

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	Active
Core Processor	eZ8
Core Size	8-Bit
Speed	20MHz
Connectivity	I ² C, IrDA, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LED, LVD, POR, PWM, Temp Sensor, WDT
Number of I/O	23
Program Memory Size	24KB (24K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	3K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SOIC (0.295", 7.50mm Width)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f2480sj020eg

Chapter 3. Address Space

The eZ8 CPU can access the following three distinct address spaces:

- The Register File contains addresses for general-purpose registers, eZ8 CPU, peripherals and GPIO port control registers
- The Program Memory contains addresses for all memory locations having executable code and/or data
- The Data Memory contains addresses for all memory locations that contain data only

These three address spaces are covered briefly in the following sections. For more details about the eZ8 CPU and its address space, refer to the [eZ8 CPU Core User Manual \(UM0128\)](#), available for download at www.zilog.com.

3.1. Register File

The Register File address space in the Z8 Encore![®] MCU is 4KB (4096 bytes). The Register File is composed of two sections: control registers and general-purpose registers. When instructions are executed, registers defined as sources are read and registers defined as destinations are written. The architecture of the eZ8 CPU allows all general-purpose registers to function as accumulators, address pointers, index registers, stack areas, or scratch pad memory.

The upper 256 bytes of the 4KB Register File address space are reserved for control of the eZ8 CPU, on-chip peripherals and the input/output ports. These registers are located at addresses F00H to FFFH. Some of the addresses within the 256 B control register sections are reserved (that is, unavailable). Reading from a reserved Register File address returns an undefined value. Zilog does not recommend writing to the reserved Register File addresses because doing so can produce unpredictable results.

The on-chip Register RAM always begins at address 000H in the Register File address space. The F1680 Series MCU contains 1KB or 2KB of on-chip Register RAM. Reading from Register File addresses outside the available RAM addresses (and not within the control register address space) returns an undefined value. Writing to these Register File addresses produces no effect.

In addition, the F1680 Series MCU contains 1KB of on-chip Program RAM. Normally it is used as Program RAM and is present in the Program Memory address space (see the [Program Memory](#) section on page 20). However, it can also be used as additional Register RAM present in the Register File address space 800H–BFFH (1KB Program RAM, 2KB Register RAM), or 400H–7FFH (1KB Program RAM, 1KB Register RAM), if you do not

Chapter 6. Low-Power Modes

The Z8 Encore! XP F1680 Series products have power-saving features. The highest level of power reduction is provided by the STOP Mode. The next lower level of power reduction is provided by the HALT Mode.

Further power savings can be implemented by disabling individual peripheral blocks while in NORMAL Mode.

6.1. STOP Mode

Executing the eZ8 CPU's Stop instruction places the device into STOP Mode. In STOP Mode, the operating characteristics are:

- Primary crystal oscillator and internal precision oscillator are stopped; XIN and XOUT (if previously enabled) are disabled and PA0/PA1 reverts to the states programmed by the GPIO registers
- System clock is stopped
- eZ8 CPU is stopped
- Program counter (PC) stops incrementing
- Watchdog Timer's internal RC oscillator continues operating if enabled by the Oscillator Control Register
- If enabled, the Watchdog Timer (WDT) logic continues operating
- If enabled, the 32kHz secondary oscillator continues operating
- If enabled for operation in STOP Mode, the Timer logic continues to operate with 32kHz secondary oscillator as the Timer clock source
- If enabled for operation in STOP Mode by the associated Flash option bit, the VBO protection circuit continues operating; the low-voltage detection circuit continues to operate if enabled by the Power Control Register
- Low-Power Operational Amplifier and comparator continue to operate if enabled by the Power Control Register
- All other on-chip peripherals are idle

Table 18. Port Alternate Function Mapping, 28-Pin Parts^{1,2} (Continued)

