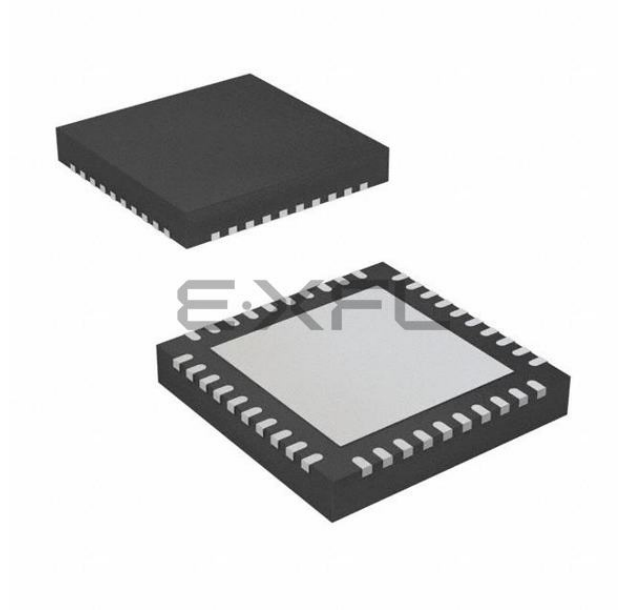

Welcome to E-XFL.COM

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	I²C, SPI, UART/USART
Peripherals	DMA, LCD, POR, PWM, WDT
Number of I/O	34
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	8.25K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.8V
Data Converters	A/D 16x10b/12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	40-VFQFN Exposed Pad
Supplier Device Package	40-QFN (6x6)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f967-a-gmr

