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	eZ8
Core Size	8-Bit
Speed	20MHz
Connectivity	I²C, IrDA, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	46
Program Memory Size	24KB (24K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 3.6V
Data Converters	A/D 12x10b
Oscillator Type	Internal
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	68-LCC (J-Lead)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f2422vs020sg

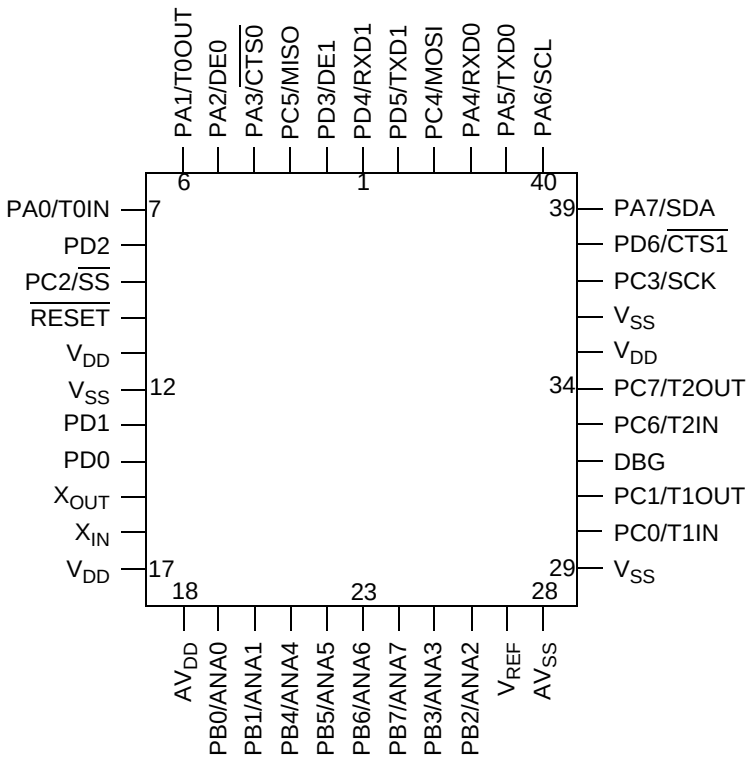

Figure 3. Z8 Encore! XP F64xx Series in 44-Pin Plastic Leaded Chip Carrier (PLCC)

Note: Timer 3 is not available in the 44-pin PLCC package.

Table 23. Interrupt Vectors in Order of Priority

Priority	Program Memory Vector Address	Interrupt Source
Highest	0002H	Reset (not an interrupt)
	0004H	Watchdog Timer (see the Watchdog Timer chapter on page 80)
	0006H	Illegal Instruction Trap (not an interrupt)
	0008H	Timer 2
	000AH	Timer 1
	000CH	Timer 0
	000EH	UART 0 receiver
	0010H	UART 0 transmitter
	0012H	I^2C
	0014H	SPI
	0016H	ADC
	0018H	Port A7 or Port D7, rising or falling input edge
	001AH	Port A6 or Port D6, rising or falling input edge
	001CH	Port A5 or Port D5, rising or falling input edge
	001EH	Port A4 or Port D4, rising or falling input edge
	0020H	Port A3 or Port D3, rising or falling input edge
	0022H	Port A2 or Port D2, rising or falling input edge
	0024H	Port A1 or Port D1, rising or falling input edge
	0026H	Port A0 or Port D0, rising or falling input edge
	0028H	Timer 3 (not available in the 44-pin package)
	002AH	UART 1 receiver
	002CH	UART 1 transmitter
	002EH	DMA
	0030H	Port C3, both input edges
	0032H	Port C2, both input edges
	0034H	Port C1, both input edges
Lowest	0036H	Port C0, both input edges

- Set or clear the CTSE bit to enable or disable control from the remote receiver via the $\overline{\text{CTS}}$ pin

8. Execute an EI instruction to enable interrupts.

The UART is now configured for interrupt-driven data transmission. Because the UART Transmit Data Register is empty, an interrupt is generated immediately. When the UART transmit interrupt is detected, the associated interrupt service routine performs the following functions:

1. Write the UART Control 1 Register to select the outgoing address bit:
   - Set the MULTIPROCESSOR Bit Transmitter (MPBT) if sending an address byte; clear it if sending a data byte.
2. Write the data byte to the UART Transmit Data Register. The transmitter automatically transfers the data to the Transmit Shift Register and transmits the data.
3. Clear the UART transmit interrupt bit in the applicable Interrupt Request Register.
4. Execute the IRET instruction to return from the interrupt service routine and wait for the Transmit Data Register to again become empty.

Receiving Data using the Polled Method

Observe the following procedure to configure the UART for polled data reception:

1. Write to the UART Baud Rate High and Low Byte registers to set the appropriate baud rate.
2. Enable the UART pin functions by configuring the associated GPIO port pins for alternate function operation.
3. Write to the UART Control 1 Register to enable MULTIPROCESSOR Mode functions, if appropriate.
4. Write to the UART Control 0 Register to:
   - Set the receive enable bit (REN) to enable the UART for data reception
   - Enable parity, if appropriate and if MULTIPROCESSOR Mode is not enabled, and select either even or odd parity
5. Check the RDA bit in the UART Status 0 Register to determine if the Receive Data Register contains a valid data byte (indicated by a 1). If RDA is set to 1 to indicate available data, continue to Step 6. If the Receive Data Register is empty (indicated by a 0), continue to monitor the RDA bit awaiting reception of the valid data.

