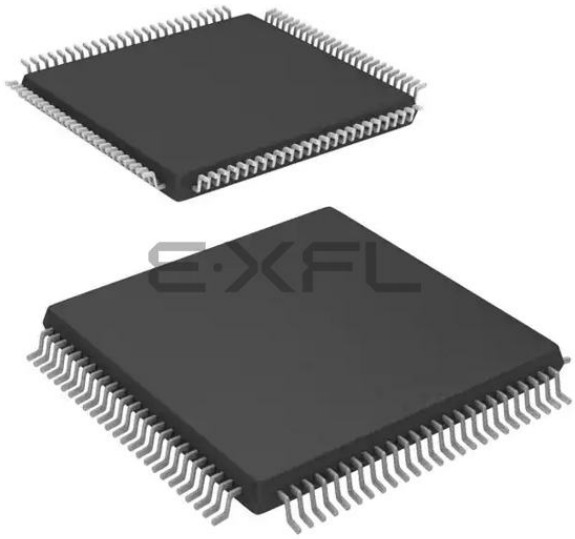


Welcome to E-XFL.COM

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	25MHz
Connectivity	EBI/EMI, SMBus (2-Wire/I²C), SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	59
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	4.25K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	A/D 2x16b
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/c8051f066r

26.2.Flash Programming Commands 322
26.3.Debug Support 325
27. Document Change List 327
27.1.Revision 1.1 to Revision 1.2 327

Figure 18.5. XBR0: Port I/O Crossbar Register 0 210
Figure 18.6. XBR1: Port I/O Crossbar Register 1 211
Figure 18.7. XBR2: Port I/O Crossbar Register 2 212
Figure 18.8. XBR3: Port I/O Crossbar Register 3 213
Figure 18.9. P0: Port0 Data Register 214
Figure 18.10. P0MDOUT: Port0 Output Mode Register 214
Figure 18.11. P1: Port1 Data Register 215
Figure 18.12. P1MDIN: Port1 Input Mode Register 215
Figure 18.13. P1MDOUT: Port1 Output Mode Register 216
Figure 18.14. P2: Port2 Data Register 216
Figure 18.15. P2MDIN: Port2 Input Mode Register 217
Figure 18.16. P2MDOUT: Port2 Output Mode Register 217
Figure 18.17. P3: Port3 Data Register 218
Figure 18.18. P3MDOUT: Port3 Output Mode Register 218
Figure 18.19. P4: Port4 Data Register 221
Figure 18.20. P4MDOUT: Port4 Output Mode Register 221
Figure 18.21. P5: Port5 Data Register 222
Figure 18.22. P5MDOUT: Port5 Output Mode Register 222
Figure 18.23. P6: Port6 Data Register 223
Figure 18.24. P6MDOUT: Port6 Output Mode Register 223
Figure 18.25. P7: Port7 Data Register 224
Figure 18.26. P7MDOUT: Port7 Output Mode Register 224
19. Controller Area Network (CAN0, C8051F060/1/2/3) 225
Figure 19.1. CAN Controller Diagram 226
Figure 19.2. Typical CAN Bus Configuration 226
Figure 19.3. CAN0DATH: CAN Data Access Register High Byte 231
Figure 19.4. CAN0DATL: CAN Data Access Register Low Byte 231
Figure 19.5. CAN0ADR: CAN Address Index Register 232
Figure 19.6. CAN0CN: CAN Control Register 232
Figure 19.7. CAN0TST: CAN Test Register 233
Figure 19.8. CAN0STA: CAN Status Register 233
20. System Management BUS / I2C BUS (SMBUS0) 235
Figure 20.1. SMBus0 Block Diagram 235
Figure 20.2. Typical SMBus Configuration 236
Figure 20.3. SMBus Transaction 237
Figure 20.4. Typical Master Transmitter Sequence 238
Figure 20.5. Typical Master Receiver Sequence 238
Figure 20.6. Typical Slave Transmitter Sequence 239
Figure 20.7. Typical Slave Receiver Sequence 240
Figure 20.8. SMB0CN: SMBus0 Control Register 243
Figure 20.9. SMB0CR: SMBus0 Clock Rate Register 244
Figure 20.10. SMB0DAT: SMBus0 Data Register 245
Figure 20.11. SMB0ADR: SMBus0 Address Register 246
Figure 20.12. SMB0STA: SMBus0 Status Register 247
21. Enhanced Serial Peripheral Interface (SPI0) 251

