perform integer or fractional multiply-accumulate and multiply operations on signed input values in two SYSCLK cycles. A rounding engine provides a rounded 16-bit fractional result after an additional (third) SYSCLK cycle. MAC0 also contains a 1-bit arithmetic shifter that will left or right-shift the contents of the 40-bit accumulator in a single SYSCLK cycle. Figure 12.1 shows a block diagram of the MAC0 unit and its associated Special Function Registers.

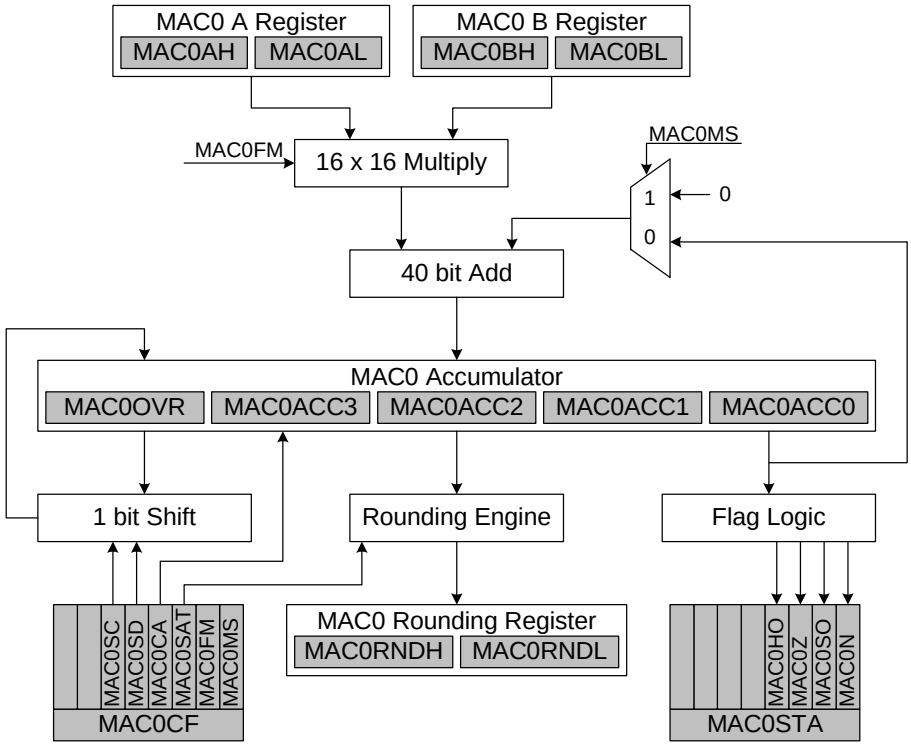

Figure 12.1. MAC0 Block Diagram

12.1. Special Function Registers

There are thirteen Special Function Register (SFR) locations associated with MAC0. Two of these registers are related to configuration and operation, while the other eleven are used to store multi-byte input and output data for MAC0. The Configuration register MAC0CF (SFR Definition 12.1) is used to configure and control MAC0. The Status register MAC0STA (SFR Definition 12.2) contains flags to indicate overflow conditions, as well as zero and negative results. The 16-bit MAC0A (MAC0AH:MAC0AL) and MAC0B (MAC0BH:MAC0BL) registers are used as inputs to the multiplier. The MAC0 Accumulator register is 40 bits long, and consists of five SFRs: MAC0OVR, MAC0ACC3, MAC0ACC2, MAC0ACC1, and MAC0ACC0. The primary results of a MAC0 operation are stored in the Accumulator registers. If they are needed, the rounded results are stored in the 16-bit Rounding Register MAC0RND (MAC0RNDH:MAC0RNDL).

13. Reset Sources

Reset circuitry allows the controller to be easily placed in a predefined default condition. On entry to this reset state, the following occur:

- CIP-51 halts program execution.
- Special Function Registers (SFRs) are initialized to their defined reset values.
- External port pins are forced to a known configuration.
- Interrupts and timers are disabled.

All SFRs are reset to the predefined values noted in the SFR detailed descriptions. The contents of internal data memory are unaffected during a reset; any previously stored data is preserved. However, since the stack pointer SFR is reset, the stack is effectively lost even though the data on the stack are not altered.