UART Control 0 and Control 1 Registers

The UART Control 0 and Control 1 Registers, shown in Tables 57 and 58, configure the properties of the UART's transmit and receive operations. The UART Control registers must not been written while the UART is enabled.

Table 57. UART Control 0 Register (U*x*CTL0)

Bit	7	6	5	4	3	2	1	0
Field	TEN	REN	CTSE	PEN	PSEL	SBRK	STOP	LBEN
RESET	0							
R/W	R/W							
Address	F42H and F4AH							

Bit	Description
[7] TEN	**Transmit Enable** This bit enables or disables the transmitter. The enable is also controlled by the $\overline{\text{CTS}}$ signal and the CTSE bit. If the $\overline{\text{CTS}}$ signal is low and the CTSE bit is 1, the transmitter is enabled. 0 = Transmitter disabled. 1 = Transmitter enabled.
[6] REN	**Receive Enable** This bit enables or disables the receiver. 0 = Receiver disabled. 1 = Receiver enabled.
[5] CTSE	**CTS Enable** 0 = The $\overline{\text{CTS}}$ signal has no effect on the transmitter. 1 = The UART recognizes the $\overline{\text{CTS}}$ signal as an enable control from the transmitter.
[4] PEN	**Parity Enable** This bit enables or disables parity. Even or odd is determined by the PSEL bit. It is overridden by the MPEN bit. 0 = Parity is disabled. 1 = The transmitter sends data with an additional parity bit and the receiver receives an additional parity bit.
[3] PSEL	**Parity Select** 0 = Even parity is transmitted and expected on all received data. 1 = Odd parity is transmitted and expected on all received data.
[2] SBRK	**Send Break** This bit pauses or breaks data transmission. Sending a break interrupts any transmission in progress, so ensure that the transmitter has finished sending data before setting this bit. 0 = No break is sent. 1 = The output of the transmitter is zero.

Slave Operation

The SPI block is configured for SLAVE Mode operation by setting the SPIEN bit to 1 and the MMEN bit to 0 in the SPICTL Register and setting the SSIO bit to 0 in the SPIMODE Register. The IRQE, PHASE, CLKPOL, WOR bits in the SPICTL Register and the NUMBITS field in the SPIMODE Register must be set to be consistent with the other SPI devices. The STR bit in the SPICTL Register may be used if appropriate to force a *start-up* interrupt. The BIRQ bit in the SPICTL Register and the SSV bit in the SPIMODE Register are not used in SLAVE Mode. The SPI baud rate generator is not used in SLAVE Mode so the SPIBRH and SPIBRL registers need not be initialized.

If the slave has data to send to the master, the data must be written to the SPIDAT Register before the transaction starts (first edge of SCK when $\overline{SS}$ is asserted). If the SPIDAT Register is not written prior to the slave transaction, the MISO pin outputs whatever value is currently in the SPIDAT Register.

Due to the delay resulting from synchronization of the SPI input signals to the internal system clock, the maximum SPICLK baud rate that can be supported in SLAVE Mode is the system clock frequency (X_{IN}) divided by 8. This rate is controlled by the SPI master.

Error Detection

The SPI contains error detection logic to support SPI communication protocols and recognize when communication errors have occurred. The SPI Status Register indicates when a data transmission error has been detected.

Overrun (Write Collision)

An overrun error (write collision) indicates that a write to the SPI Data Register was attempted while a data transfer was in progress (in either MASTER or SLAVE modes). An overrun sets the OVR bit in the SPI Status Register to 1. Writing a 1 to OVR clears this error flag. The data register is not altered when a write occurs while data transfer is in progress.

Mode Fault (Multimaster Collision)

A mode fault indicates when more than one Master is trying to communicate at the same time (a multimaster collision). The mode fault is detected when the enabled Master's $\overline{SS}$ pin is asserted. A mode fault sets the COL bit in the SPI Status Register to 1. Writing a 1 to COL clears this error flag.

Slave Mode Abort

In the SLAVE Mode of operation, if the $\overline{SS}$ pin deasserts before all bits in a character have been transferred, the transaction is aborted. When this condition occurs, the ABT bit is set in the SPISTAT Register as well as the IRQ bit (indicating the transaction is complete).