Port	Pin	Mnemonic	Alternate Function Description	Alternate Function Set Register AFS1
Port B	PB0	Reserved		AFS1[0]: 0
		ANA0/AMPOUT	ADC Analog Input/LPO Output	AFS1[0]: 1
	PB1	Reserved		AFS1[1]: 0
		ANA1/AMPINN	ADC Analog Input/LPO Input (N)	AFS1[1]: 1
	PB2	Reserved		AFS1[2]: 0
		ANA2/AMPINP	ADC Analog Input/LPO Input (P)	AFS1[2]: 1
	PB3	CLKIN	External Clock Input	AFS1[3]: 0
		ANA3	ADC Analog Input	AFS1[3]: 1
	PB4	Reserved		AFS1[4]: 0
		ANA7	ADC Analog Input	AFS1[4]: 1
	PB5	Reserved		AFS1[5]: 0
		VREF	Voltage Reference	AFS1[5]: 1

Notes:

1. Because there are at most two choices of alternate functions for some pins in Ports A and B, the Alternate Function Set Register (AFS2) is implemented but not used to select the function. The alternate function selection must also be enabled, as described in the [Port A–E Alternate Function Subregisters](#) section on page 61.
2. Because there is only one alternate function for each Port D and Port E pin, the Alternate Function Set registers are not implemented for Ports D and E. Enabling the alternate function selections automatically enables the associated alternate function, as described in the [Port A–E Alternate Function Subregisters](#) section on page 61.

One-Shot time-out. First set the TPOL bit in the Timer Control 1 Register to the start value before beginning ONE-SHOT Mode. Then, after starting the timer, set TPOL to the opposite bit value.

Observe the following steps to configure a timer for ONE-SHOT Mode and to initiate the count:

1. Write to the Timer Control 1 Register to:
 - Disable the timer
 - Configure the timer for ONE-SHOT Mode
 - Set the prescale value
 - If using the Timer Output alternate function, set the initial output level (High or Low)
2. Write to the Timer Control 2 Register to choose the timer clock source.
3. Write to the Timer Control 0 Register to set the timer interrupt configuration field TICONFIG.
4. Write to the Timer High and Low Byte registers to set the starting count value.
5. Write to the Timer Reload High and Low Byte registers to set the reload value.
6. If required, enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers.
7. If using the Timer Output function, configure the associated GPIO port pin for the Timer Output alternate function.
8. Write to the Timer Control 1 Register to enable the timer and initiate counting.

In ONE-SHOT Mode, the timer clock always provides the timer input. The timer period is calculated using the following equation:

$$\text{ONE-SHOT Mode Time-Out Period (s)} = \frac{\text{Reload Value} - \text{Start Value}}{\text{Timer Clock Frequency (Hz)}} \times \text{Prescale}$$

9.2.3.2. TRIGGERED ONE-SHOT Mode

In TRIGGERED ONE-SHOT Mode, the timer operates in the following sequence:

1. The Timer idles until a trigger is received. The Timer trigger is taken from the GPIO port pin timer input alternate function. The TPOL bit in the Timer Control 1 Register selects whether the trigger occurs on the rising edge or the falling edge of the timer input signal.
2. Following the trigger event, the Timer counts timer clocks up to the 16-bit reload value stored in the Timer Reload High and Low Byte registers.

9.2.3.10. COMPARE Mode

In COMPARE Mode, the timer counts up to the 16-bit maximum Compare value stored in the Timer Reload High and Low Byte registers. The Timer counts timer clocks up to a 16-bit reload value. Upon reaching the Compare value, the timer generates an interrupt and counting continues (the timer value is not reset to 0001H). Also, if the Timer Output alternate function is enabled, the Timer Output pin changes state (from Low to High or from High to Low) on Compare.

If the Timer reaches FFFFH, the timer rolls over to 0000H and continue counting.

