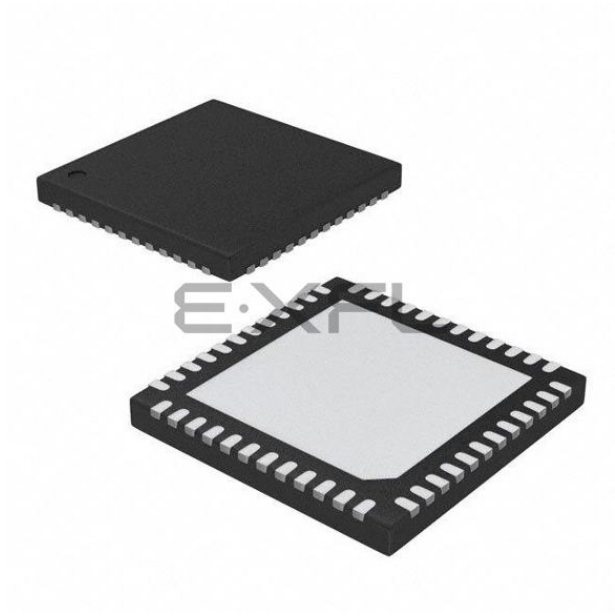


Welcome to E-XFL.COM

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	CIP-51™
Core Size	8-Bit
Speed	25MHz
Connectivity	I²C/SMBus, I²C Slave, SPI, UART/USART
Peripherals	CapSense, DMA, POR, PWM, WDT
Number of I/O	43
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	8K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 43x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	48-WFQFN Exposed Pad
Supplier Device Package	48-QFN (6x6)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f970-a-gm

8. Memory Organization

The memory organization of the CIP-51 System Controller is similar to that of a standard 8051. There are two separate memory spaces: program memory and data memory. Program and data memory share the same address space but are accessed via different instruction types. The memory organization of the C8051F97x device family is shown in Figure 8.1.

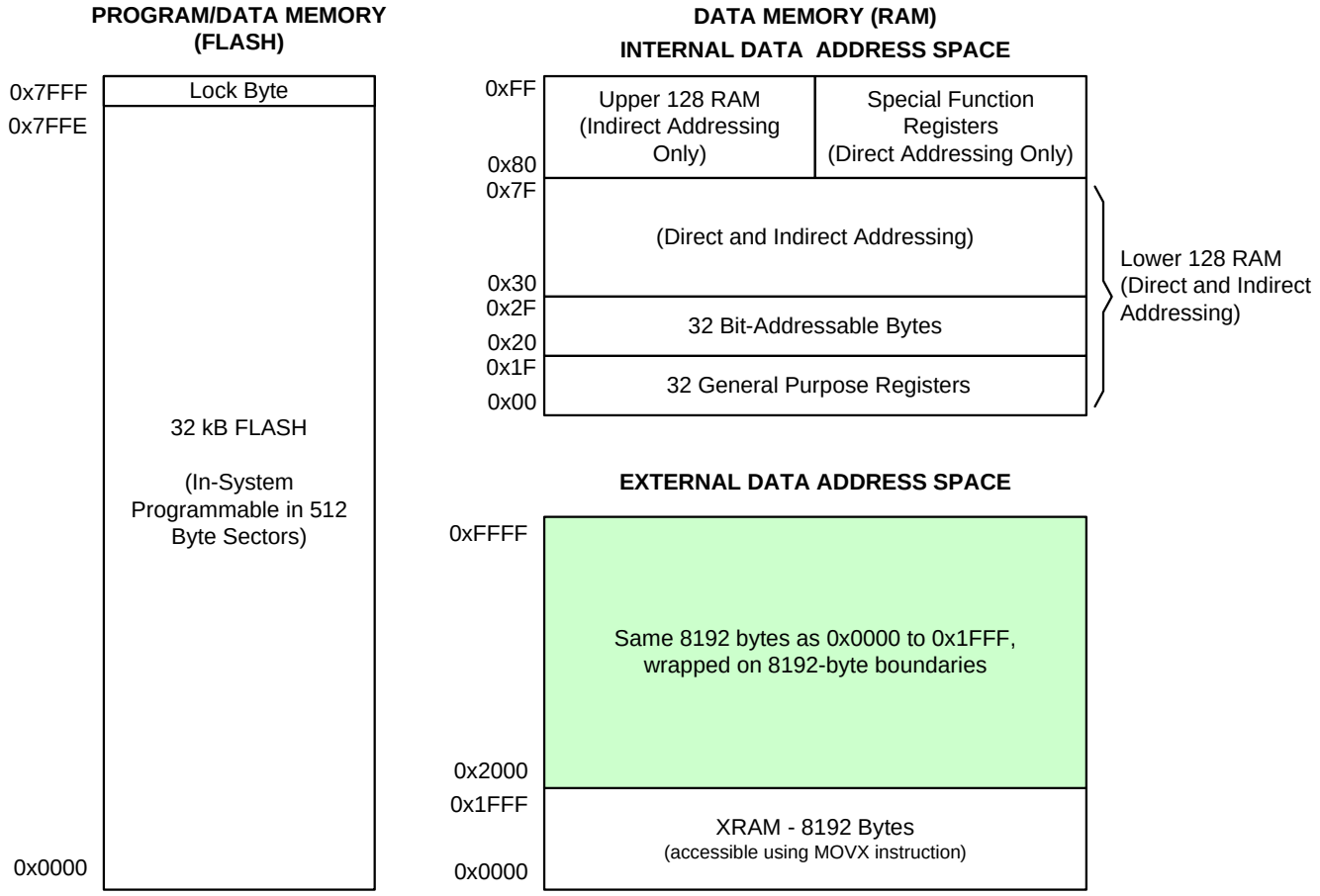


Figure 8.1. C8051F97x Memory Map (32 kB Flash Version Shown)