Bit	Description (Continued)
[5] COL	**Collision** 0 = A multimaster collision (mode fault) has not occurred. 1 = A multimaster collision (mode fault) has been detected.
[4] ABT	**Slave Mode Transaction Abort** This bit is set if the SPI is configured in slave mode, a transaction is occurring and $\overline{SS}$ deasserts before all bits of a character have been transferred as defined by the NUMBITS field of the SPIMODE Register. The IRQ bit also sets, indicating the transaction has completed. 0 = A slave mode transaction abort has not occurred. 1 = A slave mode transaction abort has been detected.
[3:2]	**Reserved** These bits are reserved and must be programmed to 00.
[1] TXST	**Transmit Status** 0 = No data transmission currently in progress. 1 = Data transmission currently in progress.
[0] SLAS	**Slave Select** If SPI enabled as a Slave, then the following conditions are true: 0 = $\overline{SS}$ input pin is asserted (Low). 1 = $\overline{SS}$ input is not asserted (High). If SPI enabled as a Master, this bit is not applicable.

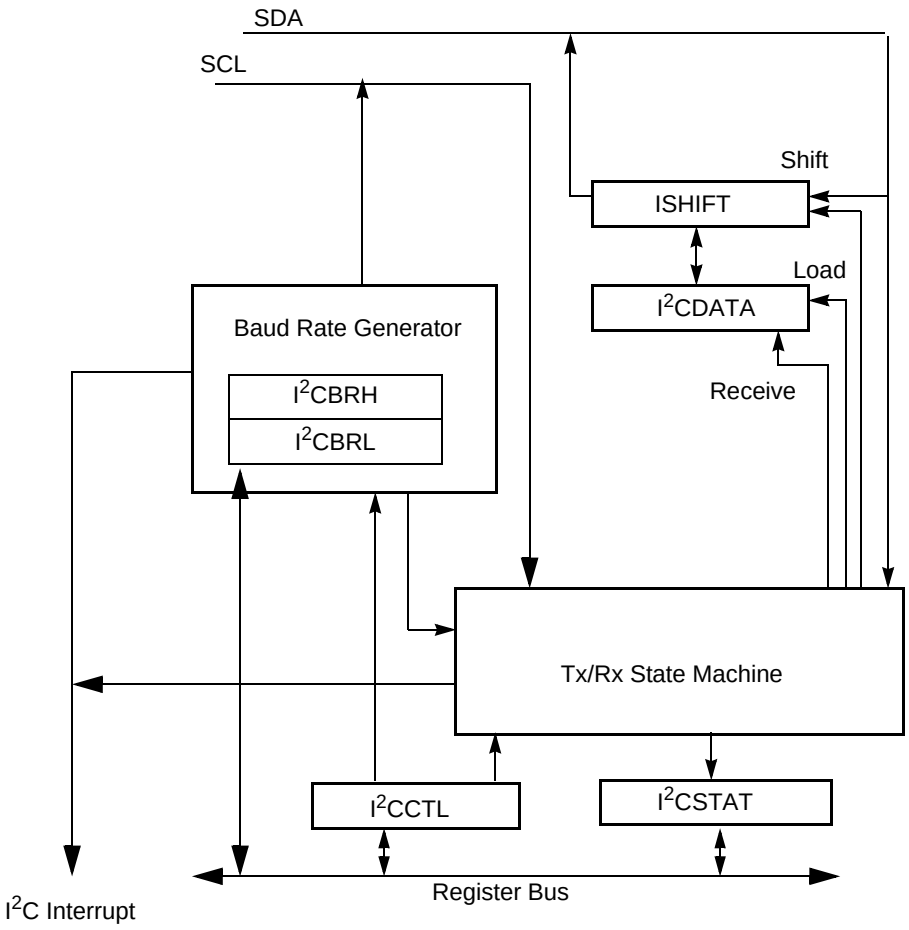

Figure 27. I²C Controller Block Diagram

Operation

The I²C Controller operates in MASTER Mode to transmit and receive data. Only a single master is supported. Arbitration between two masters must be accomplished in software. I²C supports the following operations:

- Master transmits to a 7-bit slave
- Master transmits to a 10-bit slave
- Master receives from a 7-bit slave
- Master receives from a 10-bit slave

S	Slave Address	W = 0	A	Data	A	Data	A	Data	A/A	P/S

Figure 29. 7-Bit Addressed Slave Data Transfer Format

Observe the following procedure for a transmit operation to a 7-bit addressed slave:

1. Software asserts the IEN bit in the I^2C Control Register.
2. Software asserts the TXI bit of the I^2C Control Register to enable transmit interrupts.
3. The I^2C interrupt asserts, because the I^2C Data Register is empty
4. Software responds to the TDRE bit by writing a 7-bit slave address plus write bit (=0) to the I^2C Data Register.
5. Software asserts the start bit of the I^2C Control Register.
6. The I^2C Controller sends the start condition to the I^2C slave.
7. The I^2C Controller loads the I^2C Shift Register with the contents of the I^2C Data Register.
8. After one bit of address has been shifted out by the SDA signal, the transmit interrupt is asserted (TDRE = 1).
9. Software responds by writing the transmit data into the I^2C Data Register.
10. The I^2C Controller shifts the rest of the address and write bit out by the SDA signal.
11. If the I^2C slave sends an acknowledge (by pulling the SDA signal Low) during the next High period of SCL **t**he I^2C Controller sets the ACK bit in the I^2C Status Register. Continue with Step 12.