Table 1.1. Product Selection Guide

	MIPS (Peak)	Flash Memory	RAM	External Memory Interface	SMBus/I2C and SPI	CAN	UARTS	Timers (16-bit)	Programmable Counter Array	Digital Port I/O's	16-bit 1 Msps ADC Typical INL (LSBs)	10-bit 200 ksps ADC Inputs	Voltage Reference	Temperature Sensor	DAC Resolution (bits)	DAC Outputs	Analog Comparators	Package
C8051F060	25	64 k	4352	✓	✓	✓	2	5	✓	59	±0.75	8	✓	✓	12	2	3	100 TQFP
C8051F061	25	64 k	4352	-	✓	✓	2	5	✓	24	±0.75	8	✓	✓	12	2	3	64 TQFP
C8051F062	25	64 k	4352	✓	✓	✓	2	5	✓	59	±1.5	8	✓	✓	12	2	3	100 TQFP
C8051F063	25	64 k	4352	-	✓	✓	2	5	✓	24	±1.5	8	✓	✓	12	2	3	64 TQFP
C8051F064	25	64 k	4352	✓	✓	-	2	5	✓	59	±0.75	-	✓	-	-	-	3	100 TQFP
C8051F065	25	64 k	4352	-	✓	-	2	5	✓	24	±0.75	-	✓	-	-	-	3	64 TQFP
C8051F066	25	32 k	4352	✓	✓	-	2	5	✓	59	±0.75	-	✓	-	-	-	3	100 TQFP
C8051F067	25	32 k	4352	-	✓	-	2	5	✓	24	±0.75	-	✓	-	-	-	3	64 TQFP

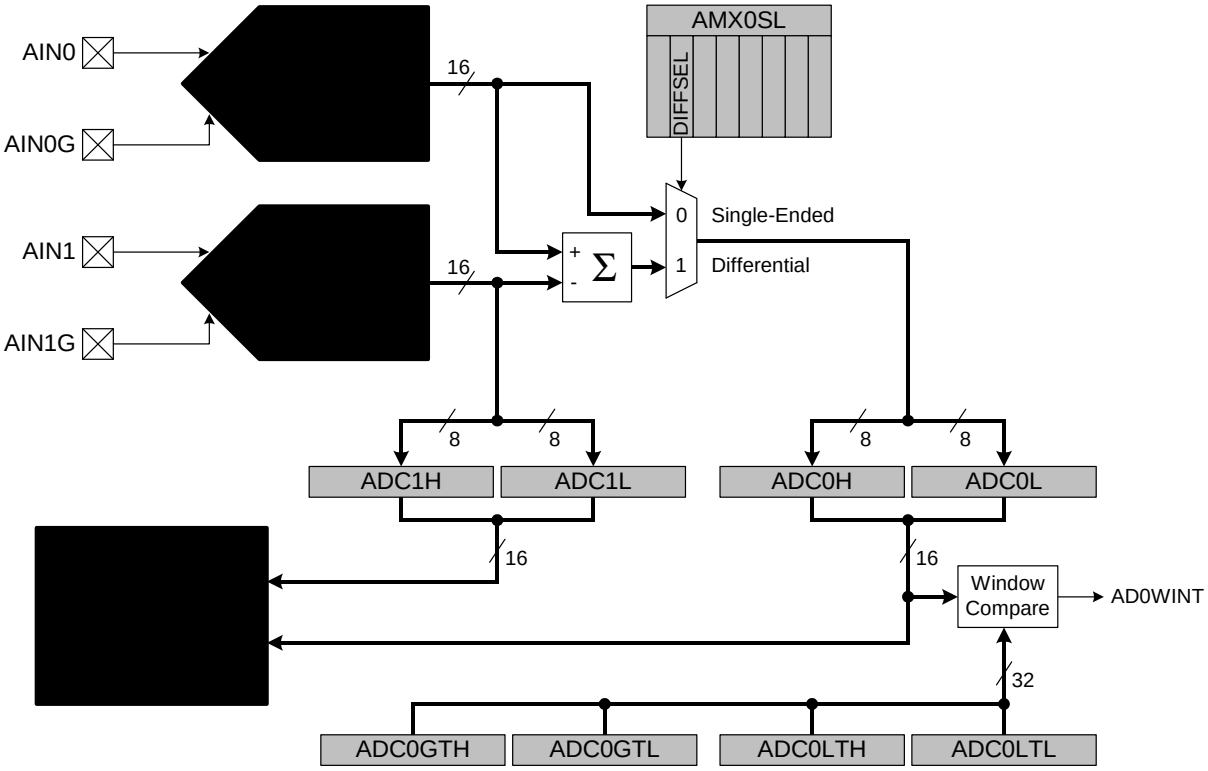


Figure 5.2. 16-bit ADC0 and ADC1 Data Path Diagram

5.1. Single-Ended or Differential Operation

ADC0 and ADC1 can be programmed to operate independently as single-ended ADCs, or together to accept a differential input. In single-ended mode, the ADCs can be configured to sample simultaneously, or to use different conversion speeds. In differential mode, ADC1 is a slave to ADC0, and its configuration is based on ADC0 settings, except during offset or gain calibrations. The DIFFSEL bit in the Channel Select Register AMX0SL (Figure 5.6) selects between single-ended and differential mode.

5.1.1. Pseudo-Differential Inputs

The inputs to the ADCs are pseudo-differential. The actual voltage measured by each ADC is equal to the voltage between the AINn pin and the AINnG pin. AINnG must be a DC signal between -0.2 and 0.6 V. In most systems, AINnG will be connected to AGND. If not tied to AGND, the AINnG signal can be used to negate a limited amount of fixed offset, but it is recommended that the internal offset calibration features of the device be used for this purpose. When operating in differential mode, AIN0G and AIN1G should be tied together. AINn must remain above AINnG in both modes for accurate conversion results.