11.2. External Memory Interface Registers

Register 11.1. EMI0CN: External Memory Interface Control

Bit	7	6	5	4	3	2	1	0
Name	PGSEL							
Type	RW							
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0; SFR Address: 0xB9								

Bit	Name	Function
7:0	PGSEL	**XRAM Page Select.** The XRAM Page Select field provides the high byte of the 16-bit external data memory address when using an 8-bit MOVX command, effectively selecting a 256-byte page of RAM. 0x00: 0x0000 to 0x00FF 0x01: 0x0100 to 0x01FF ... 0xFE: 0xFE00 to 0xFEFF 0xFF: 0xFF00 to 0xFFFF

f

Register 12.2. REVID: Revision Identifcation

Bit	7	6	5	4	3	2	1	0
Name	REVID							
Type	R							
Reset	X	X	X	X	X	X	X	X
SFR Page = 0xF; SFR Address: 0xE2								

Table 12.3. REVID Register Bit Descriptions

Bit	Name	Function
7:0	REVID	**Revision ID.** This read-only register returns the 8-bit revision ID. 00000000: Reserved. 00000001: Revision A 00000010-11111111: Reserved.

16.3. 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 execution. In Stop mode the precision internal oscillator and CPU are stopped; the state of the low power oscillator and the external oscillator circuit is not affected. Each analog peripheral (including the external oscillator circuit) may 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 in stop mode for longer than the MCD timeout.

Stop mode is a legacy 8051 power mode; it will not result in optimal power savings. Sleep or suspend mode will provide more power savings if the MCU needs to be inactive for a long period of time.

Note: To ensure the MCU enters a low power state upon entry into stop mode, the one-shot circuit should be enabled by clearing the BYPASS bit (FLSCL.6).

16.4. Suspend Mode

Setting the Suspend Mode Select bit (PMU0CF.6) causes the system clock to be gated off and all internal oscillators disabled. All digital logic (timers, communication peripherals, interrupts, CPU, etc.) stops functioning until one of the enabled wake-up sources occurs.

The following wake-up sources can be configured to wake the device from suspend mode:

- SmaRTClock oscillator fail
- SmaRTClock alarm
- Port Match event
- I2C0 address match
- CS0 comparator threshold event

Note: Upon wake-up from suspend mode, PMU0 requires two system clocks in order to update the PMU0CF wake-up flags. All flags will read back a value of '0' during the first two system clocks following a wake-up from suspend mode. The state of the wake-up source's interrupt indicator bit is not valid until 6 clock cycles after the device returns from suspend mode. If firmware needs to check a wake-up source's interrupt flag, firmware should insert instructions to wait 6 clock cycles between the call to enter suspend mode and the instruction that polls the interrupt flag.

In addition, a noise glitch on $\overline{\text{RST}}$ that is not long enough to reset the device will cause the device to exit suspend. In order for the MCU to respond to the pin reset event, software must not place the device back into suspend mode for a period of 15 µs. The PMU0CF register may be checked to determine if the wake-up was due to a falling edge on the $\overline{\text{RST}}$ pin. If the wake-up source is not due to a falling edge on $\overline{\text{RST}}$, there is no time restriction on how soon software may place the device back into suspend mode. A 4.7 kΩ pullup resistor to V_{DD} is recommend for $\overline{\text{RST}}$ to prevent noise glitches from waking the device.

16.5. Sleep Mode

Setting the Sleep Mode Select bit (PMU0CF.6) turns off the internal 1.8 V regulator (VREG0) and switches the power supply of all on-chip RAM to the V_{DD} pin (see Figure 16.2). Power to most digital logic on the chip is disconnected; only PMU0 and the SmaRTClock remain powered. All analog peripherals (ADC0, External Oscillator, etc.) should be disabled prior to entering sleep mode.

GPIO pins configured as digital outputs will retain their output state during sleep mode. They will maintain the same current drive capability in sleep mode as they have in normal active mode.

GPIO pins configured as digital inputs can be used during sleep mode as wakeup sources using the port match feature. They will maintain the same input level specifications in sleep mode as they have in normal active mode.

C8051F97x devices support a wakeup request for external devices. Upon exit from sleep mode, the wake-up request signal is driven low, allowing other devices in the system to wake up from their low power modes. The wakeup request signal is low when the MCU is awake and high when the MCU is asleep.

18.14.1. Pin Configuration for CS0 Measurements Method

A port pin selected as CS0 input should be configured as follows:

1. Set to analog mode input by clearing to 0 the corresponding bit in register PnMDIN.
2. Force the Priority Crossbar Decoder to skip the pin by setting 1 to the corresponding bit in register PnSKIP.
3. Enable or disable the auto-ground for the pin by clearing 0 or setting 1 to the corresponding bit in the port latch (Pn), respectively. Auto-grounding means that the pin will be grounded when CS0 measurement is not being performed on the pin.
4. Set to 1 the corresponding bits in AMUX0Pn that CS0 will be taking measurements on.
5. If only a single channel is to be sensed, setup the CS0 Multiplexer to select the appropriate pin for measurement. If automatic scanning is used, setup CS0SS and CS0SE registers. If multiple channels are to be binded, the CS0MC

Important Notes:

- When CS0 is active, ADC0 must not be enabled even if ADC0 is not going to perform an operation.
- Similarly, when ADC0 is active, CS0 should not be enabled.

See Section “26. Port I/O (Port 0, Port 1, Port 2, Port 3, Port 4, Port 5, Port 6, Crossbar, and Port Match)” on page 277 for more Port I/O configuration details.