Observe the following steps to configure a timer for COMPARE Mode and initiate the count:

1. Write to the Timer Control 1 Register to:
 - Disable the timer
 - Configure the timer for COMPARE Mode
 - Set the prescale valu.
 - Set the initial logic level (High or Low) for the Timer Output alternate function, if required
2. Write to the Timer Control 2 Register to choose the timer clock source.
3. Write to the Timer Control 0 Register to set the timer interrupt configuration field TICONFIG.
4. Write to the Timer High and Low Byte registers to set the starting count value.
5. Write to the Timer Reload High and Low Byte registers to set the Compare value.
6. If appropriate, enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers.
7. When using the Timer Output function, configure the associated GPIO port pin for the Timer Output alternate function.
8. Write to the Timer Control 1 Register to enable the timer and initiate counting.

In COMPARE Mode, the timer clock always provides the timer input. The Compare time is calculated using the following equation:

9.2.3.11. GATED Mode

In GATED Mode, the timer counts only when the Timer Input signal is in its active state (asserted) as determined by the TPOL bit in the Timer Control 1 Register. When the Timer Input signal is asserted, counting begins. A Timer Interrupt is generated when the Timer Input signal is deasserted or a timer reload occurs. To determine if a Timer Input signal

10.3.3. PWM Output Operation

In a PWM OUTPUT operation, the timer generates a PWM output signal on the channel output pin (T4CHA, T4CHB, T4CHC, or T4CHD). The channel output toggles whenever the timer count matches the channel compare value (defined in the MCTCHyH and MCTCHyL registers). In addition, a channel interrupt is generated and the channel event flag is set in the status register. The timer continues counting according to its programmed mode.

The channel output signal begins with the output value = CHPOL and then transitions to $\overline{\text{CHPOL}}$ when timer value matches the PWM value. If timer mode is Count Modulo, the channel output signal returns to output = CHPOL when timer reaches the reload value and is reset. If timer mode is Count up/down, channel output signal returns to output = CHPOL when the timer count matches the PWM value again (when counting down).

10.3.4. Capture Operation

In a CAPTURE operation, the current timer count is recorded when the selected transition occurs on T4CHA, T4CHB, T4CHC or T4CHD. The Capture count value is written to the Channel High and Low Byte registers. In addition, a channel interrupt is generated and the channel event flag (CHyEF) is set in the Channel Status Register. The CHPOL bit in the Channel Control Register determines if the Capture occurs on a rising edge or a falling edge of the Channel Input signal. The timer continues counting according to the programmed mode.

10.4. Multi-Channel Timer Interrupts

The Multi-Channel Timer provides a single interrupt which has five possible sources. These sources are the internal timer and the four channel inputs (T4CHA, T4CHB, T4CHC, T4CHD).

10.4.1. Timer Interrupt

If enabled by the TCIEN bit of the MCTCTL0 Register, the timer interrupt will be generated when the timer completes a count cycle. This occurs during transition from counter = reload register value to counter = 0 in count modulo mode and occurs during transition from counter = 1 to counter = 0 in count up/down mode.

10.4.2. Capture/Compare Channel Interrupt

A channel interrupt is generated whenever there is a successful Capture/Compare Event on the Timer Channel and the associated CHIEN bit is set.

Bit	Description (Continued)
[5] CHIEN	<p>Channel Interrupt Enable</p> <p>This bit enables generation of channel interrupt. A channel interrupt is generated whenever there is a capture/compare event on the Timer Channel.</p> <p>0 = Channel interrupt is disabled. 1 = Channel interrupt is enabled.</p>
[4] CHUE	<p>Channel Update Enable</p> <p>This bit determines whether writes to the Channel High and Low Byte registers are buffered when TEN = 1. Writes to these registers are not buffered when TEN = 0 regardless of the value of this bit.</p> <p>0 = Writes to the Channel High and Low Byte registers are buffered when TEN = 1 and only take affect on the next end of cycle count. 1 = Writes to the Channel High and Low Byte registers are not buffered when TEN = 1.</p>
[3]	Reserved; must be 0.
[2:0] CHOP	<p>Channel Operation Method</p> <p>This field determines the operating mode of the channel. For a detailed description of the operating modes, see Count Up/Down Mode on page 123.</p> <p>000 = One-Shot Compare operation. 001 = Continuous Compare operation. 010 = PWM Output operation. 011 = Capture operation. 100 – 111 = Reserved.</p>