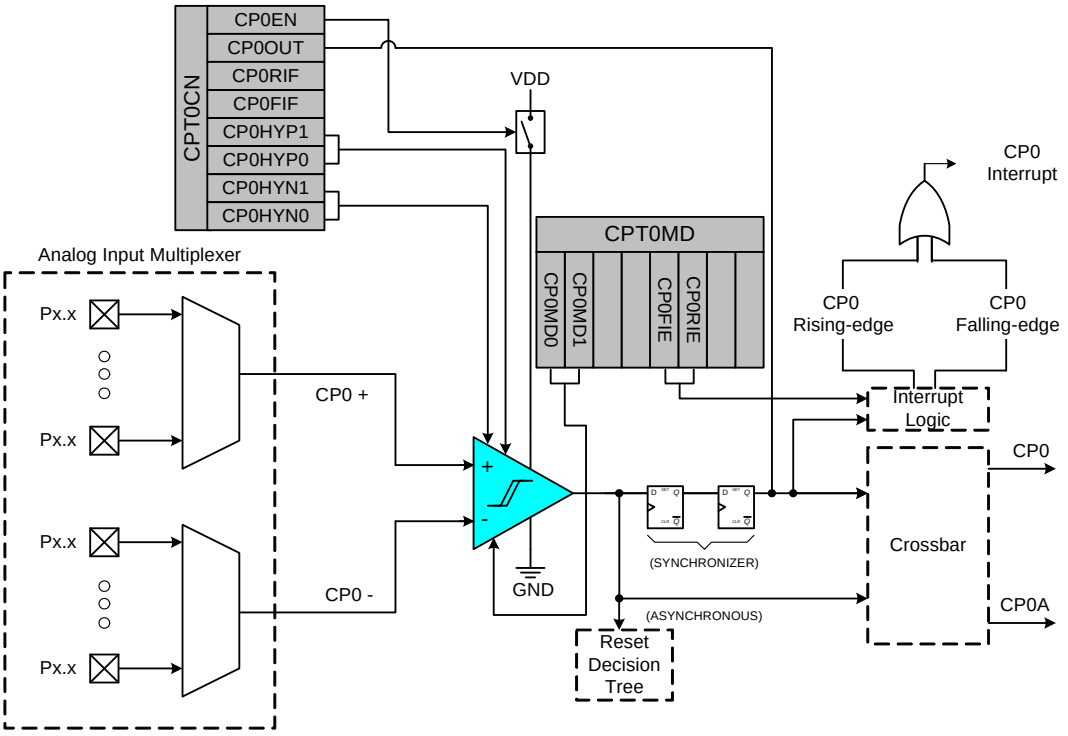

Figure 1.15. Comparator 0 Functional Block Diagram

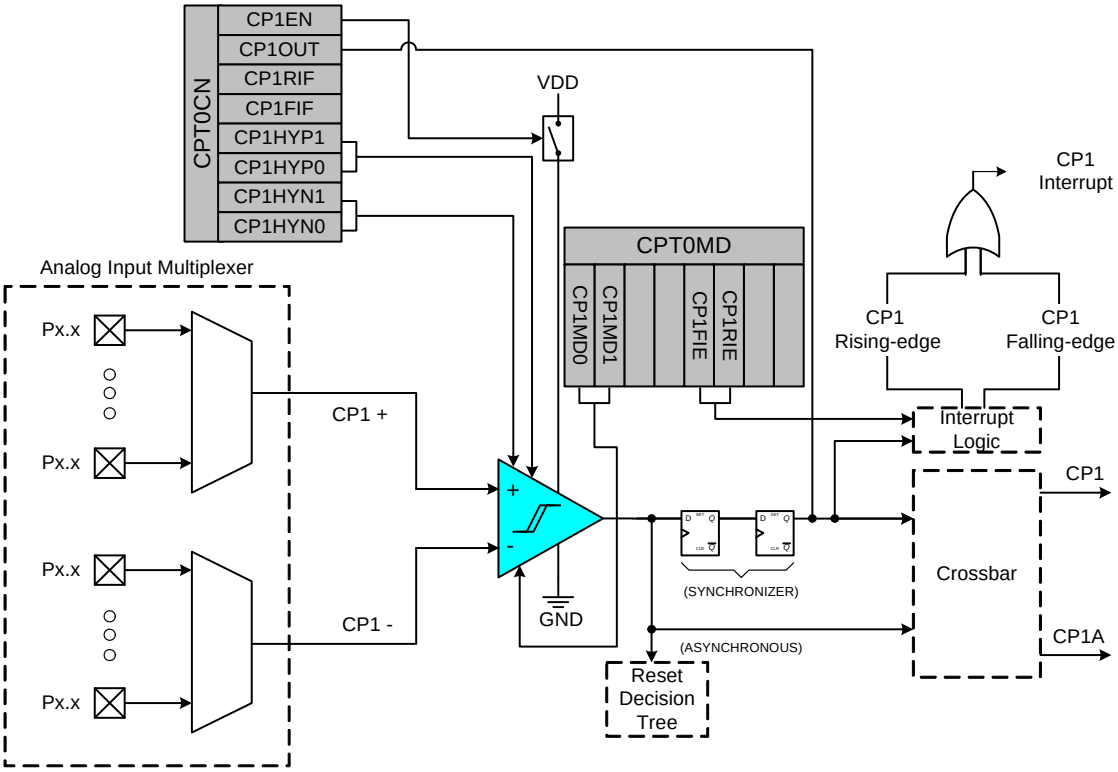

Figure 1.16. Comparator 1 Functional Block Diagram

Table 3.1. Pin Definitions for the C8051F96x (Continued)

Name	Pin Numbers			Type	Description
	DQFN76	TQFP80	QFN40		
P4.2 LCD10	B2	5		D I/O or A In A O	Port 4.2. See Port I/O Section for a complete description. LCD Segment Pin 10
P4.3 LCD11	B1	3		D I/O or A In A O	Port 4.3. See Port I/O Section for a complete description. LCD Segment Pin 11
P4.4 LCD12	D1	80		D I/O or A In A O	Port 4.4. See Port I/O Section for a complete description. LCD Segment Pin 12
P4.5 LCD13	B28	77		D I/O or A In A O	Port 4.5. See Port I/O Section for a complete description. LCD Segment Pin 13
P4.6 LCD14	B27	75		D I/O or A In A O	Port 4.6. See Port I/O Section for a complete description. LCD Segment Pin 14
P4.7 LCD15	B26	73		D I/O or A In A O	Port 4.7. See Port I/O Section for a complete description. LCD Segment Pin 15
P5.0 LCD16	B25	71		D I/O or A In A O	Port 5.0. See Port I/O Section for a complete description. LCD Segment Pin 16
P5.1 LCD17	B24	69		D I/O or A In A O	Port 5.1. See Port I/O Section for a complete description. LCD Segment Pin 17

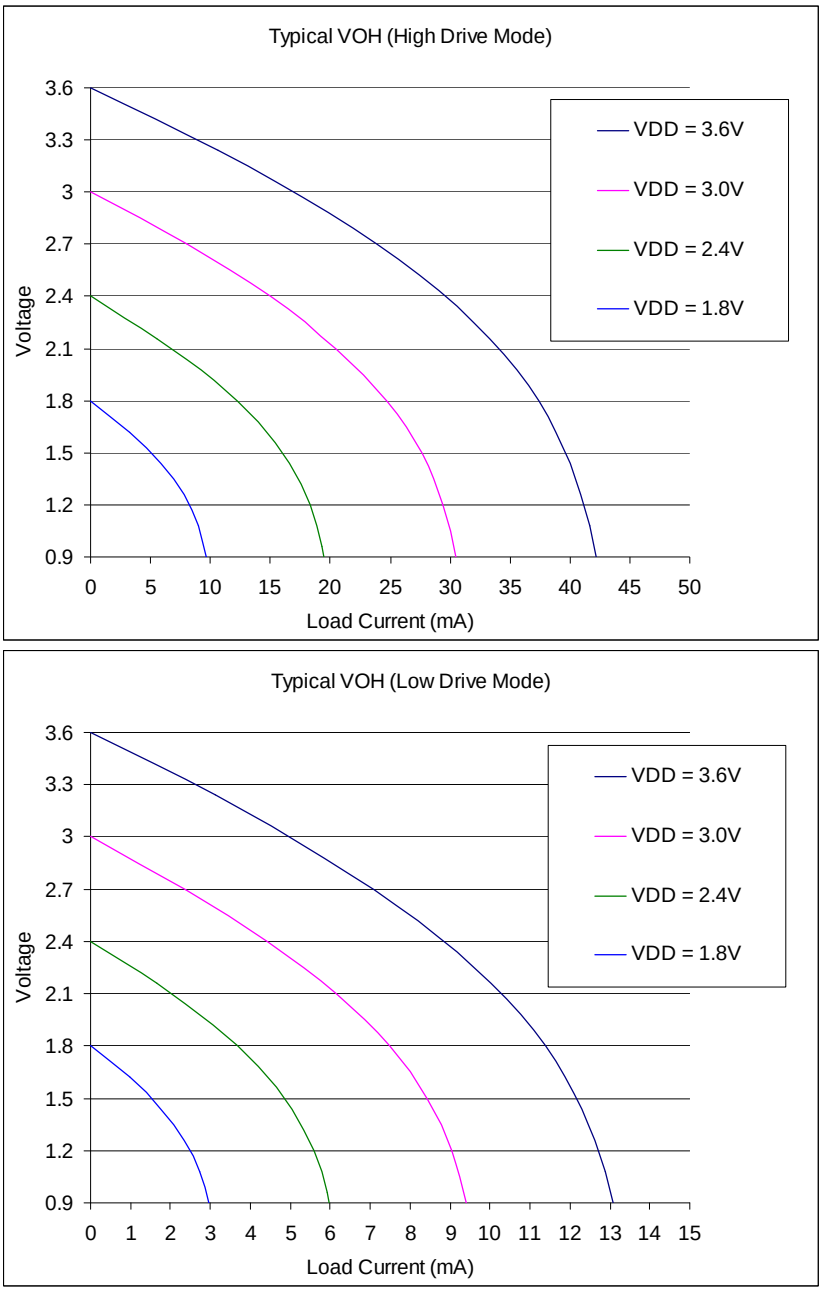

Figure 4.2. Typical VOH Curves, 1.8–3.6 V

SFR Definition 10.1. EMI0CN: External Memory Interface Control

Bit	7	6	5	4	3	2	1	0
Name	PGSEL[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Page = 0x0; SFR Address = 0xAA