Bit	Name	Function
5:0	CS0MX	**CS0 Mux Channel Select.** Selects a single input channel for Capacitive Sense conversion. 000000: Select CS0.0. 000001: Select CS0.1. 000010: Select CS0.2. 000011: Select CS0.3. 000100: Select CS0.4. 000101: Select CS0.5. 000110: Select CS0.6. 000111: Select CS0.7. 001000: Select CS0.8. 001001: Select CS0.9. 001010: Select CS0.10. 001011: Select CS0.11. 001100: Select CS0.12. 001101: Select CS0.13. 001110: Select CS0.14. 001111: Select CS0.15. 010000: Select CS0.16. 010001: Select CS0.17. 010010: Select CS0.18. 010011: Select CS0.19. 010100: Select CS0.20. 010101: Select CS0.21. 010110: Select CS0.22. 010111: Select CS0.23. 011000: Select CS0.24. 011001: Select CS0.25. 011010: Select CS0.26. 011011: Select CS0.27. 011100: Select CS0.28. 011101: Select CS0.29. 011110: Select CS0.30. 011111: Select CS0.31. 100000: Select CS0.32. 100001: Select CS0.33. 100010: Select CS0.34. 100011: Select CS0.35. 100100: Select CS0.36. 100101: Select CS0.37. 100110: Select CS0.38. 100111: Select CS0.39. 101000: Select CS0.40. 101001: Select CS0.41. 101010: Select CS0.42. 101011-111111: Reserved.

Register 20.7. SFRPAGE: SFR Page

Bit	7	6	5	4	3	2	1	0
Name	SFRPAGE							
Type	RW							
Reset	0	0	0	0	0	0	0	0
SFR Page = ALL; SFR Address: 0xA7								

Table 20.8. SFRPAGE Register Bit Descriptions

Bit	Name	Function
7:0	SFRPAGE	**SFR Page.** Specifies the SFR Page used when reading, writing, or modifying special function registers.

25. SmaRTClock (Real Time Clock, RTC0)

C8051F97x devices include an ultra low power 32-bit SmaRTClock Peripheral (Real Time Clock) with alarm. The SmaRTClock has a dedicated 32 kHz oscillator that can be configured for use with or without a crystal. No external resistor or loading capacitors are required. The on-chip loading capacitors are programmable to 16 discrete levels allowing compatibility with a wide range of crystals. The SmaRTClock can operate directly from a 0.9–3.6 V battery voltage and remains operational even when the device goes into its lowest power down mode. C8051F97x devices also support an ultra low power internal LFO that reduces sleep mode current.

The SmaRTClock allows a maximum of 36 hour 32-bit independent time-keeping when used with a 32.768 kHz Watch Crystal. The SmaRTClock provides an Alarm and Missing SmaRTClock events, which could be used as reset or wakeup sources. See Section “27. Reset Sources and Supply Monitor” on page 322 and Section “16. Power Management” on page 94 for details on reset sources and low-power mode wake-up sources, respectively.