The I/O port latches are reset to 0xFF (all logic 1's), activating internal weak pullups during and after the reset. For V_{DD} Monitor resets, the $\overline{RST}$ pin is driven low until the end of the V_{DD} reset timeout.

On exit from the reset state, the program counter (PC) is reset, and the system clock defaults to the internal oscillator running at its lowest frequency. Refer to Section "**14. Oscillators**" on page **185** for information on selecting and configuring the system clock source. The Watchdog Timer is enabled using its longest timeout interval (see Section "**13.7. Watchdog Timer Reset**" on page **179**). Once the system clock source is stable, program execution begins at location 0x0000.

There are seven sources for putting the MCU into the reset state: power-on, power-fail, external $\overline{RST}$ pin, external CNVSTR0 signal, software command, Comparator0, Missing Clock Detector, and Watchdog Timer. Each reset source is described in the following sections.

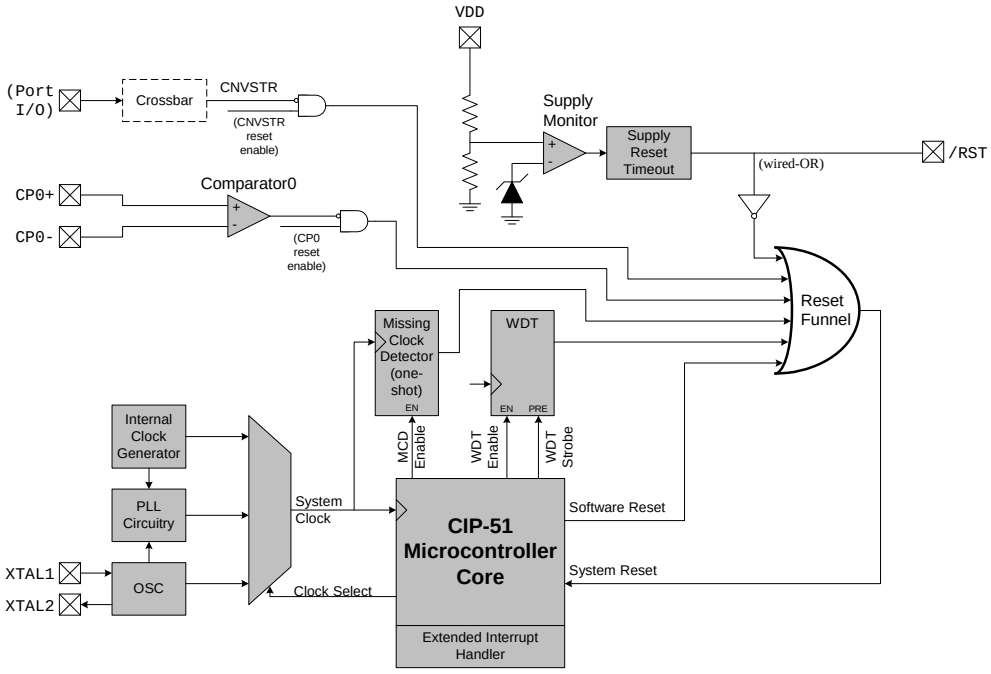

Figure 13.1. Reset Sources

13.3. External Reset

The external RST pin provides a means for external circuitry to force the MCU into a reset state. Asserting the RST pin low will cause the MCU to enter the reset state. It may be desirable to provide an external pull-up and/or decoupling of the RST pin to avoid erroneous noise-induced resets. The MCU will remain in reset until at least 12 clock cycles after the active-low RST signal is removed. The PINRSF flag (RSTSRC.0) is set on exit from an external reset.

13.4. Missing Clock Detector Reset

The Missing Clock Detector is essentially a one-shot circuit that is triggered by the MCU system clock. If the system clock goes away for more than 100 µs, the one-shot will time out and generate a reset. After a Missing Clock Detector reset, the MCDRSF flag (RSTSRC.2) will be set, signifying the MSD as the reset source; otherwise, this bit reads '0'. The state of the RST pin is unaffected by this reset. Setting the MCDRSF bit, RSTSRC.2 (see Section "**14. Oscillators**" on page **185**) enables the Missing Clock Detector.