Bit	Name	Function
7:0	PGSEL[7:0]	**XRAM Page Select Bits.** The XRAM Page Select Bits provide 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

11. Direct Memory Access (DMA0)

An on-chip direct memory access (DMA0) is included on the C8051F96x devices. The DMA0 subsystem allows autonomous variable-length data transfers between XRAM and peripheral SFR registers without CPU intervention. During DMA0 operation, the CPU is free to perform some other tasks. In order to save total system power consumption, the CPU and flash can be powered down. DMA0 improves the system performance and efficiency with high data throughput peripherals.

DMA0 contains seven independent channels, common control registers, and a DMA0 Engine (see Figure 11.1). Each channel includes a register that assigns a peripheral to the channel, a channel control register, and a set of SFRs that include XRAM address information and SFR address information used by the channel during a data transfer. The DMA0 architecture is described in detail in Section 11.1.

The DMA0 in C8051F96x devices supports four peripherals: AES0, ENC0, CRC1, and SPI1. Peripherals with DMA0 capability should be configured to work with the DMA0 through their own registers. The DMA0 provides up to seven channels, and each channel can be configured for one of nine possible data transfer functions:

- XRAM to ENC0L/M/H
- ENC0L/M/H sfrs to XRAM
- XRAM to CRC1IN sfr
- XRAM to SPI1DAT sfr
- SPI1DAT sfr to XRAM
- XRAM to AES0KIN sfr
- XRAM to AES0BIN sfr
- XRAM to AES0XIN sfr
- AES0YOUT sfr to XRAM

The DMA0 subsystem signals the MCU through a set of interrupt service routine flags. Interrupts can be generated when the DMA0 transfers half of the data length or full data length on any channel.

13. DMA-Enabled Cyclic Redundancy Check Module (CRC1)

C8051F96x devices include a DMA-enabled cyclic redundancy check module (CRC1) that can perform a CRC of data using an arbitrary 16-bit polynomial. This peripheral can compute CRC results using direct DMA access to data in XRAM.

Using a DMA transfer provides much higher data throughput than using SFR access. Since the CPU can be in Idle mode while the CRC is calculated, CRC1 also provides substantial power savings. The CRC1 module is not restricted to a limited list of fixed polynomials. Instead, the user can specify any valid 16-bit polynomial.

CRC1 accepts a stream of 8-bit data written to the CRC1IN register. A DMA transfer can be used to autonomously transfer data from XRAM to the CRC1IN SFR. The CRC1 module may also be used with SFR access by writing directly to the CRC1IN SFR. After each byte is written, the CRC resultant is updated on the CRC1OUTH:L SFRs. After writing all data bytes, the final CRC results are available from the CRC1OUTH:L registers. The final results may be flipped or inverted using the FLIP and INV bits in the CRC1CN SFR. The initial seed value can be reset to 0x0000 or seeded with 0xFFFF.

13.1. Polynomial Specification

The arbitrary polynomial should be written to the CRC1POLH:L SFRs before writing data to the CRCIN SFR.

A valid 16-bit CRC polynomial must have an x^{16} term and an x^0 term. Theoretically, a 16-bit polynomial might have 17 terms total. However, the polynomial SFR is only 16-bits wide. The convention used is to omit the x^{16} term. The polynomial should be written in big endian bit order. The most significant bit corresponds to the highest order term. Thus, the most significant bit in the CRC1POLH SFR represents the x^{15} term, and the least significant bit in the CRC1POLL SFR represents the x^0 term. The least significant bit of CRC1POLL should always be set to one. The CRC results are undefined if this bit is cleared to a zero.

Figure 13.1 depicts the polynomial representation for the CRC-16-CCIT polynomial $x^{16} + x^{12} + x^5 + 1$, or 0x1021.

CRC1POLH:L = 0x1021

	CRC1POLH								CRC1POLL							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	1
	↓			↓							↓					↓
	$x^{16}+$			$x^{12}+$							x^5+					1

Figure 13.1. Polynomial Representation

14.2.1. Key Inversion using DMA

Normally, the AES block is used with the DMA. This provides the best performance and lowest power consumption. Code examples are provided in 8051 compiler independent C code using the DMA. It is highly recommended to use the code examples. The steps are listed here for completeness.