    If the slave does not acknowledge, the Not Acknowledge interrupt occurs (NCKI bit is set in the Status Register, ACK bit is cleared). Software responds to the Not Acknowledge interrupt by setting the stop and flush bits and clearing the TXI bit. The I^2C Controller sends the stop condition on the bus and clears the stop and NCKI bits. The transaction is complete (ignore the following steps).
12. The I^2C Controller loads the contents of the I^2C Shift Register with the contents of the I^2C Data Register.
13. The I^2C Controller shifts the data out of using the SDA signal. After the first bit is sent, the transmit interrupt is asserted.
14. If more bytes remain to be sent, return to Step 9.
15. Software responds by setting the stop bit of the I^2C Control Register (or start bit to initiate a new transaction). In the stop case, software clears the TXI bit of the I^2C Control Register at the same time.

10. The I2C Controller shifts the rest of the first byte of address and write bit out the SDA signal.
11. If the I2C slave sends an acknowledge by pulling the SDA signal Low during the next High period of SCL the I2C Controller sets the ACK bit in the I2C Status Register. Continue with Step 12.

    If the slave does not acknowledge the first address byte, the I2C Controller sets the NCKI bit and clears the ACK bit in the I2C Status Register. Software responds to the Not Acknowledge interrupt by setting the stop and flush bits and clearing the TXI bit. The I2C Controller sends the stop condition on the bus and clears the stop and NCKI bits. The transaction is complete (ignore following steps).
12. The I2C Controller loads the I2C Shift Register with the contents of the I2C Data Register (2nd byte of address).
13. The I2C Controller shifts the second address byte out the SDA signal. After the first bit has been sent, the transmit interrupt is asserted.
14. Software responds by setting the stop bit in the I2C Control Register. The TXI bit can be cleared at the same time.
15. Software polls the stop bit of the I2C Control Register. Hardware deasserts the stop bit when the transaction is completed (stop condition has been sent).
16. Software checks the ACK bit of the I2C Status Register. If the slave acknowledged, the ACK bit is = 1. If the slave does not acknowledge, the ACK bit is = 0. The NCKI interrupt do not occur because the stop bit was set.

Write Transaction with a 10-Bit Address

Figure 31 displays the data transfer format for a 10-bit addressed slave. Shaded regions indicate data transferred from the I2C Controller to slaves and unshaded regions indicate data transferred from the slaves to the I2C Controller.

S	Slave Address 1st 7 bits	W = 0	A	Slave Address 2nd Byte	A	Data	A	Data	A/A	P/S

Figure 31. 10-Bit Addressed Slave Data Transfer Format

The first seven bits transmitted in the first byte are 11110XX. The two bits XX are the two most significant bits of the 10-bit address. The lowest bit of the first byte transferred is the read/write control bit (=0). The transmit operation is carried out in the same manner as 7-bit addressing.

Observe the following procedure for a transmit operation on a 10-bit addressed slave:

1. Software asserts the IEN bit in the I2C Control Register.

I²C Control Register

The I²C Control Register, shown in Table 73, enables I²C operation.

Table 73. I²C Control Register (I2CCTL)

Bit	7	6	5	4	3	2	1	0
Field	IEN	START	STOP	BIRQ	TXI	NAK	FLUSH	FILTEN
RESET	0							
R/W	R/W	R/W1	R/W1	R/W	R/W	R/W1	W1	R/W
Address	F52H							

Bit	Description
[7] IEN	**I²C Enable** 1 = The I²C transmitter and receiver are enabled. 0 = The I²C transmitter and receiver are disabled.
[6] START	**Send Start Condition** This bit sends the Start condition. Once asserted, it is cleared by the I²C Controller after it sends the START condition or if the IEN bit is deasserted. If this bit is 1, it cannot be cleared to 0 by writing to the register. After this bit is set, the Start condition is sent if there is data in the I²C Data Register or I²C Shift Register. If there is no data in one of these registers, the I²C Controller waits until the Data Register is written. If this bit is set while the I²C Controller is shifting out data, it generates a start condition after the byte shifts and the acknowledge phase completes. If the stop bit is also set, it also waits until the stop condition is sent before the sending the start condition.
[5] STOP	**Send Stop Condition** This bit causes the I²C Controller to issue a Stop condition after the byte in the I²C Shift Register has completed transmission or after a byte has been received in a receive operation. AFter it is set, this bit is reset by the I²C Controller after a Stop condition has been sent or by deasserting the IEN bit. If this bit is 1, it cannot be cleared to 0 by writing to the register.
[4] BIRQ	**Baud Rate Generator Interrupt Request** This bit allows the I²C Controller to be used as an additional timer when the I²C Controller is disabled. This bit is ignored when the I²C Controller is enabled. 1 = An interrupt occurs every time the baud rate generator counts down to one. 0 = No baud rate generator interrupt occurs.
[3] TXI	**Enable TDRE Interrupts** This bit enables the transmit interrupt when the I²C Data Register is empty (TDRE = 1). 1 = Transmit interrupt (and DMA transmit request) is enabled. 0 = Transmit interrupt (and DMA transmit request) is disabled.
[2] NAK	**Send NAK** This bit sends a Not Acknowledge condition after the next byte of data has been read from the I²C slave. Once asserted, it is deasserted after a Not Acknowledge is sent or the IEN bit is deasserted. If this bit is 1, it cannot be cleared to 0 by writing to the register.