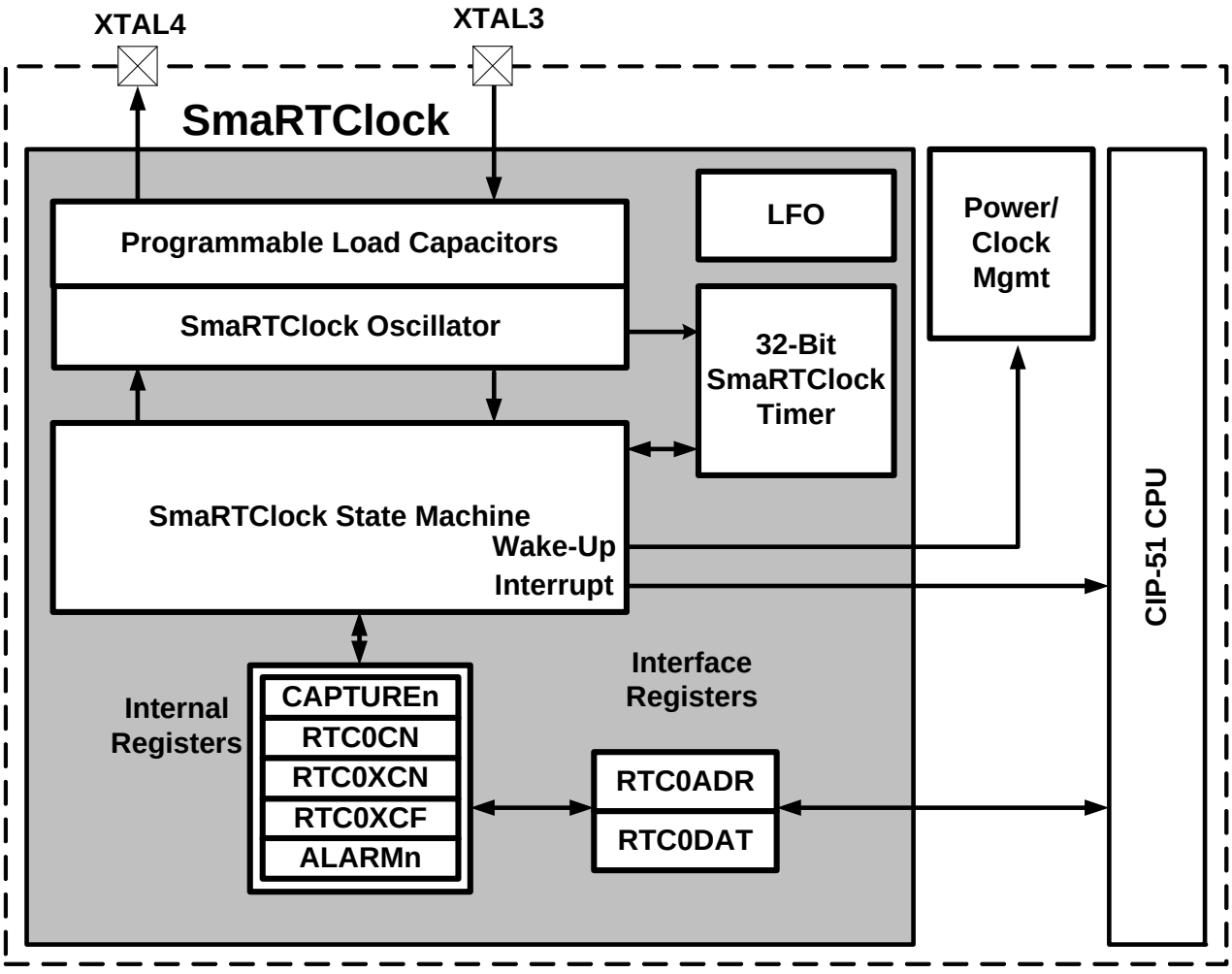


Figure 25.1. SmaRTClock Block Diagram

Register 25.2. RTC0DAT: RTC Data

Bit	7	6	5	4	3	2	1	0
Name	RTC0DAT							
Type	RW							
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0; SFR Address: 0xAD								

Bit	Name	Function
7:0	RTC0DAT	**RTC Data.** Holds data transferred to/from the internal RTC register selected by RTC0ADR.
Note: Read-modify-write instructions (orl, anl, etc.) should not be used on this register.		

Register 25.13. ALARM3: RTC Alarm Programmed Value 3

Bit	7	6	5	4	3	2	1	0
Name	ALARM3							
Type	RW							
Reset	0	0	0	0	0	0	0	0
Indirect Address: 0x0B								

Bit	Name	Function
7:0	ALARM3	**RTC Alarm Programmed Value 3.** The ALARM3-ALARM0 registers are used to set an alarm event for the RTC timer. The RTC alarm should be disabled (RTC0AEN=0) when updating these registers.

28.1. Signal Descriptions

The four signals used by SPI0 (MOSI, MISO, SCK, NSS) are described below.

28.1.1. Master Out, Slave In (MOSI)

The master-out, slave-in (MOSI) signal is an output from a master device and an input to slave devices. It is used to serially transfer data from the master to the slave. This signal is an output when SPI0 is operating as a master and an input when SPI0 is operating as a slave. Data is transferred most-significant bit first. When configured as a master, MOSI is driven by the MSB of the shift register in both 3- and 4-wire mode.

28.1.2. Master In, Slave Out (MISO)

The master-in, slave-out (MISO) signal is an output from a slave device and an input to the master device. It is used to serially transfer data from the slave to the master. This signal is an input when SPI0 is operating as a master and an output when SPI0 is operating as a slave. Data is transferred most-significant bit first. The MISO pin is placed in a high-impedance state when the SPI module is disabled and when the SPI operates in 4-wire mode as a slave that is not selected. When acting as a slave in 3-wire mode, MISO is always driven by the MSB of the shift register.

28.1.3. Serial Clock (SCK)

The serial clock (SCK) signal is an output from the master device and an input to slave devices. It is used to synchronize the transfer of data between the master and slave on the MOSI and MISO lines. SPI0 generates this signal when operating as a master. The SCK signal is ignored by a SPI slave when the slave is not selected (NSS = 1) in 4-wire slave mode.

28.1.4. Slave Select (NSS)

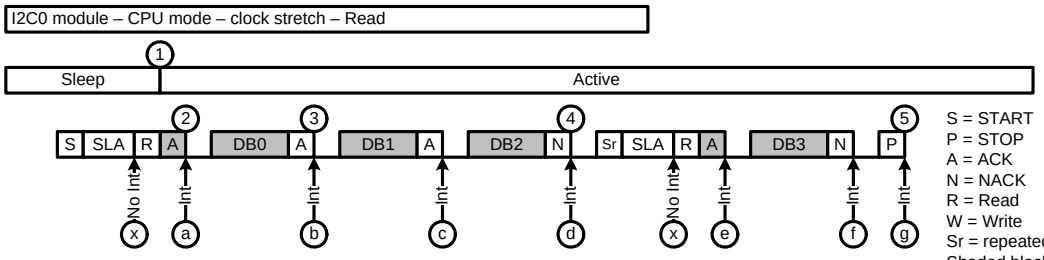
The function of the slave-select (NSS) signal is dependent on the setting of the NSSMD1 and NSSMD0 bits in the SPI0CN register. There are three possible modes that can be selected with these bits:

1. NSSMD[1:0] = 00: 3-Wire Master or 3-Wire Slave Mode: SPI0 operates in 3-wire mode, and NSS is disabled. When operating as a slave device, SPI0 is always selected in 3-wire mode. Since no select signal is present, SPI0 must be the only slave on the bus in 3-wire mode. This is intended for point-to-point communication between a master and one slave.
2. NSSMD[1:0] = 01: 4-Wire Slave or Multi-Master Mode: SPI0 operates in 4-wire mode, and NSS is enabled as an input. When operating as a slave, NSS selects the SPI0 device. When operating as a master, a 1-to-0 transition of the NSS signal disables the master function of SPI0 so that multiple master devices can be used on the same SPI bus.
3. NSSMD[1:0] = 1x: 4-Wire Master Mode: SPI0 operates in 4-wire mode, and NSS is enabled as an output. The setting of NSSMD0 determines what logic level the NSS pin will output. This configuration should only be used when operating SPI0 as a master device.

See Figure 28.2, Figure 28.3, and Figure 28.4 for typical connection diagrams of the various operational modes. **Note that the setting of NSSMD bits affects the pinout of the device.** When in 3-wire master or 3-wire slave mode, the NSS pin will not be mapped by the crossbar. In all other modes, the NSS signal will be mapped to a pin on the device.

30.5.2. I²C Read Sequence (CPU mode)

Figure 30.6 shows the details of how the I2C0STAT status bits change during an I²C Read data transfer.