Steps to generate the Decryption Key from Encryption Key

- Prepare encryption key and dummy data in xram.
- Reset AES module by clearing bit 3 of AES0BCFG.
- Disable the first three DMA channels by clearing bits 0 to 2 in the DMA0EN sfr.
- Configure the first DMA channel for the AES0KIN sfr.
  - Select the first DMA channel by writing 0x00 to the DMA0SEL sfr.
  - Configure the first DMA channel to move xram to AES0KIN sfr by writing 0x05 to the DMA0NCF sfr.
  - Clear DMA0NMD to disable wrapping.
  - Write the xram address of the encryption key to the DMA0NBAH and DMA0NBAL sfrs.
  - Write the key length in bytes to DMA0NSZL sfr.
  - Clear DMA0NSZH
  - Clear DMA0NAOH and DMA0NAOL.
- Configure the second DMA channel for the AES0BIN sfr.
  - Select the second DMA channel by writing 0x01 to the DMA0SEL sfr.
  - Configure the second DMA channel to move xram to AES0BIN sfr by writing 0x06 to the DMA0NCF sfr.
  - Clear DMA0NMD to disable wrapping.
  - Write the xram address of dummy data to the DMA0NBAH and DMA0NBAL sfrs.
  - Write 0x10 (16) to the DMA0NSZL sfr.
  - Clear DMA0NSZH
  - Clear DMA0NAOH and DMA0NAOL
- Configure the third DMA channel for the AES0YOUT sfr.
  - Select the third DMA channel by writing 0x02 to the DMA0SEL sfr.
  - Configure the third DMA channel to move the contents of the AES0YOUT sfr to xram by writing 0x08 to the DMA0NCF sfr.
  - Enable transfer complete interrupt by setting bit 7 of DMA0NCF sfr.
  - Clear DMA0NMD to disable wrapping.
  - Write the xram address for the decryption key to the DMA0NBAH and DMA0NBAL sfrs.
  - Write the key length in bytes to DMA0NSZL sfr.
  - Clear DMA0NSZH.
  - Clear DMA0NAOH and DMA0NAOL.
- Clear first three DMA interrupts by clearing bits 0 to 2 in the DMA0INT sfr.
- Enable first three DMA channels setting bits 0 to 2 in the DMA0EN sfr
- Configure the AES Module data flow for inverse key generation by writing 0x04 to the AES0DCFG sfr.
- Write key size to bits 1 and 0 of the AES0BCFG.
- Configure the AES core for encryption by setting the bit 2 of AES0BCFG.
- Initiate the encryption operation by setting bit 3 of AES0BCFG.
- Wait on the DMA interrupt from DMA channel 2.
- Disable the AES Module by clearing bit 2 of AES0BCFG.
- Disable the DMA by writing 0x00 to DMA0EN.

SFR Definition 16.4. SFRLAST: SFR Last

Bit	7	6	5	4	3	2	1	0
Name	SFRLAST[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

;SFR Page = All Pages; SFR Address = 0x86

Bit	Name	Function
7:0	SFRLAST[7:0]	**SFR Page Stack Bits.** This is the value that will go to the SFRNEXT register upon a return from interrupt. Write: Sets the SFR Page in the last entry of the SFR Stack. This will cause the SFRNEXT SFR to have this SFR page value upon a return from interrupt. Read: Returns the value of the SFR page contained in the last entry of the SFR stack. SFR page context is retained upon interrupts/return from interrupts in a 3 byte SFR Page Stack: SFRPAGE is the first entry, SFRNEXT is the second, and SFRLAST is the third entry. The SFR stack bytes may be used alter the context in the SFR Page Stack, and will not cause the stack to “push” or “pop”. Only interrupts and return from interrupts cause pushes and pops of the SFR Page Stack.

SFR Definition 17.3. EIE1: Extended Interrupt Enable 1

Bit	7	6	5	4	3	2	1	0
Name	ET3	ECP1	ECP0	EPCA0	EADC0	EWADC0	ERTC0A	ESMB0
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Page = All Pages; SFR Address = 0xE6

Bit	Name	Function
7	ET3	**Enable Timer 3 Interrupt.** This bit sets the masking of the Timer 3 interrupt. 0: Disable Timer 3 interrupts. 1: Enable interrupt requests generated by the TF3L or TF3H flags.
6	ECP1	**Enable Comparator1 (CP1) Interrupt.** This bit sets the masking of the CP1 interrupt. 0: Disable CP1 interrupts. 1: Enable interrupt requests generated by the CP1RIF or CP1FIF flags.
5	ECP0	**Enable Comparator0 (CP0) Interrupt.** This bit sets the masking of the CP0 interrupt. 0: Disable CP0 interrupts. 1: Enable interrupt requests generated by the CP0RIF or CP0FIF flags.
4	EPCA0	**Enable Programmable Counter Array (PCA0) Interrupt.** This bit sets the masking of the PCA0 interrupts. 0: Disable all PCA0 interrupts. 1: Enable interrupt requests generated by PCA0.
3	EADC0	**Enable ADC0 Conversion Complete Interrupt.** This bit sets the masking of the ADC0 Conversion Complete interrupt. 0: Disable ADC0 Conversion Complete interrupt. 1: Enable interrupt requests generated by the AD0INT flag.
2	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 Compare flag (AD0WINT).
1	ERTC0A	**Enable SmaRTClock Alarm Interrupts.** This bit sets the masking of the SmaRTClock Alarm interrupt. 0: Disable SmaRTClock Alarm interrupts. 1: Enable interrupt requests generated by a SmaRTClock Alarm.
0	ESMB0	**Enable SMBus (SMB0) Interrupt.** This bit sets the masking of the SMB0 interrupt. 0: Disable all SMB0 interrupts. 1: Enable interrupt requests generated by SMB0.

18.1.4. Flash Write Optimization

The flash write procedure includes a block write option to optimize the time to perform consecutive byte writes. When block write is enabled by setting the CHBLKW bit (FLRBCN.0), writes to flash will occur in blocks of 4 bytes and require the same amount of time as a single byte write. This is performed by caching the bytes whose address end in 00b, 01b, and 10b that is written to flash and then committing all four bytes to flash when the byte with address 11b is written. When block writes are enabled, if the write to the byte with address 11b does not occur, the other three data bytes written is not committed to flash.

A write to flash memory can clear bits to logic 0 but cannot set them; only an erase operation can set bits to logic 1 in flash. **The Flash Block to be programmed should be erased before a new value is written.**

The recommended procedure for writing a 4-byte flash block is as follows:

1. Save current interrupt state and disable interrupts.
2. Set the CHBLKW bit (register FLRBCN).
3. Set the PSWE bit (register PSCTL).
4. Clear the PSEE bit (register PSCTL).
5. If writing to an address in Banks 1, 2, or 3, set the COBANK[1:0] bits (register PSBANK) for the appropriate bank
6. Write the first key code to FLKEY: 0xA5.
7. Write the second key code to FLKEY: 0xF1.
8. Using the MOVX instruction, write the first data byte to the desired location within the 1024-byte sector whose address ends in 00b.
9. Write the first key code to FLKEY: 0xA5.
10. Write the second key code to FLKEY: 0xF1.
11. Using the MOVX instruction, write the second data byte to the next higher flash address ending in 01b.
12. Write the first key code to FLKEY: 0xA5.
13. Write the second key code to FLKEY: 0xF1.
14. Using the MOVX instruction, write the third data byte to the next higher flash address ending in 10b.
15. Write the first key code to FLKEY: 0xA5.
16. Write the second key code to FLKEY: 0xF1.
17. Using the MOVX instruction, write the final data byte to the next higher flash address ending in 11b.
18. Clear the PSWE bit.
19. Clear the CHBLKW bit.
20. Restore previous interrupt state.