Bit	Description (Continued)
[1] FLUSH	**Flush Data** Setting this bit to 1 clears the I²C Data Register and sets the TDRE bit to 1. This bit allows flushing of the I²C Data Register when a Not Acknowledge interrupt is received after the data has been sent to the I²C Data Register. Reading this bit always returns 0.
[0] FILTEN	**I²C Signal Filter Enable** This bit enables low-pass digital filters on the SDA and SCL input signals. These filters reject any input pulse with periods less than a full system clock cycle. The filters introduce a 3-system clock cycle latency on the inputs. 1 = low-pass filters are enabled. 0 = low-pass filters are disabled.

I²C Baud Rate High and Low Byte Registers

The I²C Baud Rate High and Low Byte registers, shown in Tables 74 and 75, combine to form a 16-bit reload value, BRG[15:0], for the I²C Baud Rate Generator.

When the I²C is disabled, the Baud Rate Generator can function as a basic 16-bit timer with interrupt on time-out. To configure the Baud Rate Generator as a timer with interrupt on time-out, complete the following procedure:

1. Disable the I²C by clearing the IEN bit in the I²C Control Register to 0.
2. Load the appropriate 16-bit count value into the I²C Baud Rate High and Low Byte registers.
3. Enable the Baud Rate Generator timer function and associated interrupt by setting the BIRQ bit in the I²C Control Register to 1.

When configured as a general purpose timer, the interrupt interval is calculated using the following equation:

Interrupt Interval (s) = System Clock Period (s) × BRG[15:0]

Observe the following procedure to setup the Flash Sector Protect Register from user code:

1. Write 00H to the Flash Control Register to reset the Flash Controller.
2. Write 5EH to the Flash Control Register to select the Flash Sector Protect Register.
3. Read and/or write the Flash Sector Protect Register which is now at Register File address FF9H.
4. Write 00H to the Flash Control Register to return the Flash Controller to its reset state.

Flash Write Protection Option Bit

The Flash Write Protect option bit can be enabled to block all program and erase operations from user code. For more information, see the Option Bits chapter on page 180.

Byte Programming

When the Flash Controller is unlocked, writes to Flash memory from user code will program a byte into the Flash if the address is located in the unlocked page. An erased Flash byte contains all ones (FFH). The programming operation can only be used to change bits from one to zero. To change a Flash bit (or multiple bits) from zero to one requires a Page Erase or Mass Erase operation.

Byte programming can be accomplished using the eZ8 CPU's LDC or LDCI instructions. For a description of the LDC and LDCI instructions, refer to the eZ8 CPU Core User Manual (UM0128), which is available for download on www.zilog.com.

While the Flash Controller programs Flash memory, the eZ8 CPU idles but the system clock and on-chip peripherals continue to operate. Interrupts that occur when a programming operation is in progress are serviced after the programming operation is complete. To exit programming mode and lock the Flash Controller, write 00H to the Flash Control Register.

User code cannot program Flash memory on a page that resides in a protected sector. When user code writes memory locations, only addresses located in the unlocked page are programmed. Memory writes outside of the unlocked page are ignored.

! **Caution:** Each memory location must not be programmed more than twice before an erase occurs.

Observe the following procedure to program the Flash from user code:

1. Write 00H to the Flash Control Register to reset the Flash Controller.
2. Write the page of memory to be programmed to the Page Select Register.
3. Write the first unlock command 73H to the Flash Control Register.

For more information about bypassing the Flash Controller, refer to the Third Party Flash Programming Support for Z8 Encore! MCUs Application Note (AN0117), which is available for download at www.zilog.com.

Flash Controller Behavior in Debug Mode

The following changes in Flash Controller behavior occur when the Flash Controller is accessed using the On-Chip Debugger:

- The Flash Write Protect option bit is ignored
- The Flash Sector Protect Register is ignored for programming and erase operations
- Programming operations are not limited to the page selected in the Page Select Register
- Bits in the Flash Sector Protect Register can be written to one or zero
- The second write of the Page Select Register to unlock the Flash Controller is not necessary
- The Page Select Register can be written when the Flash Controller is unlocked
- The Mass Erase command is enabled through the Flash Control Register

! **Caution:** For security reasons, the Flash Controller allows only a single page to be opened for write/erase operations. When writing multiple Flash pages, the Flash Controller must go through the unlock sequence again to select another page.

Flash Control Register Definitions

This section defines the features of the following Flash Control registers.