5.3.3. Settling Time Requirements

The ADC requires a minimum tracking time before an accurate conversion can be performed. This tracking time is determined by the ADC input resistance, the ADC sampling capacitance, any external source resistance, and the accuracy required for the conversion. Figure 5.5 shows the equivalent ADC input circuits for both Differential and Single-ended modes. Notice that the equivalent time constant for both input circuits is the same. The required settling time for a given settling accuracy (*SA*) may be approximated by Equation 5.1. An absolute minimum tracking time of 280 ns is required prior to the start of a conversion.

$$t = \ln\left(\frac{2^n}{SA}\right) \times R_{TOTAL} C_{SAMPLE}$$

Equation 5.1. ADC0 Settling Time Requirements

Where:
SA is the settling accuracy, given as a fraction of an LSB (for example, 0.25 to settle within 1/4 LSB)
t is the required settling time in seconds
R_{TOTAL} is the sum of the ADC input resistance and any external source resistance.
n is the ADC resolution in bits (16).

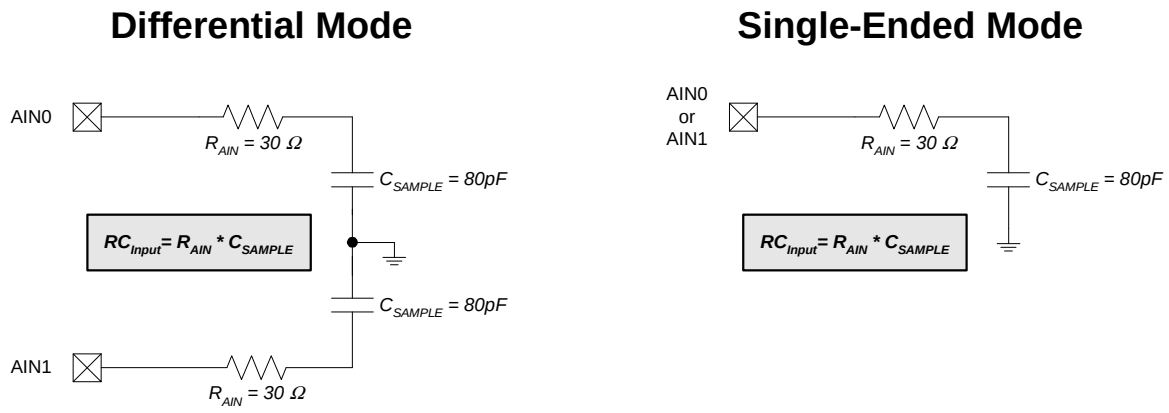


Figure 5.5. ADC0 and ADC1 Equivalent Input Circuits

6.2. DMA0 Instruction Format

DMA instructions can request single-ended data from both ADC0 and ADC1, as well as the differential combination of the two ADC inputs. The instruction format is identical to the DMA0IDT register, shown in Figure 6.7. Depending on which bits are set to ‘1’ in the instruction word, either 2 or 4 bytes of data will be written to XRAM for each DMA instruction cycle (excluding End-Of-Operation instructions). Table 6.1 details all of the valid DMA instructions. Instructions not listed in the table are not valid DMA instructions, and should not be used. Note that the ADCs can be independently controlled by the microcontroller when their outputs are not requested by the DMA.

Table 6.1. DMA0 Instruction Set

Instruction Word	Description	First Data Written to XRAM (2 bytes)	Second Data Written to XRAM (2 bytes)
00000000b	End-Of-Operation	none	none
10000000b	End-Of-Operation with Continuous Conversion	none	none
x0010000b	Retrieve ADC0 Data	ADC0H:ADC0L	none
x0100000b	Retrieve ADC1 Data	ADC1H:ADC1L	none
x0110000b	Retrieve ADC0 and ADC1 Data	ADC0H:ADC0L	ADC1H:ADC1L
x10x0000b	Retrieve Differential Data	ADC0H:ADC0L (differential result from both ADCs)	none
x11x0000b	Retrieve Differential and ADC1 Data	ADC0H:ADC0L (differential result from both ADCs)	ADC1H:ADC1L

6.3. XRAM Addressing and Setup

The DMA Interface can be configured to access either on-chip or off-chip XRAM. Any writes to on-chip XRAM by the DMA Control Logic occur when the processor core is not accessing the on-chip XRAM. This ensures that the DMA will not interfere with processor instruction timing.