Steps 5–17 must be repeated for each flash block to be written.

Notes:

1. Flash security settings may prevent writes to some areas of flash, such as the reserved area. For a summary of flash security settings and restrictions affecting flash write operations, please see Section “18.3. Security Options” on page 247.
2. 8-bit MOVX instructions cannot be used to erase or write to flash memory at addresses higher than 0x00FF.

SFR Definition 26.3. LCD0CNTRST: LCD0 Contrast Adjustment

Bit	7	6	5	4	3	2	1	0
Name	Reserved	Reserved	Reserved	CNTRST				
Type	R/W	R/W	R/W	R/W				
Reset	0	0	0	0	0	0	0	0

SFR Page = 0x2; SFR Address = 0x9C

Bit	Name	Function
7:5	Reserved	Read = 000. Write = Must write 000.
4:0	CNTRST	**Contrast Setpoint.** Determines the setpoint for the VLCD voltage necessary to achieve the desired contrast. 00000: 1.90 00001: 1.96 00010: 2.02 00011: 2.08 00100: 2.13 00101: 2.19 00110: 2.25 00111: 2.31 01000: 2.37 01001: 2.43 01010: 2.49 01011: 2.55 01100: 2.60 01101: 2.66 01110: 2.72 01111: 2.78 10000: 2.84 10001: 2.90 10010: 2.96 10011: 3.02 10100: 3.07 10101: 3.13 10110: 3.19 10111: 3.25 11000: 3.31 11001: 3.37 11010: 3.43 11011: 3.49 11100: 3.54 11101: 3.60 11110: 3.66 11111: 3.72

SFR Definition 27.40. P7MDOUT: Port7 Output Mode

Bit	7	6	5	4	3	2	1	0
Name								P7MDOUT
Type	R	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

SFR Page = 0xF; SFR Address = 0xFC

Bit	Name	Function
7:1	Unused	Read = 0000000b; Write = Don't Care.
0	P7MDOUT	**Output Configuration Bits for P7.0.** These bits control the digital driver. 0: P7.0 Output is open-drain. 1: P7.0 Output is push-pull.

SFR Definition 27.41. P7DRV: Port7 Drive Strength

Bit	7	6	5	4	3	2	1	0
Name								P7DRV
Type	R	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

SFR Page = 0xF; SFR Address = 0xAB

Bit	Name	Function
7:1	Unused	Read = 0000000b; Write = Don't Care.
0	P7DRV	**Drive Strength Configuration Bits for P7.0.** Configures digital I/O Port cells to high or low output drive strength. 0: P7.0 Output has low output drive strength. 1: P7.0 Output has high output drive strength.

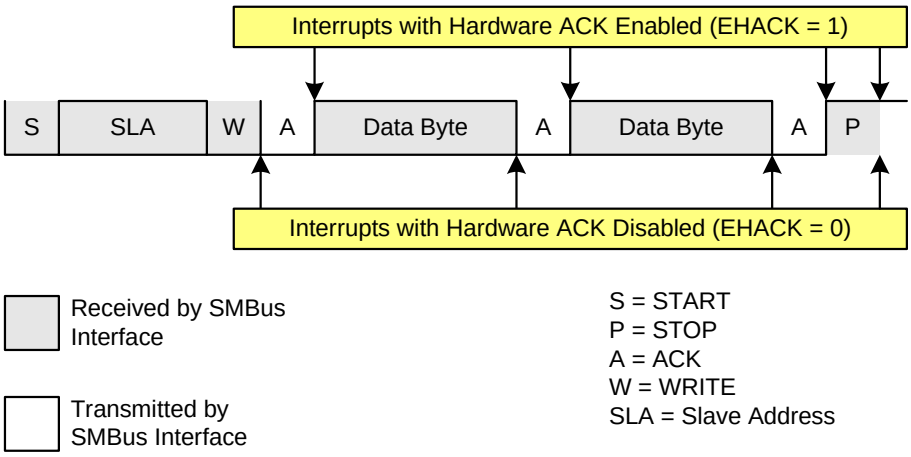

Figure 28.7. Typical Slave Write Sequence

28.5.4. Read Sequence (Slave)

During a read sequence, an SMBus master reads data from a slave device. The slave in this transfer will be a receiver during the address byte, and a transmitter during all data bytes. When slave events are enabled (INH = 0), the interface enters Slave Receiver Mode (to receive the slave address) when a START followed by a slave address and direction bit (READ in this case) is received. If hardware ACK generation is disabled, upon entering Slave Receiver Mode, an interrupt is generated and the ACKRQ bit is set. The software must respond to the received slave address with an ACK, or ignore the received slave address with a NACK. If hardware ACK generation is enabled, the hardware will apply the ACK for a slave address which matches the criteria set up by SMB0ADR and SMB0ADM. The interrupt will occur after the ACK cycle.