Flash Control Register: see page 175

Flash Status Register: see page 177

Page Select Register: see page 177

Flash Sector Protect Register: see page 178

Flash Frequency High and Low Byte Registers: see page 179

Flash Control Register

The Flash Control Register, shown in Table 93, unlocks the Flash Controller for programming and erase operations, or to select the Flash Sector Protect Register.

```
DBG ← Size[7:0]
DBG ← 1-256 data bytes
```

Read Register (09H). The Read Register command reads data from the Register File. Data can be read 1-256 bytes at a time (256 bytes can be read by setting size to zero). If the device is not in DEBUG Mode or if the Read Protect option bit is enabled, this command returns FFH for all the data values.

```
DBG ← 09H
DBG ← {4'h0,Register Address[11:8]
DBG ← Register Address[7:0]
DBG ← Size[7:0]
DBG 1-256 data bytes
```

Write Program Memory (0AH). The Write Program Memory command writes data to program memory. This command is equivalent to the LDC and LDCI instructions. Data can be written 1-65536 bytes at a time (65536 bytes can be written by setting size to zero). The on-chip Flash Controller must be written to and unlocked for the programming operation to occur. If the Flash Controller is not unlocked, the data is discarded. If the device is not in DEBUG Mode or if the Read Protect option bit is enabled, the data is discarded.

```
DBG ← 0AH
DBG ← Program Memory Address[15:8]
DBG ← Program Memory Address[7:0]
DBG ← Size[15:8]
DBG ← Size[7:0]
DBG ← 1-65536 data bytes
```

Read Program Memory (0BH). The Read Program Memory command reads data from program memory. This command is equivalent to the LDC and LDCI instructions. Data can be read 1-65536 bytes at a time (65536 bytes can be read by setting size to zero). If the device is not in DEBUG Mode or if the Read Protect option bit is enabled, this command returns FFH for the data.

```
DBG ← 0BH
DBG ← Program Memory Address[15:8]
DBG ← Program Memory Address[7:0]
DBG ← Size[15:8]
DBG ← Size[7:0]
DBG 1-65536 data bytes
```

Write Data Memory (0CH). The Write Data Memory command writes data to Data Memory. This command is equivalent to the LDE and LDEI instructions. Data can be written 1-65536 bytes at a time (65536 bytes can be written by setting size to zero). If the device is not in DEBUG Mode or if the Read Protect option bit is enabled, the data is discarded.

```
DBG ← 0CH
DBG ← Data Memory Address[15:8]
DBG ← Data Memory Address[7:0]
DBG ← Size[15:8]
DBG ← Size[7:0]
DBG ← 1-65536 data bytes
```

Table 111 list the Flash memory electrical characteristics and timing.

Table 111. Flash Memory Electrical Characteristics and Timing

	V_{DD} = 3.0–3.6V T_A = –40°C to 125°C				
Parameter	**Minimum**	**Typical**	**Maximum**	**Units**	**Notes**
Flash Byte Read Time	50	–	–	ns	
Flash Byte Program Time	20	–	40	µs	
Flash Page Erase Time	10	–	–	ms	
Flash Mass Erase Time	200	–	–	ms	
Writes to Single Address Before Next Erase	–	–	2		
Flash Row Program Time	–	–	8	ms	Cumulative program time for single row cannot exceed limit before next erase. This parameter is only an issue when bypassing the Flash Controller.
Data Retention	100	–	–	years	25°C
Endurance, –40°C to 105°C	10,000	–	–	cycles	Program/erase cycles
Endurance, 106°C to 125°C	1,000	–	–	cycles	Program/erase cycles

Table 112 lists the Watchdog Timer electrical characteristics and timing.

Table 112. Watchdog Timer Electrical Characteristics and Timing

		V_{DD} = 3.0–3.6V T_A = –40°C to 125°C				
Symbol	**Parameter**	**Minimum**	**Typical**	**Maximum**	**Units**	**Conditions**
F_{WDT}	WDT Oscillator Frequency	5	10	20	kHz	
I_{WDT}	WDT Oscillator Current including internal RC Oscillator	–	<1	5	µA	

SPI Master Mode Timing

Figure 53 and Table 118 provide timing information for SPI Master Mode pins. Timing is shown with SCK rising edge used to source MOSI output data, SCK falling edge used to sample MISO input data. Timing on the SS output pin(s) is controlled by software.