1. SLA+R wakes CPU from sleep. SLA is ACK if BUSY was 0.
2. INT generated. CPU writes DB0 into I2C0DOUT. SCL is held until CPU clears I2C0INT. CPU clears I2C0INT, releasing SCL
3. SCL held low; NACK bit reads '0'; CPU writes DB1 into I2C0DOUT and clears I2C0INT
4. NACK switches I2C slave to IDLE state and generates INT. Any following DB are ignored. If RpStart follows after NACK, the START sticky bit will be set, but no interrupt.
5. STOP generates interrupt. No Clock Stretch. CPU clears I2C0INT

No int (x) I2C0STAT = x1001000 at 8th SCL rising edge

I2C0 int (a) I2C0STAT = x1101001; CPU clears START. CPU writes DB0 into I2C0DOUT and clears I2C0INT

I2C0 int (b) I2C0STAT = x1100001; CPU writes DB1 into I2C0DOUT and clears I2C0INT

I2C0 int (c) I2C0STAT = x1100001; CPU writes DB2 into I2C0DOUT and clears I2C0INT

I2C0 int (d) I2C0STAT = x0110001; CPU prepares for STOP or RpStart and clears I2C0INT and clears NACK

I2C0 int (e) I2C0STAT = x1101001; CPU clears START. CPU writes DB3 into I2C0DOUT and clears I2C0INT

I2C0 int (f) I2C0STAT = x0110001; CPU prepares for STOP or RpStart and clears I2C0INT and clears NACK

I2C0 int (g) I2C0STAT = x0100100; CPU clears STOP and clears I2C0INT

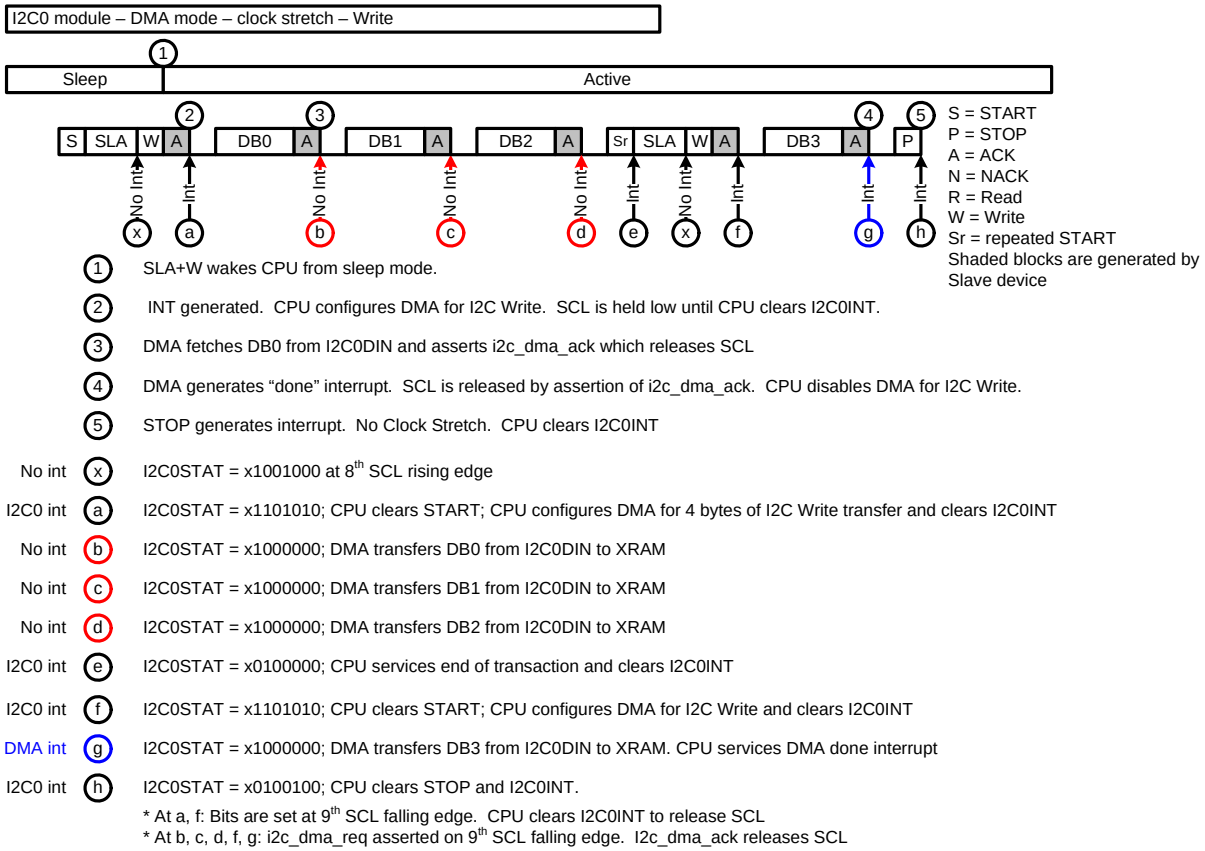
* At a, b, c, d, e, f : Bits are set/cleared at 9th SCL falling edge. CPU clears I2C0INT to release SCL

Figure 30.6. Typical I2C Read Sequence in CPU Mode

Note that the I²C Master MUST always generate a NACK before it can generate a repeated START bit or a STOP bit. This is because the NACK will cause the I²C Slave to release the SDA line for the I²C Master to generate the START or STOP bit.

30.5.3. I²C Write Sequence (DMA mode)

Figure 30.7 shows the details of how the I2C0STAT status bits change during an I²C Write data transfer.