Off-chip XRAM access (only available on the C8051F060/2/4/6) is controlled by the DMA0HLT bit in DMA0CF (DMA Configuration Register, Figure 6.5). The DMA will have full access to off-chip XRAM when this bit is ‘0’, and the processor core will have full access to off-chip XRAM when this bit is ‘1’. The DMA0HLT bit should be controlled in software when both the processor core and the DMA Interface require access to off-chip XRAM data space. Before setting DMA0HLT to ‘1’, the software should check the DMA0XBY bit to ensure that the DMA is not currently accessing off-chip XRAM. The processor core cannot access off-chip XRAM while DMA0HLT is ‘0’. The processor will continue as though it was able to perform the desired memory access, but the data will not be written to or read from off-chip XRAM. When the processor core is finished accessing off-chip XRAM, DMA0HLT should be set back to ‘0’in software to return control to the DMA Interface. The DMA Control Logic will wait until DMA0HLT is ‘0’ before writing data to off-chip XRAM. If new data becomes available to the DMA Interface before the previous data has been written, an overflow condition will occur, and the new data word may be lost.

The Data Address Pointer Registers (DMA0DSH and DMA0DSL) contain the 16-bit XRAM address location where the DMA interface will write data. When the DMA is initially enabled, the DMA Data Address

Figure 7.8. ADC2H: ADC2 Data Word MSB Register

SFR Page: 2
SFR Address: 0xBF

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	

Bits7-0: ADC2 Data Word High-Order Bits.
For AD2LJST = 0: Bits 7-2 are the sign extension of Bit 1. Bits 1-0 are the upper 2 bits of the 10-bit ADC2 Data Word.
For AD2LJST = 1: Bits 7-0 are the most-significant bits of the 10-bit ADC2 Data Word.

Figure 7.9. ADC2L: ADC2 Data Word LSB Register

SFR Page: 2
SFR Address: 0xBE

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	

Bits7-0: ADC2 Data Word Low-Order Bits.
For AD2LJST = 0: Bits 7-0 are the lower 8 bits of the 10-bit Data Word.
For AD2LJST = 1: Bits 7-6 are the lower 2 bits of the 10-bit Data Word. Bits 5-0 will always read ‘0’.

Figure 7.10. ADC2CN: ADC2 Control Register

SFR Page: 2
SFR Address: 0xE8 (bit addressable)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
AD2EN	AD2TM	AD2INT	AD2BUSY	AD2CM1	AD2CM0	AD2WINT	AD2LJST	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

Bit 7: AD2EN: ADC2 Enable Bit.
0: ADC2 Disabled. ADC2 is in low-power shutdown.
1: ADC2 Enabled. ADC2 is active and ready for data conversions.

Bit6: AD2TM: ADC2 Track Mode Bit.
0: Normal Track Mode: When ADC2 is enabled, tracking is continuous unless a conversion is in progress.
1: Low-power Track Mode: Tracking Defined by AD2CM2-0 bits (see below).

Bit5: AD2INT: ADC2 Conversion Complete Interrupt Flag.
0: ADC2 has not completed a data conversion since the last time AD2INT was cleared.
1: ADC2 has completed a data conversion.

Bit 4: AD2BUSY: ADC2 Busy Bit.
Read:
0: ADC2 conversion is complete or a conversion is not currently in progress. AD2INT is set to logic 1 on the falling edge of AD2BUSY.
1: ADC2 conversion is in progress.
Write:
0: No Effect.
1: Initiates ADC2 Conversion if AD2CM2-0 = 000b

Bits 3-2: AD2CM1-0: ADC2 Start of Conversion Mode Select.
When AD2TM = 0:
00: ADC2 conversion initiated on every write of ‘1’ to AD2BUSY.
01: ADC2 conversion initiated on overflow of Timer 3.
10: ADC2 conversion initiated on rising edge of external CNVSTR2 pin.
11: ADC2 conversion initiated on overflow of Timer 2.
When AD2TM = 1:
00: Tracking initiated on write of ‘1’ to AD2BUSY and lasts 3 SAR clocks, followed by conversion.
01: Tracking initiated on overflow of Timer 3 and lasts 3 SAR clocks, followed by conversion.
10: ADC2 tracks only when CNVSTR2 input is logic low; conversion starts on rising CNVSTR2 edge.
11: Tracking initiated on overflow of Timer 2 and lasts 3 SAR clocks, followed by conversion.

Bit 1: AD2WINT: ADC2 Window Compare Interrupt Flag.
0: ADC2 Window Comparison Data match has not occurred since this flag was last cleared.
1: ADC2 Window Comparison Data match has occurred.

Bit 0: AD2LJST: ADC2 Left Justify Select.
0: Data in ADC2H:ADC2L registers are right-justified.
1: Data in ADC2H:ADC2L registers are left-justified.

7.3. Programmable Window Detector

The ADC Programmable Window Detector continuously compares the ADC2 output registers 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 (ADC2GTH, ADC2GTL) and Less-Than (ADC2LTH, ADC2LTL) registers hold the comparison values. 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 ADC2 Less-Than and ADC2 Greater-Than registers.

Figure 7.11. ADC2GTH: ADC2 Greater-Than Data High Byte Register

SFR Page: 2
SFR Address: 0xC5

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	

Bits7-0: High byte of ADC2 Greater-Than Data Word.

Figure 7.12. ADC2GTL: ADC2 Greater-Than Data Low Byte Register

SFR Page: 2
SFR Address: 0xC4

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	

Bits7-0: Low byte of ADC2 Greater-Than Data Word.

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 13.6 below.