13.5. Comparator0 Reset

Comparator0 can be configured as a reset input by writing a '1' to the C0RSEF flag (RSTSRC.5). Comparator0 should be enabled using CPT0CN.7 (see Section "**10. Comparators**" on page **119**) prior to writing to C0RSEF to prevent any turn-on chatter on the output from generating an unwanted reset. The Comparator0 reset is active-low: if the non-inverting input voltage (CP0+ pin) is less than the inverting input voltage (CP0- pin), the MCU is put into the reset state. After a Comparator0 Reset, the C0RSEF flag (RSTSRC.5) will read '1' signifying Comparator0 as the reset source; otherwise, this bit reads '0'. The state of the RST pin is unaffected by this reset.

13.6. External CNVSTR0 Pin Reset

The external CNVSTR0 signal can be configured as a reset input by writing a '1' to the CNVRSEF flag (RSTSRC.6). The CNVSTR0 signal can appear on any of the P0, P1, P2 or P3 I/O pins as described in Section "**18.1. Ports 0 through 3 and the Priority Crossbar Decoder**" on page **238**. Note that the Crossbar must be configured for the CNVSTR0 signal to be routed to the appropriate Port I/O. The Crossbar should be configured and enabled before the CNVRSEF is set. When configured as a reset, CNVSTR0 is active-low and level sensitive. CNVSTR0 cannot be used to start ADC0 conversions when it is configured as a reset source. After a CNVSTR0 reset, the CNVRSEF flag (RSTSRC.6) will read '1' signifying CNVSTR0 as the reset source; otherwise, this bit reads '0'. The state of the /RST pin is unaffected by this reset.

13.7. Watchdog Timer Reset

The MCU includes a programmable Watchdog Timer (WDT) running off the system clock. A WDT overflow will force the MCU into the reset state. To prevent the reset, the WDT must be restarted by application software before overflow. If the system experiences a software or hardware malfunction preventing the software from restarting the WDT, the WDT will overflow and cause a reset. This should prevent the system from running out of control.

Following a reset the WDT is automatically enabled and running with the default maximum time interval. If desired the WDT can be disabled by system software or locked on to prevent accidental disabling. Once locked, the WDT cannot be disabled until the next system reset. The state of the RST pin is unaffected by this reset.

The WDT consists of a 21-bit timer running from the programmed system clock. The timer measures the period between specific writes to its control register. If this period exceeds the programmed limit, a WDT

19.4. SMBus Special Function Registers

The SMBus0 serial interface is accessed and controlled through five SFR's: SMB0CN Control Register, SMB0CR Clock Rate Register, SMB0ADR Address Register, SMB0DAT Data Register and SMB0STA Status Register. The five special function registers related to the operation of the SMBus0 interface are described in the following sections.

19.4.1. Control Register

The SMBus0 Control register SMB0CN is used to configure and control the SMBus0 interface. All of the bits in the register can be read or written by software. Two of the control bits are also affected by the SMBus0 hardware. The Serial Interrupt flag (SI, SMB0CN.3) is set to logic 1 by the hardware when a valid serial interrupt condition occurs. It can only be cleared by software. The Stop flag (STO, SMB0CN.4) is set to logic 1 by software. It is cleared to logic 0 by hardware when a STOP condition is detected on the bus.

Setting the ENSMB flag to logic 1 enables the SMBus0 interface. Clearing the ENSMB flag to logic 0 disables the SMBus0 interface and removes it from the bus. Momentarily clearing the ENSMB flag and then resetting it to logic 1 will reset SMBus0 communication. However, ENSMB should not be used to temporarily remove a device from the bus since the bus state information will be lost. Instead, the Assert Acknowledge (AA) flag should be used to temporarily remove the device from the bus (see description of AA flag below).

Setting the Start flag (STA, SMB0CN.5) to logic 1 will put SMBus0 in a master mode. If the bus is free, SMBus0 will generate a START condition. If the bus is not free, SMBus0 waits for a STOP condition to free the bus and then generates a START condition after a 5 µs delay per the SMB0CR value (In accordance with the SMBus protocol, the SMBus0 interface also considers the bus free if the bus is idle for 50 µs and no STOP condition was recognized). If STA is set to logic 1 while SMBus0 is in master mode and one or more bytes have been transferred, a repeated START condition will be generated.