	1	SLA+W wakes CPU from sleep mode.
	2	INT generated. CPU configures DMA for I2C Write. SCL is held low until CPU clears I2C0INT.
	3	DMA fetches DB0 from I2C0DIN and asserts i2c_dma_ack which releases SCL
	4	DMA generates “done” interrupt. SCL is released by assertion of i2c_dma_ack. CPU disables DMA for I2C Write.
	5	STOP generates interrupt. No Clock Stretch. CPU clears I2C0INT
No int	x	I2C0STAT = x1001000 at 8th SCL rising edge
I2C0 int	a	I2C0STAT = x1101010; CPU clears START; CPU configures DMA for 4 bytes of I2C Write transfer and clears I2C0INT
No int	b	I2C0STAT = x1000000; DMA transfers DB0 from I2C0DIN to XRAM
No int	c	I2C0STAT = x1000000; DMA transfers DB1 from I2C0DIN to XRAM
No int	d	I2C0STAT = x1000000; DMA transfers DB2 from I2C0DIN to XRAM
I2C0 int	e	I2C0STAT = x0100000; CPU services end of transaction and clears I2C0INT
I2C0 int	f	I2C0STAT = x1101010; CPU clears START; CPU configures DMA for I2C Write and clears I2C0INT
DMA int	g	I2C0STAT = x1000000; DMA transfers DB3 from I2C0DIN to XRAM. CPU services DMA done interrupt
I2C0 int	h	I2C0STAT = x0100100; CPU clears STOP and I2C0INT.

* At a, f: Bits are set at 9th SCL falling edge. CPU clears I2C0INT to release SCL
* At b, c, d, f, g: i2c_dma_req asserted on 9th SCL falling edge. I2c_dma_ack releases SCL

Figure 30.7. Typical I²C Write Sequence in DMA Mode

31. Universal Asynchronous Receiver/Transmitter (UART0)

UART0 is an asynchronous, full duplex serial port offering modes 1 and 3 of the standard 8051 UART. Enhanced baud rate support allows a wide range of clock sources to generate standard baud rates (details in Section “31.1. Enhanced Baud Rate Generation” on page 380). Received data buffering allows UART0 to start reception of a second incoming data byte before software has finished reading the previous data byte.

UART0 has two associated SFRs: Serial Control Register 0 (SCON0) and Serial Data Buffer 0 (SBUF0). The single SBUF0 location provides access to both transmit and receive registers. **Writes to SBUF0 always access the transmit register. Reads of SBUF0 always access the buffered receive register; it is not possible to read data from the transmit register.**

With UART0 interrupts enabled, an interrupt is generated each time a transmit is completed (TI is set in SCON0), or a data byte has been received (RI is set in SCON0). The UART0 interrupt flags are not cleared by hardware when the CPU vectors to the interrupt service routine. They must be cleared manually by software, allowing software to determine the cause of the UART0 interrupt (transmit complete or receive complete).

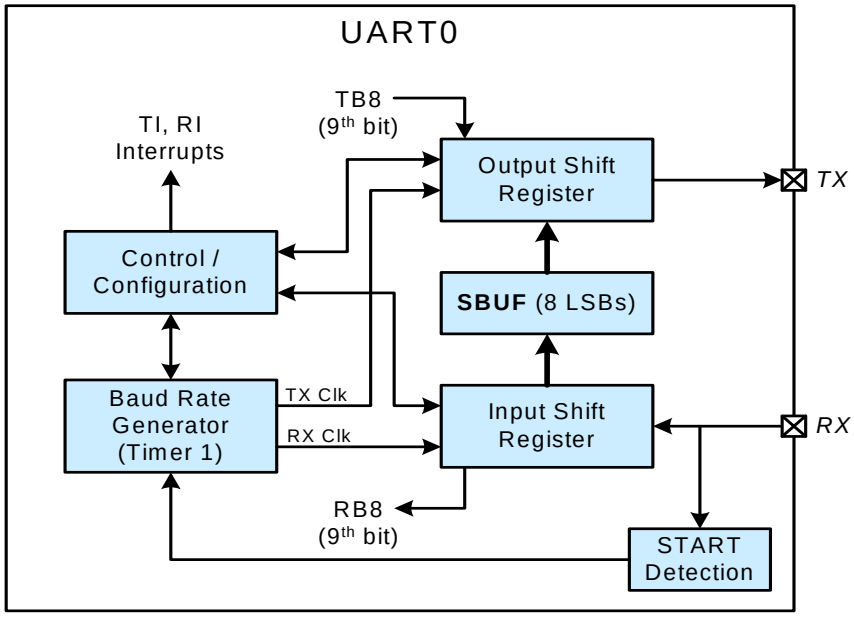


Figure 31.1. UART0 Block Diagram

31.1. Enhanced Baud Rate Generation

The UART0 baud rate is generated by Timer 1 in 8-bit auto-reload mode. The TX clock is generated by TL1; the RX clock is generated by a copy of TL1 (shown as RX Timer in Figure 31.2), which is not user-accessible. Both TX and RX Timer overflows are divided by two to generate the TX and RX baud rates. The RX Timer runs when Timer 1 is enabled, and uses the same reload value (TH1). However, an RX Timer reload is forced when a START condition is detected on the RX pin. This allows a receive to begin any time a START is detected, independent of the TX Timer state.

32.1. Timer 0 and Timer 1

Timer 0 and Timer 1 are each implemented as a16-bit register accessed as two separate bytes: a low byte (TL0 or TL1) and a high byte (TH0 or TH1). The Counter/Timer Control register (TCON) is used to enable Timer 0 and Timer 1 as well as indicate status. Timer 0 interrupts can be enabled by setting the ET0 bit in the IE register. Timer 1 interrupts can be enabled by setting the ET1 bit in the IE register. Both counter/timers operate in one of four primary modes selected by setting the Mode Select bits T1M1–T0M0 in the Counter/Timer Mode register (TMOD). Each timer can be configured independently for the operating modes described below.