Figure 13.6. SFR Page Stack Upon PCA Interrupt Occurring During an ADC2 ISR

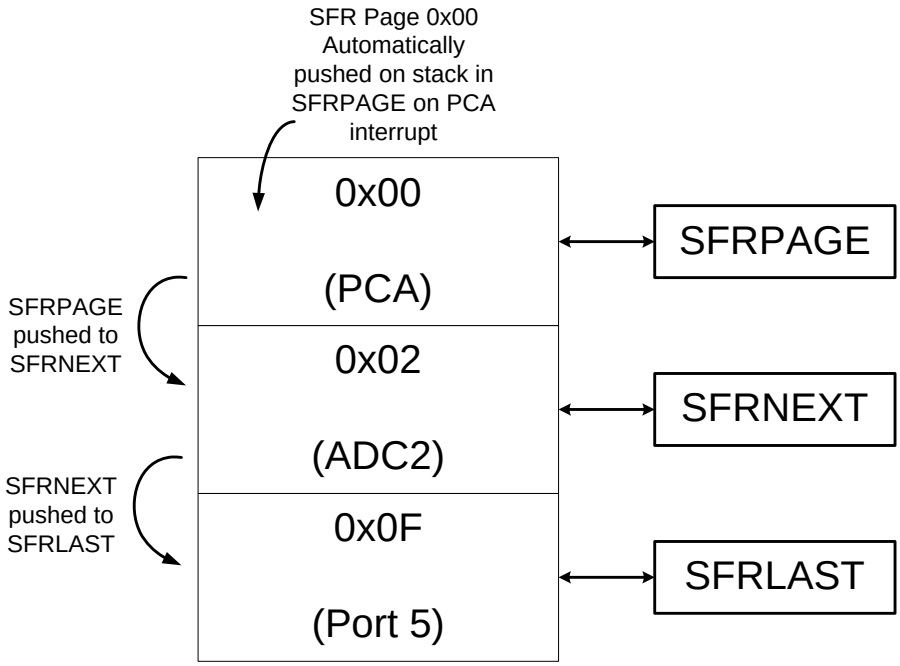


Table 13.3. Special Function Registers (Continued)

SFRs are listed in alphabetical order. All undefined SFR locations are reserved.

Register	Address	SFR Page	Description	Page No.
TMR4L	0xCC	2	Timer/Counter 4 Low	page 301
WDTCN	0xFF	All Pages	Watchdog Timer Control	page 167
XBR0	0xE1	F	Port I/O Crossbar Control 0	page 210
XBR1	0xE2	F	Port I/O Crossbar Control 1	page 211
XBR2	0xE3	F	Port I/O Crossbar Control 2	page 212
XBR3	0xE4	F	Port I/O Crossbar Control 3	page 213

*1 Refers to a register in the C8051F060/2/4/6 only.

*2 Refers to a register in the C8051F060/2 only.

*3 Refers to a register in the C8051F061/3 only.

*4 Refers to a register in the C8051F060/1/2/3 only.

*5 Refers to a register in the C8051F064/5/6/7 only.

Figure 13.16. PSW: Program Status Word

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
CY	AC	F0	RS1	RS0	OV	F1	PARITY	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Bit Addressable

SFR Address: 0xD0
SFR Page: All Pages

Bit7: CY: Carry Flag.
This bit is set when the last arithmetic operation resulted in a carry (addition) or a borrow (subtraction). It is cleared to 0 by all other arithmetic operations.

Bit6: AC: Auxiliary Carry Flag.
This bit is set when the last arithmetic operation resulted in a carry into (addition) or a borrow from (subtraction) the high order nibble. It is cleared to 0 by all other arithmetic operations.

Bit5: F0: User Flag 0.
This is a bit-addressable, general purpose flag for use under software control.

Bits4-3: RS1-RS0: Register Bank Select.
These bits select which register bank is used during register accesses.

RS1	RS0	Register Bank	Address
0	0	0	0x00 - 0x07
0	1	1	0x08 - 0x0F
1	0	2	0x10 - 0x17
1	1	3	0x18 - 0x1F

Bit2: OV: Overflow Flag.
This bit is set to 1 under the following circumstances:
- An ADD, ADDC, or SUBB instruction causes a sign-change overflow.
- A MUL instruction results in an overflow (result is greater than 255).
- A DIV instruction causes a divide-by-zero condition.

The OV bit is cleared to 0 by the ADD, ADDC, SUBB, MUL, and DIV instructions in all other cases.

Bit1: F1: User Flag 1.
This is a bit-addressable, general purpose flag for use under software control.

Bit0: PARITY: Parity Flag.
This bit is set to 1 if the sum of the eight bits in the accumulator is odd and cleared if the sum is even.

Figure 13.21. EIE1: Extended Interrupt Enable 1

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
EADC0	CP2IE	CP1IE	CP0IE	EPCA0	EWADC0	ESMB0	ESPI0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

SFR Address: 0xE6
SFR Page: All Pages

Bit7: EADC0: Enable ADC0 End of Conversion Interrupt.
This bit sets the masking of the ADC0 End of Conversion Interrupt.
0: Disable ADC0 Conversion Interrupt.
1: Enable interrupt requests generated by the ADC1 Conversion Interrupt.