When the Stop flag (STO, SMB0CN.4) is set to logic 1 while the SMBus0 interface is in master mode, the interface generates a STOP condition. In a slave mode, the STO flag may be used to recover from an error condition. In this case, a STOP condition is not generated on the bus, but the SMBus hardware behaves as if a STOP condition has been received and enters the "not addressed" slave receiver mode. Note that this simulated STOP will not cause the bus to appear free to SMBus0. The bus will remain occupied until a STOP appears on the bus or a Bus Free Timeout occurs. Hardware automatically clears the STO flag to logic 0 when a STOP condition is detected on the bus.

The Serial Interrupt flag (SI, SMB0CN.3) is set to logic 1 by hardware when the SMBus0 interface enters one of 27 possible states. If interrupts are enabled for the SMBus0 interface, an interrupt request is generated when the SI flag is set. The SI flag must be cleared by software.

Important Note: If SI is set to logic 1 while the SCL line is low, the clock-low period of the serial clock will be stretched and the serial transfer is suspended until SI is cleared to logic 0. A high level on SCL is not affected by the setting of the SI flag.

The Assert Acknowledge flag (AA, SMB0CN.2) is used to set the level of the SDA line during the acknowledge clock cycle on the SCL line. Setting the AA flag to logic 1 will cause an ACK (low level on SDA) to be sent during the acknowledge cycle if the device has been addressed. Setting the AA flag to logic 0 will cause a NACK (high level on SDA) to be sent during acknowledge cycle. After the transmission of a byte in slave mode, the slave can be temporarily removed from the bus by clearing the AA flag. The slave's own address and general call address will be ignored. To resume operation on the bus, the AA flag must be reset to logic 1 to allow the slave's address to be recognized.

21.2. Multiprocessor Communications

Modes 2 and 3 support multiprocessor communication between a master processor and one or more slave processors by special use of the ninth data bit and the built-in UART0 address recognition hardware. 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. UART0 will recognize as “valid” (i.e., capable of causing an interrupt) **two** types of addresses: (1) a *masked* address and (2) a *broadcast* address **at any given time**. Both are described below.

21.2.1. Configuration of a Masked Address

The UART0 address is configured via two SFR’s: SADDR0 (Serial Address) and SADEN0 (Serial Address Enable). SADEN0 sets the bit mask for the address held in SADDR0: bits set to logic 1 in SADEN0 correspond to bits in SADDR0 that are checked against the received address byte; bits set to logic 0 in SADEN0 correspond to “don’t care” bits in SADDR0.

Example 1, SLAVE #1	
SADDR0	= 00110101
SADEN0	= 00001111
UART0 Address	= xxxx0101

Example 2, SLAVE #2	
SADDR0	= 00110101
SADEN0	= 11110011
UART0 Address	= 0011xx01

Example 3, SLAVE #3	
SADDR0	= 00110101
SADEN0	= 11000000
UART0 Address	= 00xxxxxx

Setting the SM20 bit (SCON0.5) configures UART0 such that when a stop bit is received, UART0 will generate an interrupt only if the ninth bit is logic 1 (RB80 = ‘1’) and the received data byte matches the UART0 slave address. Following the received address interrupt, the slave will clear its SM20 bit to enable interrupts on the reception of the following data byte(s). Once the entire message is received, the addressed slave resets its SM20 bit to ignore all transmissions until it receives the next address byte. While SM20 is logic 1, UART0 ignores all bytes that do not match the UART0 address and include a ninth bit that is logic 1.

21.2.2. Broadcast Addressing

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 broadcast address is the logical OR of registers SADDR0 and SADEN0, and ‘0’s of the result are treated as “don’t cares”. Typically a broadcast address of 0xFF (hexadecimal) is acknowledged by all slaves, assuming “don’t care” bits as ‘1’s. 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)..

Example 4, SLAVE #1	
SADDR0	= 00110101
SADEN0	= 00001111
Broadcast Address	= 00111111

Example 5, SLAVE #2	
SADDR0	= 00110101
SADEN0	= 11110011
Broadcast Address	= 11110111

Example 6, SLAVE #3	
SADDR0	= 00110101
SADEN0	= 11000000
Broadcast Address	= 11110101

Where all ZEROES in the Broadcast address are don’t cares.