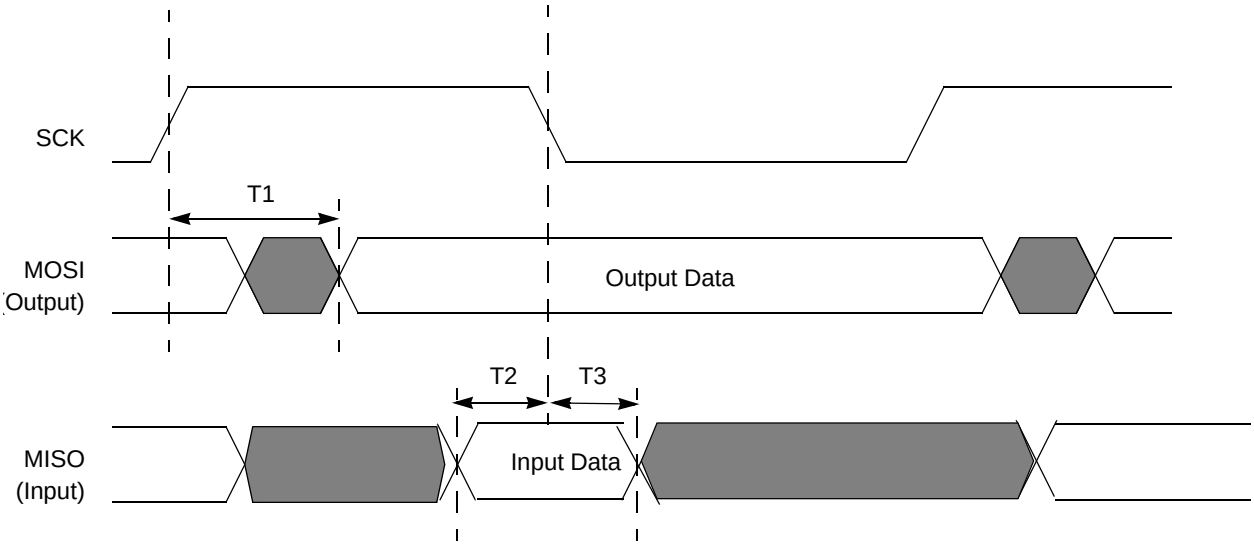

Figure 53. SPI Master Mode Timing

Table 118. SPI Master Mode Timing

		Delay (ns)	
Parameter	**Abbreviation**	**Min**	**Max**
SPI Master			
T_1	SCK Rise to MOSI output Valid Delay	–5	+5
T_2	MISO input to SCK (receive edge) Setup Time	20	
T_3	MISO input to SCK (receive edge) Hold Time	0	

Upper Nibble (Hex) \ Lower Nibble (Hex)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	1.2 **BRK**	2.2 **SRP** IM	2.3 **ADD** r1,r2	2.4 **ADD** r1,Ir2	3.3 **ADD** R2,R1	3.4 **ADD** IR2,R1	3.3 **ADD** R1,IM	3.4 **ADD** IR1,IM	4.3 **ADDX** ER2,ER1	4.3 **ADDX** IM,ER1	2.3 **DJNZ** r1,X	2.2 **JR** cc,X	2.2 **LD** r1,IM	3.2 **JP** cc,DA	1.2 **INC** r1	1.2 **NOP**
1	2.2 **RLC** R1	2.3 **RLC** IR1	2.3 **ADC** r1,r2	2.4 **ADC** r1,Ir2	3.3 **ADC** R2,R1	3.4 **ADC** IR2,R1	3.3 **ADC** R1,IM	3.4 **ADC** IR1,IM	4.3 **ADCX** ER2,ER1	4.3 **ADCX** IM,ER1						See 2nd Opcode Map
2	2.2 **INC** R1	2.3 **INC** IR1	2.3 **SUB** r1,r2	2.4 **SUB** r1,Ir2	3.3 **SUB** R2,R1	3.4 **SUB** IR2,R1	3.3 **SUB** R1,IM	3.4 **SUB** IR1,IM	4.3 **SUBX** ER2,ER1	4.3 **SUBX** IM,ER1						1,2 **ATM**
3	2.2 **DEC** R1	2.3 **DEC** IR1	2.3 **SBC** r1,r2	2.4 **SBC** r1,Ir2	3.3 **SBC** R2,R1	3.4 **SBC** IR2,R1	3.3 **SBC** R1,IM	3.4 **SBC** IR1,IM	4.3 **SBCX** ER2,ER1	4.3 **SBCX** IM,ER1						
4	2.2 **DA** R1	2.3 **DA** IR1	2.3 **OR** r1,r2	2.4 **OR** r1,Ir2	3.3 **OR** R2,R1	3.4 **OR** IR2,R1	3.3 **OR** R1,IM	3.4 **OR** IR1,IM	4.3 **ORX** ER2,ER1	4.3 **ORX** IM,ER1						
5	2.2 **POP** R1	2.3 **POP** IR1	2.3 **AND** r1,r2	2.4 **AND** r1,Ir2	3.3 **AND** R2,R1	3.4 **AND** IR2,R1	3.3 **AND** R1,IM	3.4 **AND** IR1,IM	4.3 **ANDX** ER2,ER1	4.3 **ANDX** IM,ER1						1.2 **WDT**
6	2.2 **COM** R1	2.3 **COM** IR1	2.3 **TCM** r1,r2	2.4 **TCM** r1,Ir2	3.3 **TCM** R2,R1	3.4 **TCM** IR2,R1	3.3 **TCM** R1,IM	3.4 **TCM** IR1,IM	4.3 **TCMX** ER2,ER1	4.3 **TCMX** IM,ER1						1.2 **STOP**
7	2.2 **PUSH** R2	2.3 **PUSH** IR2	2.3 **TM** r1,r2	2.4 **TM** r1,Ir2	3.3 **TM** R2,R1	3.4 **TM** IR2,R1	3.3 **TM** R1,IM	3.4 **TM** IR1,IM	4.3 **TMX** ER2,ER1	4.3 **TMX** IM,ER1						1.2 **HALT**
8	2.5 **DECW** RR1	2.6 **DECW** IRR1	2.5 **LDE** r1,Irr2	2.9 **LDEI** Ir1,Irr2	3.2 **LDX** r1,ER2	3.3 **LDX** Ir1,ER2	3.4 **LDX** IRR2,R1	3.5 **LDX** IRR2,IR1	3.4 **LDX** r1,rr2,X	3.4 **LDX** rr1,r2,X						1.2 **DI**
9	2.2 **RL** R1	2.3 **RL** IR1	2.5 **LDE** r2,Irr1	2.9 **LDEI** Ir2,Irr1	3.2 **LDX** r2,ER1	3.3 **LDX** Ir2,ER1	3.4 **LDX** R2,IRR1	3.5 **LDX** IR2,IRR1	3.3 **LEA** r1,r2,X	3.5 **LEA** rr1,rr2,X						1.2 **EI**
A	2.5 **INCW** RR1	2.6 **INCW** IRR1	2.3 **CP** r1,r2	2.4 **CP** r1,Ir2	3.3 **CP** R2,R1	3.4 **CP** IR2,R1	3.3 **CP** R1,IM	3.4 **CP** IR1,IM	4.3 **CPX** ER2,ER1	4.3 **CPX** IM,ER1						1.4 **RET**
B	2.2 **CLR** R1	2.3 **CLR** IR1	2.3 **XOR** r1,r2	2.4 **XOR** r1,Ir2	3.3 **XOR** R2,R1	3.4 **XOR** IR2,R1	3.3 **XOR** R1,IM	3.4 **XOR** IR1,IM	4.3 **XORX** ER2,ER1	4.3 **XORX** IM,ER1						1.5 **IRET**
C	2.2 **RRC** R1	2.3 **RRC** IR1	2.5 **LDC** r1,Irr2	2.9 **LDCI** Ir1,Irr2	2.3 **JP** IRR1	2.9 **LDC** Ir1,Irr2		3.4 **LD** r1,r2,X	3.2 **PUSHX** ER2							1.2 **RCF**
D	2.2 **SRA** R1	2.3 **SRA** IR1	2.5 **LDC** r2,Irr1	2.9 **LDCI** Ir2,Irr1	2.6 **CALL** IRR1	2.2 **BSWAP** R1	3.3 **CALL** DA	3.4 **LD** r2,r1,X	3.2 **POPX** ER1							1.2 **SCF**
E	2.2 **RR** R1	2.3 **RR** IR1	2.2 **BIT** p,b,r1	2.3 **LD** r1,Ir2	3.2 **LD** R2,R1	3.3 **LD** IR2,R1	3.2 **LD** R1,IM	3.3 **LD** IR1,IM	4.2 **LDX** ER2,ER1	4.2 **LDX** IM,ER1						1.2 **CCF**
F	2.2 **SWAP** R1	2.3 **SWAP** IR1	2.6 **TRAP** Vector	2.3 **LD** Ir1,r2	2.8 **MULT** RR1	3.3 **LD** R2,IR1	3.3 **BTJ** p,b,r1,X	3.4 **BTJ** p,b,Ir1,X								