If the received slave address is ignored (by software or hardware), slave interrupts will be inhibited until the next START is detected. If the received slave address is acknowledged, zero or more data bytes are transmitted. If the received slave address is acknowledged, data should be written to SMB0DAT to be transmitted. The interface enters Slave Transmitter Mode, and transmits one or more bytes of data. After each byte is transmitted, the master sends an acknowledge bit; if the acknowledge bit is an ACK, SMB0DAT should be written with the next data byte. If the acknowledge bit is a NACK, SMB0DAT should not be written to before SI is cleared (an error condition may be generated if SMB0DAT is written following a received NACK while in Slave Transmitter Mode). The interface exits Slave Transmitter Mode after receiving a STOP. Note that the interface will switch to Slave Receiver Mode if SMB0DAT is not written following a Slave Transmitter interrupt. Figure 28.8 shows a typical slave read sequence. Two transmitted data bytes are shown, though any number of bytes may be transmitted. All of the “data byte transferred” interrupts occur **after** the ACK cycle in this mode, regardless of whether hardware ACK generation is enabled.

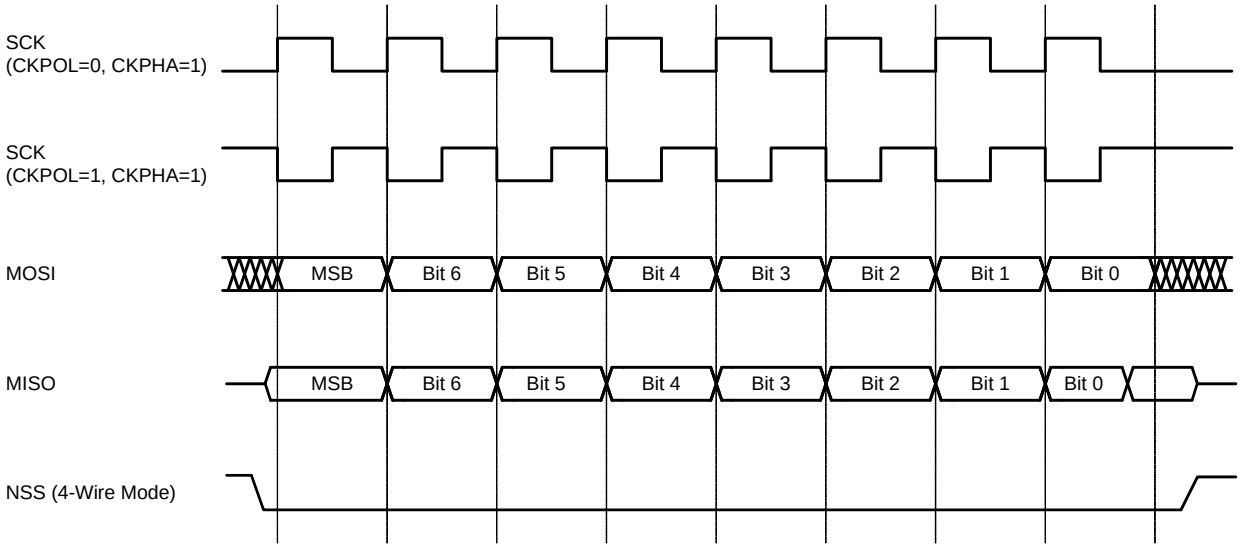

Figure 30.7. Slave Mode Data/Clock Timing (CKPHA = 1)

30.6. SPI Special Function Registers

SPI0 is accessed and controlled through four special function registers in the system controller: SPI0CN Control Register, SPI0DAT Data Register, SPI0CFG Configuration Register, and SPI0CKR Clock Rate Register. The four special function registers related to the operation of the SPI0 Bus are described in the following figures.

SFR Definition 30.1. SPI0CFG: SPI0 Configuration

Bit	7	6	5	4	3	2	1	0
Name	SPIBSY	MSTEN	CKPHA	CKPOL	SLVSEL	NSSIN	SRMT	RXBMT
Type	R	R/W	R/W	R/W	R	R	R	R
Reset	0	0	0	0	0	1	1	1

SFR Page = 0x0; SFR Address = 0xA1

Bit	Name	Function
7	SPIBSY	**SPI Busy.** This bit is set to logic 1 when a SPI transfer is in progress (master or slave mode).
6	MSTEN	**Master Mode Enable.** 0: Disable master mode. Operate in slave mode. 1: Enable master mode. Operate as a master.
5	CKPHA	**SPI0 Clock Phase.** 0: Data centered on first edge of SCK period.* 1: Data centered on second edge of SCK period.*
4	CKPOL	**SPI0 Clock Polarity.** 0: SCK line low in idle state. 1: SCK line high in idle state.
3	SLVSEL	**Slave Selected Flag.** This bit is set to logic 1 whenever the NSS pin is low indicating SPI0 is the selected slave. It is cleared to logic 0 when NSS is high (slave not selected). This bit does not indicate the instantaneous value at the NSS pin, but rather a de-glitched version of the pin input.
2	NSSIN	**NSS Instantaneous Pin Input.** This bit mimics the instantaneous value that is present on the NSS port pin at the time that the register is read. This input is not de-glitched.
1	SRMT	**Shift Register Empty (valid in slave mode only).** This bit will be set to logic 1 when all data has been transferred in/out of the shift register, and there is no new information available to read from the transmit buffer or write to the receive buffer. It returns to logic 0 when a data byte is transferred to the shift register from the transmit buffer or by a transition on SCK. SRMT = 1 when in Master Mode.
0	RXBMT	**Receive Buffer Empty (valid in slave mode only).** This bit will be set to logic 1 when the receive buffer has been read and contains no new information. If there is new information available in the receive buffer that has not been read, this bit will return to logic 0. RXBMT = 1 when in Master Mode.

Note: In slave mode, data on MOSI is sampled in the center of each data bit. In master mode, data on MISO is sampled one SYSCLK before the end of each data bit, to provide maximum settling time for the slave device. See Table 30.1 for timing parameters.

31.6. Using SPI1 with the DMA

SPI1 is a DMA-enabled peripheral that can provide autonomous data transfers when used with the DMA. The DMA-enabled SPI1 supports both master and slave mode. The SPI requires two DMA channels for a bidirectional data transfer and also supports unidirectional data transfers using a single DMA channel.