Bit6: CP2IE: Enable Comparator (CP2) Interrupt.
This bit sets the masking of the CP2 interrupt.
0: Disable CP2 interrupts.
1: Enable interrupt requests generated by the CP2IF flag.

Bit6: CP1IE: Enable Comparator (CP1) Interrupt.
This bit sets the masking of the CP1 interrupt.
0: Disable CP1 interrupts.
1: Enable interrupt requests generated by the CP1IF flag.

Bit6: CP0IE: Enable Comparator (CP0) Interrupt.
This bit sets the masking of the CP0 interrupt.
0: Disable CP0 interrupts.
1: Enable interrupt requests generated by the CP0IF flag.

Bit3: 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.

Bit2: EWADC0: Enable Window Comparison ADC0 Interrupt.
This bit sets the masking of ADC0 Window Comparison interrupt.
0: Disable ADC0 Window Comparison Interrupt.
1: Enable Interrupt requests generated by ADC0 Window Comparisons.

Bit1: ESMB0: Enable System Management Bus (SMBus0) Interrupt.
This bit sets the masking of the SMBus interrupt.
0: Disable all SMBus interrupts.
1: Enable interrupt requests generated by the SI flag.

Bit0: ESPI0: Enable Serial Peripheral Interface (SPI0) Interrupt.
This bit sets the masking of SPI0 interrupt.
0: Disable all SPI0 interrupts.
1: Enable Interrupt requests generated by the SPI0 flag.

16. Flash Memory

The C8051F060/1/2/3/4/5/6/7 devices include on-chip, reprogrammable Flash memory for program code and non-volatile data storage. The C8051F060/1/2/3/4/5 include 64 k + 128 bytes of Flash, and the C8051F066/7 include 32 k + 128 bytes of Flash. The Flash memory can be programmed in-system, a single byte at a time, through the JTAG interface or by software using the MOVX write instructions. Once cleared to logic 0, a Flash bit must be erased to set it back to logic 1. The bytes would typically be erased (set to 0xFF) before being reprogrammed. Flash write and erase operations are automatically timed by hardware for proper execution; data polling to determine the end of the write/erase operation is not required. The CPU is stalled during write/erase operations while the device peripherals remain active. Interrupts that occur during Flash write/erase operations are held, and are then serviced in their priority order once the Flash operation has completed. Refer to Table 16.1 for the electrical characteristics of the Flash memory.

16.1. Programming The Flash Memory

The simplest means of programming the Flash memory is through the JTAG interface using programming tools provided by Silicon Labs or a third party vendor. This is the only means for programming a non-initialized device. For details on the JTAG commands to program Flash memory, see Section “26. JTAG (IEEE 1149.1)” on page 317.

The Flash memory can be programmed from software using the MOVX write instruction with the address and data byte to be programmed provided as normal operands. Before writing to Flash memory using MOVX, Flash write operations must be enabled by setting the PSWE Program Store Write Enable bit (PSCTL.0) to logic 1. This directs the MOVX writes to Flash memory instead of to XRAM, which is the default target. The PSWE bit remains set until cleared by software. To avoid errant Flash writes, it is recommended that interrupts be disabled while the PSWE bit is logic 1.

Flash memory is read using the MOVC instruction. MOVX reads are always directed to XRAM, regardless of the state of PSWE.

NOTE: To ensure the integrity of Flash memory contents, it is strongly recommended that the on-chip VDD monitor be enabled by connecting the VDD monitor enable pin (MONEN) to VDD and setting the PORSF bit in the RSTSRC register to ‘1’ in any system that writes and/or erases Flash memory from software. See “Reset Sources” on page 163 for more information.

A write to Flash memory can clear bits but cannot set them; only an erase operation can set bits in Flash. **A byte location to be programmed must be erased before a new value can be written**. The Flash memory is organized in 512-byte pages. The erase operation applies to an entire page (setting all bytes in the page to 0xFF). The following steps illustrate the algorithm for programming Flash from user software.

Step 1. Disable interrupts.
Step 2. Set FLWE (FLSCL.0) to enable Flash writes/erases via user software.
Step 3. Set PSEE (PSCTL.1) to enable Flash erases.
Step 4. Set PSWE (PSCTL.0) to redirect MOVX commands to write to Flash.
Step 5. Use the MOVX command to write a data byte to any location within the 512-byte page to be erased.
Step 6. Clear PSEE to disable Flash erases
Step 7. Use the MOVX command to write a data byte to the desired byte location within the erased 512-byte page. Repeat this step until all desired bytes are written (within the target page).

16.3.1. Summary of Flash Security Options

There are three Flash access methods supported on the C8051F060/1/2/3/4/5/6/7; 1) Accessing Flash through the JTAG debug interface, 2) Accessing Flash from firmware residing below the Flash Access Limit, and 3) Accessing Flash from firmware residing at or above the Flash Access Limit.