Figure 60. First Op Code Map

Hex Address: F4A

Table 182. UART Control 0 Register (U*x*CTL0)

Bit	7	6	5	4	3	2	1	0
Field	TEN	REN	CTSE	PEN	PSEL	SBRK	STOP	LBEN
RESET	0							
R/W	R/W							
Address	F42H and F4AH							

Hex Address: F4B

Table 183. UART Control 1 Register (U*x*CTL1)

Bit	7	6	5	4	3	2	1	0
Field	MPMD[1]	MPEN	MPMD[0]	MPBT	DEPOL	BRGCTL	RDAIRQ	IREN
RESET	0							
R/W	R/W							
Address	F43H and F4BH							

Hex Address: F4C

Table 184. UART Status 1 Register (U*x*STAT1)

Bit	7	6	5	4	3	2	1	0
Field	Reserved						NEWFRM	MPRX
RESET	0							
R/W	R				R/W		R	
Address	F44H and F4CH							

Hex Address: F4D

Table 185. UART Address Compare Register (U*x*ADDR)

Bit	7	6	5	4	3	2	1	0
Field	COMP_ADDR							
RESET	0							
R/W	R/W							
Address	F45H and F4DH							

Hex Address: F51

Table 189. I²C Status Register (I2CSTAT)

Bit	7	6	5	4	3	2	1	0
Field	TDRE	RDRF	ACK	10B	RD	TAS	DSS	NCKI
RESET	1	0						
R/W	R							
Address	F51H							

Hex Address: F52

Table 190. I²C Control Register (I2CCTL)

Bit	7	6	5	4	3	2	1	0
Field	IEN	START	STOP	BIRQ	TXI	NAK	FLUSH	FILTEN
RESET	0							
R/W	R/W	R/W1	R/W1	R/W	R/W	R/W1	W1	R/W
Address	F52H							

Hex Address: F53

Table 191. I²C Baud Rate High Byte Register (I2CBRH)

Bit	7	6	5	4	3	2	1	0
Field	BRH							
RESET	FFH							
R/W	R/W							
Address	F53H							

Hex Address: F54

Table 192. I²C Baud Rate Low Byte Register (I2CBRL)

Bit	7	6	5	4	3	2	1	0
Field	BRL							
RESET	FFH							
R/W	R/W							
Address	F54H							

G

H

I