Note in the above examples 4, 5, and 6, each slave would recognize as “valid” an address of 0xFF as a broadcast address. Also note that examples 4, 5, and 6 uses the same SADDR0 and SADEN0 register values as shown in the examples 1, 2, and 3 respectively (slaves #1, 2, and 3). Thus, a master could address each slave device individually using a masked address, and also broadcast to all three slave devices. For example, if a Master were to send an address “11110101”, only slave #1 would recognize the address as valid. If a master were to then send an address of “11111111”, all three slave devices would recognize the address as a valid broadcast address.

The C/T0 bit (TMOD.2) selects the counter/timer's clock source. When C/T0 is set to logic 1, high-to-low transitions at the selected Timer 0 input pin (T0) increment the timer register (Refer to **Section "18.1. Ports 0 through 3 and the Priority Crossbar Decoder" on page 238** for information on selecting and configuring external I/O pins). Clearing C/T selects the clock defined by the T0M bit (CKCON.3). When T0M is set, Timer 0 is clocked by the system clock. When T0M is cleared, Timer 0 is clocked by the source selected by the Clock Scale bits in CKCON (see SFR Definition 23.3).

Setting the TR0 bit (TCON.4) enables the timer when either GATE0 (TMOD.3) is logic 0 or the input signal /INT0 is logic-level 1. Setting GATE0 to '1' allows the timer to be controlled by the external input signal /INT0 (see **Section "11.3.5. Interrupt Register Descriptions" on page 157**), facilitating pulse width measurements.

TR0	GATE0	/INT0	Counter/Timer
0	X	X	Disabled
1	0	X	Enabled
1	1	0	Disabled
1	1	1	Enabled

X = Don't Care

Setting TR0 does not force the timer to reset. The timer registers should be loaded with the desired initial value before the timer is enabled.

TL1 and TH1 form the 13-bit register for Timer 1 in the same manner as described above for TL0 and TH0. Timer 1 is configured and controlled using the relevant TCON and TMOD bits just as with Timer 0. The input signal /INT1 is used with Timer 1.

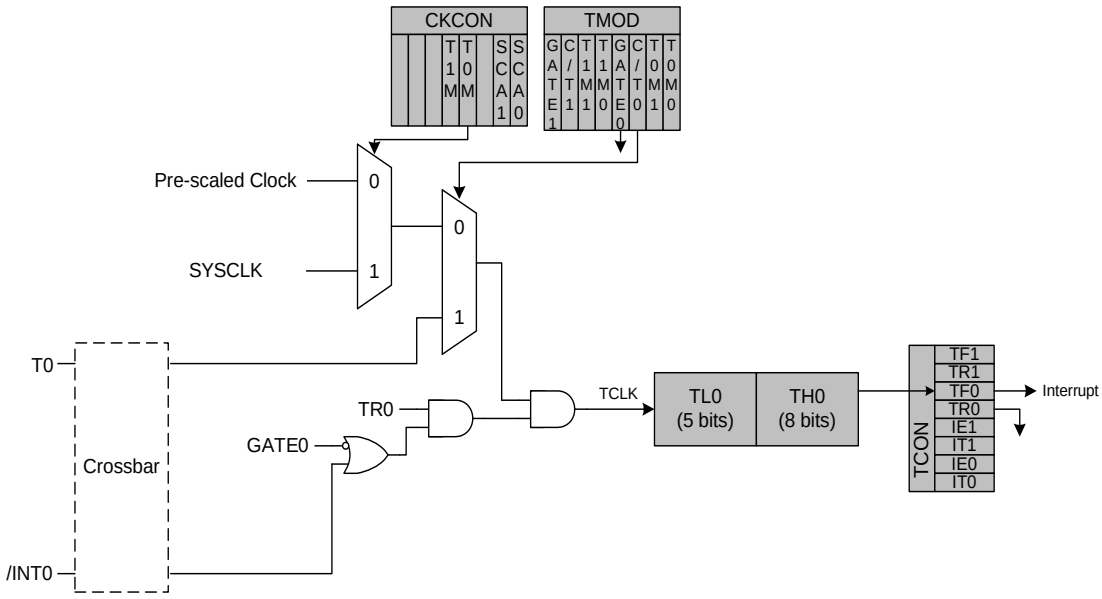

Figure 23.1. T0 Mode 0 Block Diagram