Accessing Flash through the JTAG debug interface:

1. The Read and Write/Erase Lock bytes (security bytes) provide security for Flash access through the JTAG interface.
2. Any unlocked page may be read from, written to, or erased.
3. Locked pages cannot be read from, written to, or erased.
4. Reading the security bytes is always permitted.
5. Locking additional pages by writing to the security bytes is always permitted.
6. If the page containing the security bytes is **unlocked**, it can be directly erased. **Doing so will reset the security bytes and unlock all pages of Flash.**
7. If the page containing the security bytes is **locked**, it cannot be directly erased. **To unlock the page containing the security bytes, a full JTAG device erase is required.** A full JTAG device erase will erase all Flash pages, including the page containing the security bytes and the security bytes themselves.
8. The Reserved Area cannot be read from, written to, or erased at any time.

Accessing Flash from firmware residing below the Flash Access Limit:

1. The Read and Write/Erase Lock bytes (security bytes) do not restrict Flash access from user firmware.
2. Any page of Flash except the page containing the security bytes may be read from, written to, or erased.
3. **The page containing the security bytes cannot be erased.** Unlocking pages of Flash can only be performed via the JTAG interface.
4. The page containing the security bytes may be read from or written to. Pages of Flash can be locked from JTAG access by writing to the security bytes.
5. The Reserved Area cannot be read from, written to, or erased at any time.

Accessing Flash from firmware residing at or above the Flash Access Limit:

1. The Read and Write/Erase Lock bytes (security bytes) do not restrict Flash access from user firmware.
2. Any page of Flash at or above the Flash Access Limit except the page containing the security bytes may be read from, written to, or erased.
3. Any page of Flash below the Flash Access Limit cannot be read from, written to, or erased.
4. Code branches to locations below the Flash Access Limit are permitted.
5. **The page containing the security bytes cannot be erased.** Unlocking pages of Flash can only be performed via the JTAG interface.
6. The page containing the security bytes may be read from or written to. Pages of Flash can be locked from JTAG access by writing to the security bytes.
7. The Reserved Area cannot be read from, written to, or erased at any time.

17.5.3. Split Mode with Bank Select

When EMI0CF.[3:2] are set to ‘10’, the XRAM memory map is split into two areas, on-chip space and off-chip space.

- Effective addresses below the 4 kB boundary will access on-chip XRAM space.
- Effective addresses beyond the 4 kB boundary will access off-chip space.
- 8-bit MOVX operations use the contents of EMI0CN to determine whether the memory access is on-chip or off-chip. The upper 8-bits of the Address Bus A[15:8] are determined by EMI0CN, and the lower 8-bits of the Address Bus A[7:0] are determined by R0 or R1. All 16-bits of the Address Bus A[15:0] are driven in “Bank Select” mode.
- 16-bit MOVX operations use the contents of DPTR to determine whether the memory access is on-chip or off-chip, and the full 16-bits of the Address Bus A[15:0] are driven during the off-chip transaction.

17.5.4. External Only

When EMI0CF[3:2] are set to ‘11’, all MOVX operations are directed to off-chip space. On-chip XRAM is not visible to the CPU. This mode is useful for accessing off-chip memory located between 0x0000 and the 4 kB boundary.

- 8-bit MOVX operations ignore the contents of EMI0CN. The upper Address bits A[15:8] are not driven (identical behavior to an off-chip access in “Split Mode without Bank Select” described above). This allows the user to manipulate the upper address bits at will by setting the Port state directly. The lower 8-bits of the effective address A[7:0] are determined by the contents of R0 or R1.
- 16-bit MOVX operations use the contents of DPTR to determine the effective address A[15:0]. The full 16-bits of the Address Bus A[15:0] are driven during the off-chip transaction.

28 SMBus0 states, along with their corresponding status codes, are given in Table 1.1.

Figure 20.12. SMB0STA: SMBus0 Status Register

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
STA7	STA6	STA5	STA4	STA3	STA2	STA1	STA0	11111000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

SFR Address: 0xC1
SFR Page: 0

Bits7-3: STA7-STA3: SMBus0 Status Code.
These bits contain the SMBus0 Status Code. There are 28 possible status codes; each status code corresponds to a single SMBus state. A valid status code is present in SMB0STA when the SI flag (SMB0CN.3) is set to logic 1. The content of SMB0STA is not defined when the SI flag is logic 0. Writing to the SMB0STA register at any time will yield indeterminate results.

Bits2-0: STA2-STA0: The three least significant bits of SMB0STA are always read as logic 0 when the SI flag is logic 1.

Figure 21.10. SPI0CKR: SPI0 Clock Rate Register

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
SCR7	SCR6	SCR5	SCR4	SCR3	SCR2	SCR1	SCR0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

SFR Address: 0x9D
SFR Page: 0

Bits 7-0: SCR7-SCR0: SPI0 Clock Rate.
These bits determine the frequency of the SCK output when the SPI0 module is configured for master mode operation. The SCK clock frequency is a divided version of the system clock, and is given in the following equation, where *SYSCLK* is the system clock frequency and *SPI0CKR* is the 8-bit value held in the SPI0CKR register.

$$f_{SCK} = \frac{SYSCLK}{2 \times (SPI0CKR + 1)}$$

for 0 <= SPI0CKR <= 255

Example: If SYSCLK = 2 MHz and SPI0CKR = 0x04,

$$f_{SCK} = \frac{2000000}{2 \times (4 + 1)}$$

$$f_{SCK} = 200kHz$$

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 205 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 Figure 24.6).