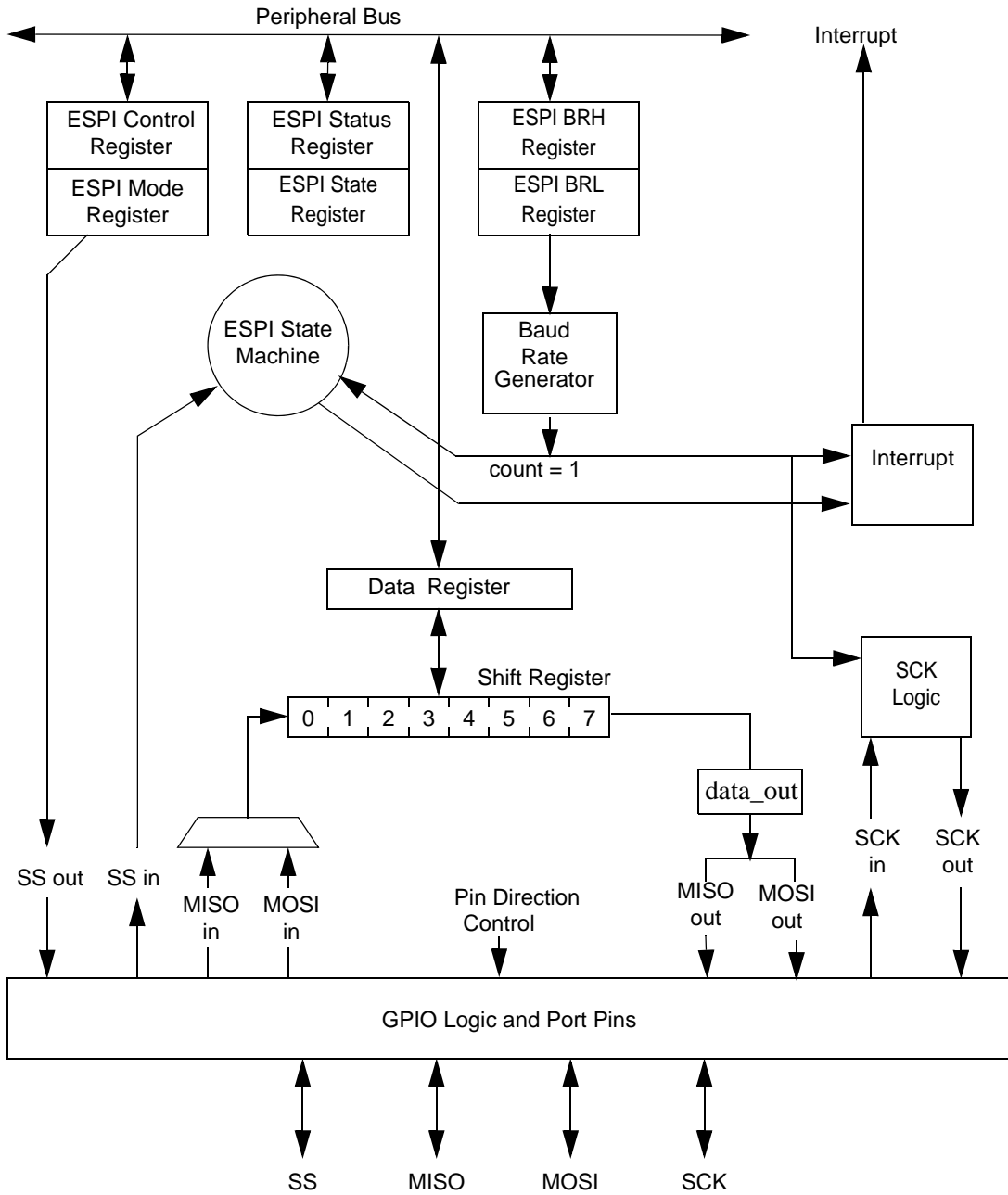


Figure 33. ESPI Block Diagram

be configured with $MMEN = 1$, $SSIO = 0$ (\overline{SS} is an input) and \overline{SS} input = 0. A mode fault sets the COL bit in the ESPI Status Register to 1. Writing a 1 to COL clears this error flag.

16.3.5.3. Receive Overrun

A receive overrun error occurs when a transfer completes and the RDRNE bit is still set from the previous transfer. A receive overrun sets the ROVR bit in the ESPI Status Register to 1. Writing a 1 to ROVR clears this error flag. The Receive Data Register is not overwritten and will contain the data from the transfer which initially set the RDRNE bit. Subsequent received data is lost until the RDRNE bit is cleared.

In SPI MASTER Mode, a receive overrun will not occur. Instead, the SCK will be paused until software responds to the previous RDRNE/TDRE requests.

16.3.5.4. SLAVE Mode Abort

In SLAVE Mode, 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 ESPI Status Register. A Slave abort error resets the Slave control logic to idle state.

A Slave abort error is also asserted in SLAVE Mode, if $BRGCTL = 1$ and a baud rate generator time-out occurs. When $BRGCTL = 1$ in SLAVE Mode, the baud rate generator functions as a Watchdog Timer monitoring the SCK signal. The BRG counter is reloaded every time a transition on SCK occurs while \overline{SS} is asserted. The Baud Rate Reload registers must be programmed with a value longer than the expected