There are no additional control bits in the SPI1 control and configuration SFRs. The configuration is the same in DMA and non-DMA mode. While the SPIF flag and/or SPI interrupts are normally used for non-DMA SPI transfers, a DMA transfer is managed using the DMA enable and DMA full transfer complete flags.

More information on using the SPI1 peripheral can be found in the detailed example code for SPI1 Master and Slave modes.

31.7. Master Mode SPI1 DMA Transfers

The SPI interface does not normally have any handshaking or flow control. Therefore, the Master will transmit all of the output data without waiting on the slave peripheral. The system designer must ensure that the slave peripheral can accept all of the data at the transfer rate.

31.8. Master Mode Bidirectional Data Transfer

A bidirectional SPI Master Mode DMA transfer will transmit a specified number of bytes out on the MOSI pin and receive the same number of bytes on the MISO pin. The MOSI data must be stored in XRAM before initiating the DMA transfers. The DMA will also transfer all the MISO data to XRAM, overwriting any data at the target location.

A bidirectional transfer requires two DMA channels. The first DMA channel transfers data from XRAM to the SPI1DAT SFR and the second DMA channel transfers data from the SPI1DAT SFR to XRAM. The second channel DMA interrupt indicates SPI transfer completion.

In master mode, the NSS pin is an output and the hardware does not manage the NSS pin automatically. Normally, firmware should assert the NSS pin before the SPI transfer and deassert it upon completion of the transfer. When using 4-wire Master mode, bit 2 of SPI1CN controls the state of the NSS pin. When using 3-wire master mode, firmware may use any GPIO pin as NSS.

SFR Definition 31.2. SPI1CN: SPI1 Control

Bit	7	6	5	4	3	2	1	0
Name	SPIF	WCOL	MODF	RXOVRN	NSSMD[1:0]		TXBMT	SPIEN
Type	R/W	R/W	R/W	R/W	R/W		R	R/W
Reset	0	0	0	0	0	1	1	0

SFR Page = 0x0; SFR Address = 0xB0; Bit-Addressable

Bit	Name	Function
7	SPIF	**SPI1 Interrupt Flag.** This bit is set to logic 1 by hardware at the end of a data transfer. If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by software.
6	WCOL	**Write Collision Flag.** This bit is set to logic 1 if a write to SPI1DAT is attempted when TXBMT is 0. When this occurs, the write to SPI1DAT will be ignored, and the transmit buffer will not be written. If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by software.
5	MODF	**Mode Fault Flag.** This bit is set to logic 1 by hardware when a master mode collision is detected (NSS is low, MSTEN = 1, and NSSMD[1:0] = 01). If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by software.
4	RXOVRN	**Receive Overrun Flag (valid in slave mode only).** This bit is set to logic 1 by hardware when the receive buffer still holds unread data from a previous transfer and the last bit of the current transfer is shifted into the SPI1 shift register. If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by software.
3:2	NSSMD[1:0]	**Slave Select Mode.** Selects between the following NSS operation modes: (See Section 31.2 and Section 31.3). 00: 3-Wire Slave or 3-Wire Master Mode. NSS signal is not routed to a port pin. 01: 4-Wire Slave or Multi-Master Mode (Default). NSS is an input to the device. 1x: 4-Wire Single-Master Mode. NSS signal is mapped as an output from the device and will assume the value of NSSMD0.
1	TXBMT	**Transmit Buffer Empty.** This bit will be set to logic 0 when new data has been written to the transmit buffer. When data in the transmit buffer is transferred to the SPI shift register, this bit will be set to logic 1, indicating that it is safe to write a new byte to the transmit buffer.
0	SPIEN	**SPI1 Enable.** 0: SPI disabled. 1: SPI enabled.

Table 31.1. SPI Slave Timing Parameters

Parameter	Description	Min	Max	Units
Master Mode Timing (See Figure 31.8 and Figure 31.9)				
T_{MCKH}	SCK High Time	1 x T_{SYSCLK}	—	ns
T_{MCKL}	SCK Low Time	1 x T_{SYSCLK}	—	ns
T_{MIS}	MISO Valid to SCK Shift Edge	1 x T_{SYSCLK} + 20	—	ns
T_{MIH}	SCK Shift Edge to MISO Change	0	—	ns
Slave Mode Timing (See Figure 31.10 and Figure 31.11)				
T_{SE}	NSS Falling to First SCK Edge	2 x T_{SYSCLK}	—	ns
T_{SD}	Last SCK Edge to NSS Rising	2 x T_{SYSCLK}	—	ns
T_{SEZ}	NSS Falling to MISO Valid	—	4 x T_{SYSCLK}	ns
T_{SDZ}	NSS Rising to MISO High-Z	—	4 x T_{SYSCLK}	ns
T_{CKH}	SCK High Time	5 x T_{SYSCLK}	—	ns
T_{CKL}	SCK Low Time	5 x T_{SYSCLK}	—	ns
T_{SIS}	MOSI Valid to SCK Sample Edge	2 x T_{SYSCLK}	—	ns
T_{SIH}	SCK Sample Edge to MOSI Change	2 x T_{SYSCLK}	—	ns
T_{SOH}	SCK Shift Edge to MISO Change	—	4 x T_{SYSCLK}	ns
T_{SLH}	Last SCK Edge to MISO Change (CKPHA = 1 ONLY)	6 x T_{SYSCLK}	8 x T_{SYSCLK}	ns
Note: T_{SYSCLK} is equal to one period of the device system clock (SYSCLK).				