32.2.2. 8-bit Timers with Auto-Reload

When T2SPLIT is set, Timer 2 operates as two 8-bit timers (TMR2H and TMR2L). Both 8-bit timers operate in auto-reload mode as shown in Figure 32.5. TMR2RLL holds the reload value for TMR2L; TMR2RLH holds the reload value for TMR2H. The TR2 bit in TMR2CN handles the run control for TMR2H. TMR2L is always running when configured for 8-bit Mode.

Each 8-bit timer may be configured to use SYSCLK, SYSCLK divided by 12, SmaRTClock divided by 8 or Comparator 0 output. The Timer 2 Clock Select bits (T2MH and T2ML in CKCON) select either SYSCLK or the clock defined by the Timer 2 External Clock Select bits (T2XCLK[1:0] in TMR2CN), as follows:

T2MH	T2XCLK[1:0]	TMR2H Clock Source
0	00	SYSCLK / 12
0	01	SmaRTClock / 8
0	10	Reserved
0	11	Comparator 0
1	X	SYSCLK

T2ML	T2XCLK[1:0]	TMR2L Clock Source
0	00	SYSCLK / 12
0	01	SmaRTClock / 8
0	10	Reserved
0	11	Comparator 0
1	X	SYSCLK

The TF2H bit is set when TMR2H overflows from 0xFF to 0x00; the TF2L bit is set when TMR2L overflows from 0xFF to 0x00. When Timer 2 interrupts are enabled (IE.5), an interrupt is generated each time TMR2H overflows. If Timer 2 interrupts are enabled and TF2LINT (TMR2CN.5) is set, an interrupt is generated each time either TMR2L or TMR2H overflows. When TF2LINT is enabled, software must check the TF2H and TF2L flags to determine the source of the Timer 2 interrupt. The TF2H and TF2L interrupt flags are not cleared by hardware and must be manually cleared by software.

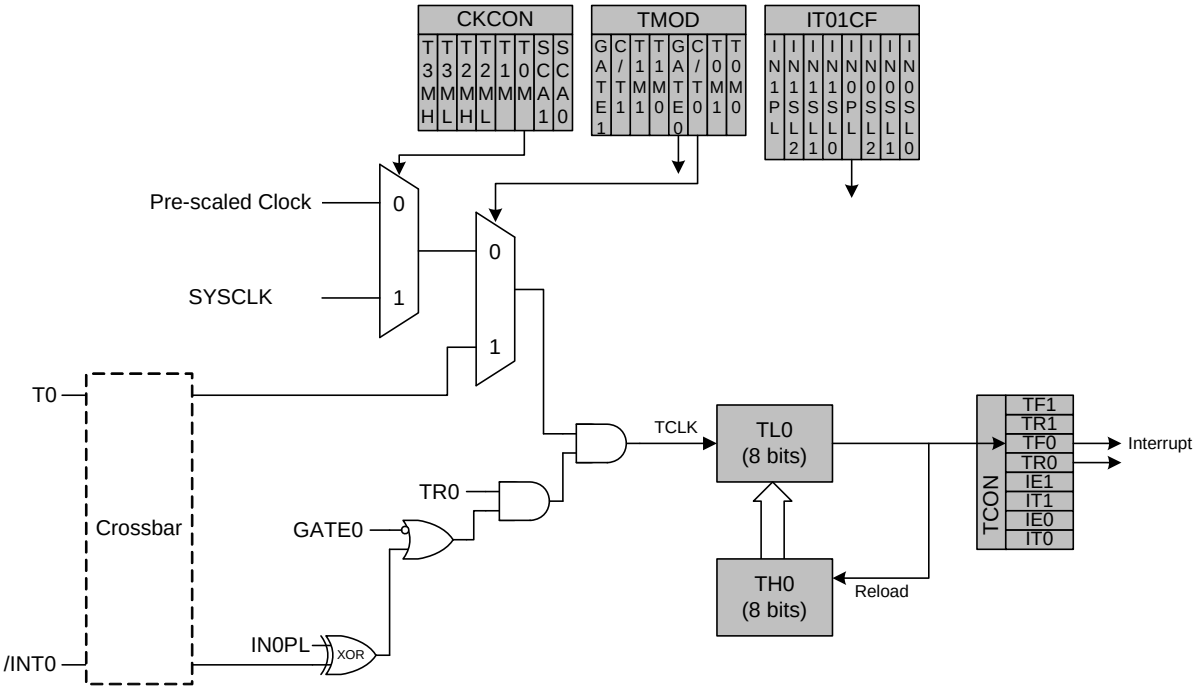


Figure 32.5. Timer 2 8-Bit Mode Block Diagram

Register 32.10. TMR2RLH: Timer 2 Reload High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR2RLH							
Type	RW							
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0; SFR Address: 0xCB								

Table 32.12. TMR2RLH Register Bit Descriptions

Bit	Name	Function
7:0	TMR2RLH	**Timer 2 Reload High Byte.** When operating in one of the auto-reload modes, TMR2RLH holds the reload value for the high byte of Timer 2 (TMR2H). When operating in capture mode, TMR2RLH is the captured value of TMR2H.

33.3.2. Software Timer (Compare) Mode

In Software Timer mode, the PCA counter/timer value is compared to the module's 16-bit capture/compare register (PCA0CPHn and PCA0CPLn). When a match occurs, the Capture/Compare Flag (CCFn) in PCA0CN is set to logic 1. An interrupt request is generated if the CCFn interrupt for that module is enabled. The CCFn bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. Setting the ECOMn and MATn bits in the PCA0CPMn register enables Software Timer mode.

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.