time between the \overline{SS} assertion and the first SCK edge, between SCK transitions while \overline{SS} is asserted and between the last SCK edge and \overline{SS} deassertion. A time-out indicates the Master is stalled or disabled. Writing a 1 to ABT clears this error flag.

16.3.6. ESPI Interrupts

ESPI has a single interrupt output which is asserted when any of the TDRE, TUND, COL, ABT, ROVR or RDRNE bits are set in the ESPI Status Register. The interrupt is a pulse which is generated when any one of the source bits initially sets. The TDRE and RDRNE interrupts can be enabled/disabled via the Data Interrupt Request Enable (DIRQE) bit of the ESPI Control Register.

A transmit interrupt is asserted by the TDRE status bit when the ESPI block is enabled and the DIRQE bit is set. The TDRE bit in the Status register is cleared automatically when the Data Register is written or the ESPI block is disabled. After the Data Register is loaded into the Shift Register to start a new transfer, the TDRE bit will be set again, causing a new transmit interrupt. In SLAVE Mode, if information is being received but not transmitted the transmit interrupts can be eliminated by selecting Receive Only mode ($ESPIEN1,0 = 01$). A Master cannot operate in Receive Only mode since a write to the ESPI (Transmit) Data Register is still required to initiate the transfer of a character even if information is being received but not transmitted by the software application.