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 “13.3.5. Interrupt Register Descriptions” on page 154), facilitating pulse width measurements.

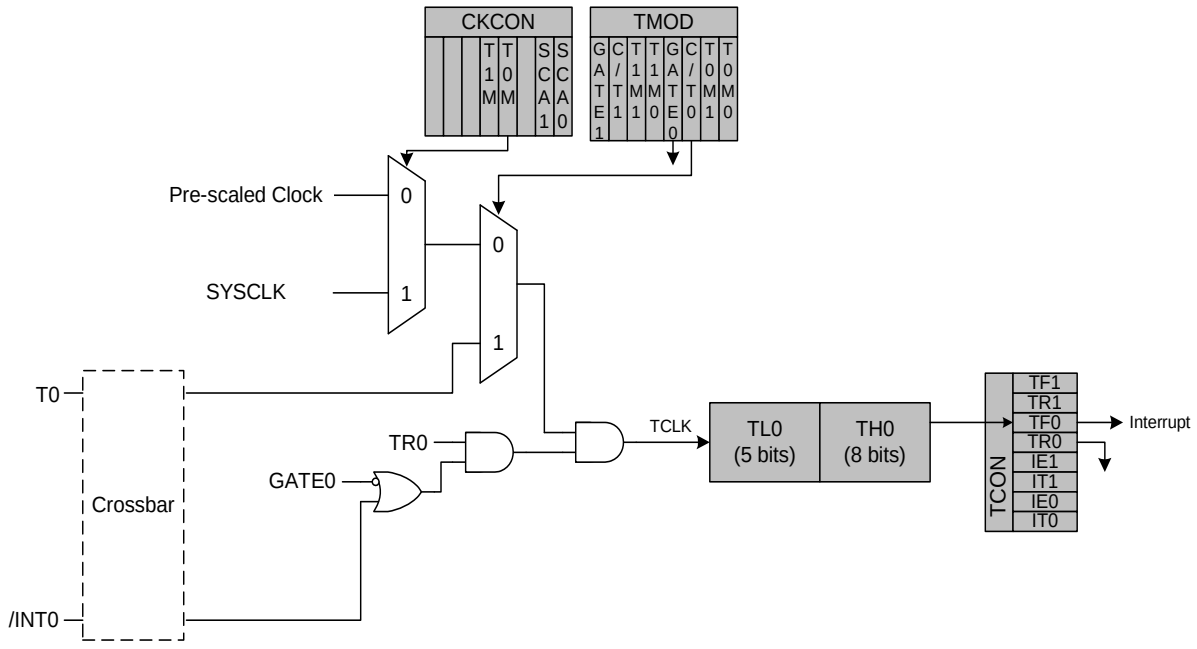
TR0	GATE0	/INT0	Counter/Timer
0	X	X	Disabled
1	0	X	Enabled
1	1	0	Disabled
1	1	1	Enabled

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 24.1. T0 Mode 0 Block Diagram



24.2.3. Auto-Reload Mode

In Auto-Reload Mode, the counter/timer can be configured to count up or down and cause an interrupt/flag to occur upon an overflow/underflow event. When counting up, the counter/timer will set its overflow/underflow flag (TFn) and cause an interrupt (if enabled) upon overflow/underflow, and the values in the Reload/Capture Registers (RCAPnH and RCAPnL) are loaded into the timer and the timer is restarted. When the Timer External Enable Bit (EXENn) bit is set to '1' and the Decrement Enable Bit (DCENn) is '0', a falling edge ('1'-to-'0' transition) on the TnEX pin (configured as an input in the digital crossbar) will cause a timer reload (in addition to timer overflows causing auto-reloads). When DCENn is set to '1', the state of the TnEX pin controls whether the counter/timer counts *up* (increments) or *down* (decrements), and will not cause an auto-reload or interrupt event. See Section 24.2.1 for information concerning configuration of a timer to count down.

When counting down, the counter/timer will set its overflow/underflow flag (TFn) and cause an interrupt (if enabled) when the value in the timer (TMRnH and TMRnL registers) matches the 16-bit value in the Reload/Capture Registers (RCAPnH and RCAPnL). This is considered an underflow event, and will cause the timer to load the value 0xFFFF. The timer is automatically restarted when an underflow occurs.

Counter/Timer with Auto-Reload mode is selected by clearing the CP/RLn bit. Setting TRn to logic 1 enables and starts the timer.

In Auto-Reload Mode, the External Flag (EXFn) toggles upon every overflow or underflow and does not cause an interrupt. The EXFn flag can be thought of as the most significant bit (MSB) of a 17-bit counter.

.

Figure 24.12. T2, 3, and 4 Auto-reload Mode Block Diagram