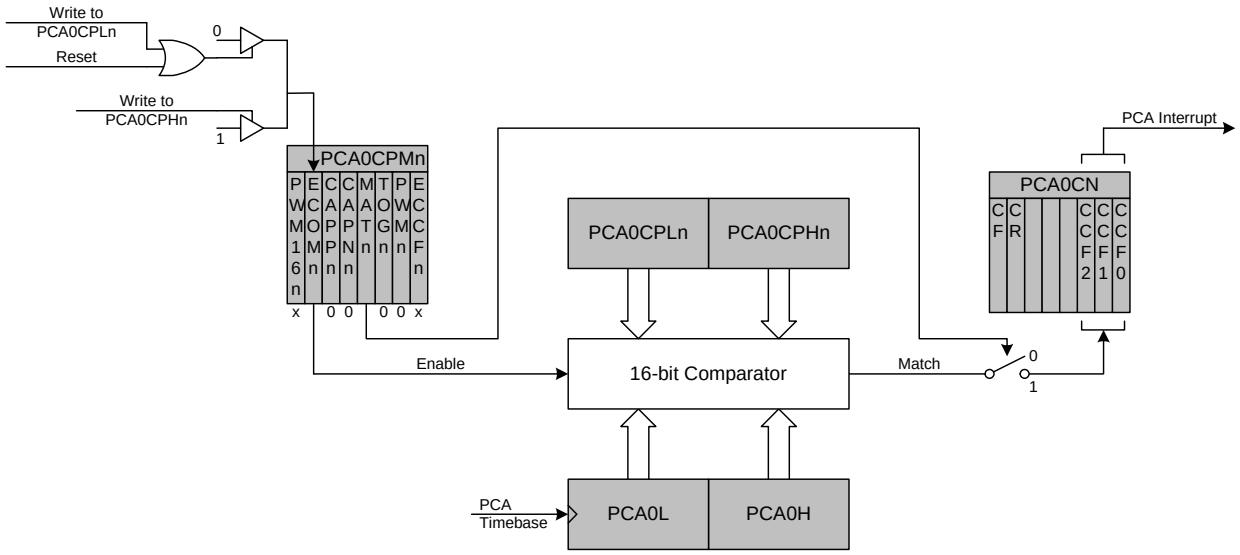


Figure 33.5. PCA Software Timer Mode Diagram

Register 33.14. PCA0CPH2: PCA Channel 2 Capture Module High Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPH2							
Type	RW							
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0; SFR Address: 0xEC								

Table 33.17. PCA0CPH2 Register Bit Descriptions

Bit	Name	Function
7:0	PCA0CPH2	**PCA Channel 2 Capture Module High Byte.** The PCA0CPH2 register holds the high byte (MSB) of the 16-bit capture module. This register address also allows access to the high byte of the corresponding PCA channel's auto-reload value for 9 to 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed.
Note: A write to this register will set the module's ECOM bit to a 1.		

34. C2 Interface

C8051F97x devices include an on-chip Silicon Labs 2-Wire (C2) debug interface to allow flash programming and in-system debugging with the production part installed in the end application. The C2 interface uses a clock signal (C2CK) and a bidirectional C2 data signal (C2D) to transfer information between the device and a host system. Details on the C2 protocol can be found in the C2 Interface Specification.

34.1. C2 Pin Sharing

The C2 protocol allows the C2 pins to be shared with user functions so that in-system debugging and flash programming may be performed. C2CK is shared with the RST pin, while the C2D signal is shared with a port I/O pin. This is possible because C2 communication is typically performed when the device is in the halt state, where all on-chip peripherals and user software are stalled. In this halted state, the C2 interface can safely “borrow” the C2CK and C2D pins. In most applications, external resistors are required to isolate C2 interface traffic from the user application. A typical isolation configuration is shown in Figure 34.1.

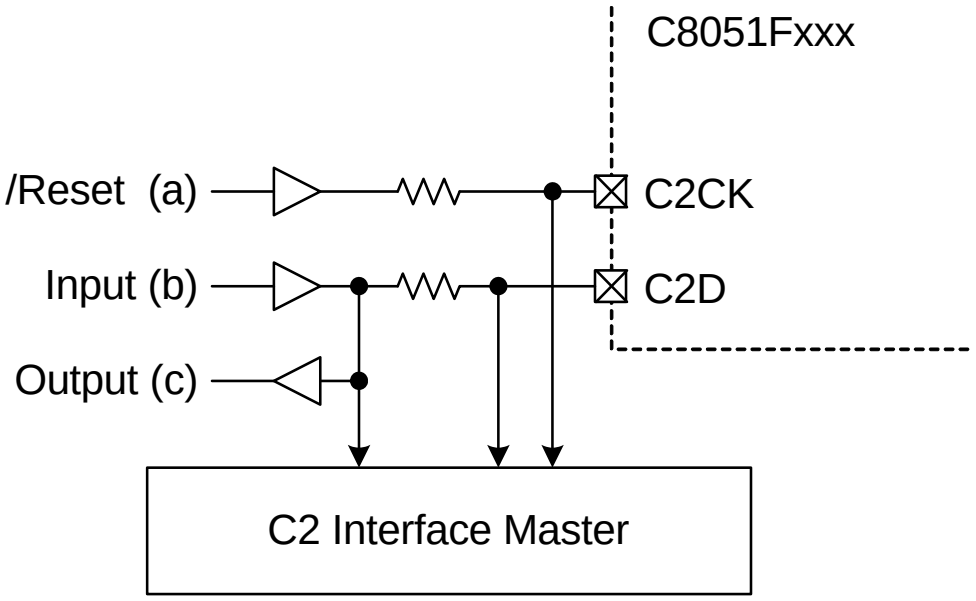


Figure 34.1. Typical C2 Pin Sharing

The configuration in Figure 34.1 assumes the following:

1. The user input (b) cannot change state while the target device is halted.
2. The RST pin on the target device is used as an input only.

Additional resistors may be necessary depending on the specific application.