SFR Definition 32.2. TCON: Timer Control

Bit	7	6	5	4	3	2	1	0
Name	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Page = All Pages; SFR Address = 0x88; Bit-Addressable

Bit	Name	Function
7	TF1	**Timer 1 Overflow Flag.** Set to 1 by hardware when Timer 1 overflows. This flag can be cleared by software but is automatically cleared when the CPU vectors to the Timer 1 interrupt service routine.
6	TR1	**Timer 1 Run Control.** Timer 1 is enabled by setting this bit to 1.
5	TF0	**Timer 0 Overflow Flag.** Set to 1 by hardware when Timer 0 overflows. This flag can be cleared by software but is automatically cleared when the CPU vectors to the Timer 0 interrupt service routine.
4	TR0	**Timer 0 Run Control.** Timer 0 is enabled by setting this bit to 1.
3	IE1	**External Interrupt 1.** This flag is set by hardware when an edge/level of type defined by IT1 is detected. It can be cleared by software but is automatically cleared when the CPU vectors to the External Interrupt 1 service routine in edge-triggered mode.
2	IT1	**Interrupt 1 Type Select.** This bit selects whether the configured $\overline{\text{INT1}}$ interrupt will be edge or level sensitive. $\overline{\text{INT1}}$ is configured active low or high by the IN1PL bit in the IT01CF register (see SFR Definition 17.7). 0: $\overline{\text{INT1}}$ is level triggered. 1: $\overline{\text{INT1}}$ is edge triggered.
1	IE0	**External Interrupt 0.** This flag is set by hardware when an edge/level of type defined by IT1 is detected. It can be cleared by software but is automatically cleared when the CPU vectors to the External Interrupt 0 service routine in edge-triggered mode.
0	IT0	**Interrupt 0 Type Select.** This bit selects whether the configured $\overline{\text{INT0}}$ interrupt will be edge or level sensitive. $\overline{\text{INT0}}$ is configured active low or high by the IN0PL bit in register IT01CF (see SFR Definition 17.7). 0: $\overline{\text{INT0}}$ is level triggered. 1: $\overline{\text{INT0}}$ is edge triggered.

32.3.2. 8-Bit Timers with Auto-Reload

When T3SPLIT is set, Timer 3 operates as two 8-bit timers (TMR3H and TMR3L). Both 8-bit timers operate in auto-reload mode as shown in Figure 32.8. TMR3RLL holds the reload value for TMR3L; TMR3RLH holds the reload value for TMR3H. The TR3 bit in TMR3CN handles the run control for TMR3H. TMR3L is always running when configured for 8-bit Mode.

Each 8-bit timer may be configured to use SYSCLK, SYSCLK divided by 12, the external oscillator clock source divided by 8, or the SmaRTClock. The Timer 3 Clock Select bits (T3MH and T3ML in CKCON) select either SYSCLK or the clock defined by the Timer 3 External Clock Select bits (T3XCLK[1:0] in TMR3CN), as follows:

T3MH	T3XCLK[1:0]	TMR3H Clock Source
0	00	SYSCLK / 12
0	01	SmaRTClock
0	10	Reserved
0	11	External Clock / 8
1	X	SYSCLK

T3ML	T3XCLK[1:0]	TMR3L Clock Source
0	00	SYSCLK / 12
0	01	SmaRTClock
0	10	Reserved
0	11	External Clock / 8
1	X	SYSCLK

The TF3H bit is set when TMR3H overflows from 0xFF to 0x00; the TF3L bit is set when TMR3L overflows from 0xFF to 0x00. When Timer 3 interrupts are enabled, an interrupt is generated each time TMR3H overflows. If Timer 3 interrupts are enabled and TF3LEN (TMR3CN.5) is set, an interrupt is generated each time either TMR3L or TMR3H overflows. When TF3LEN is enabled, software must check the TF3H and TF3L flags to determine the source of the Timer 3 interrupt. The TF3H and TF3L interrupt flags are not cleared by hardware and must be manually cleared by software.

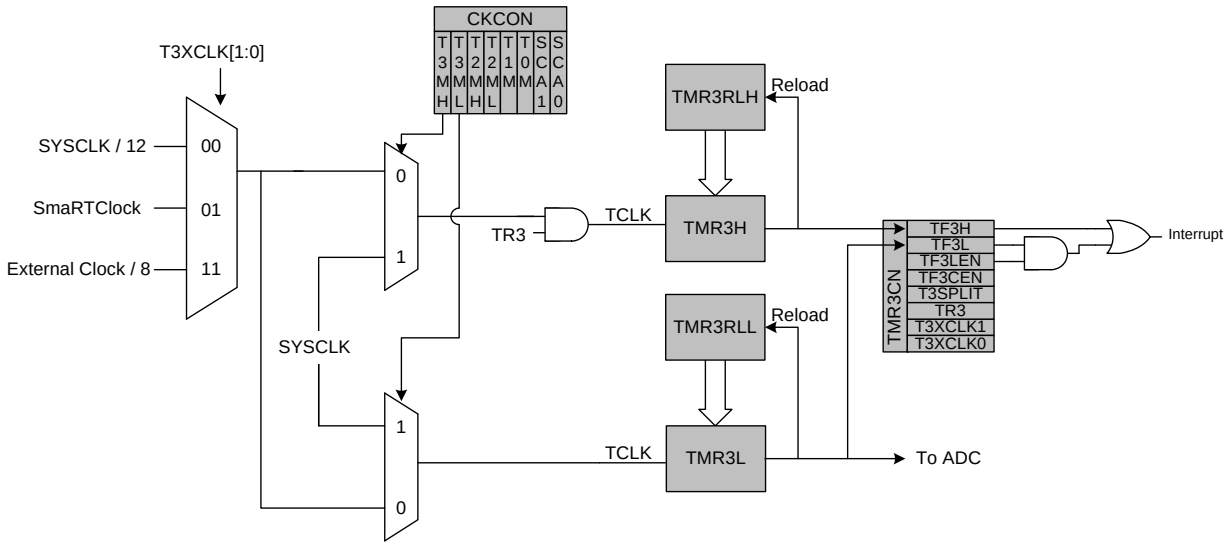

Figure 32.8. Timer 3 8-Bit Mode Block Diagram

32.3.3. SmaRTClock/External Oscillator Capture Mode

The Capture Mode in Timer 3 allows either SmaRTClock or the external oscillator period to be measured against the system clock or the system clock divided by 12. SmaRTClock and the external oscillator period can also be compared against each other.