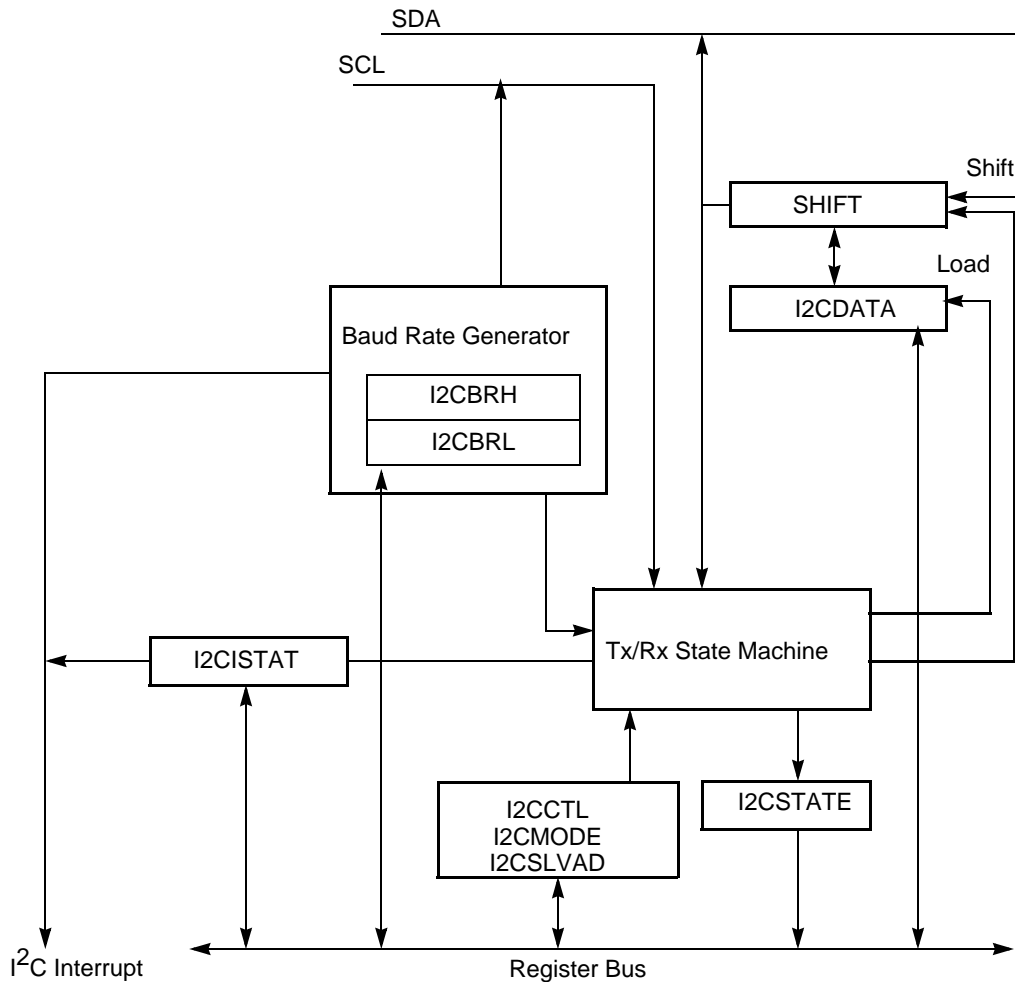


Figure 42. I²C Controller Block Diagram

17.1.1. I²C Master/Slave Controller Registers

Table 118 summarizes the I²C Master/Slave controller's software-accessible registers.

Table 118. I²C Master/Slave Controller Registers

Name	Abbreviation	Description
I ² C Data	I2CDATA	Transmit/Receive Data Register.
I ² C Interrupt Status	I2CISTAT	Interrupt status register.
I ² C Control	I2CCTL	Control Register—basic control functions.

4. If this operation is a single-byte transfer, the software asserts the NAK bit of the I²C Control Register so that after the first byte of data has been read by the I²C controller, a Not Acknowledge instruction is sent to the I²C slave.
5. The I²C controller sends a start condition.
6. The I²C controller sends the address and Read bit out via the SDA signal.
7. The I²C slave acknowledges the address by pulling the SDA signal Low during the next High period of SCL.

If the slave does not acknowledge the address byte, the I²C controller sets the NCKI bit in the I²C Status Register, sets the ACKV bit and clears the ACK bit in the I²C State Register. The software responds to the Not Acknowledge interrupt by setting the stop bit and clearing the TXI bit. The I²C controller flushes the Transmit Data Register, sends a stop condition on the bus and clears the stop and NCKI bits. The transaction is complete and the following steps can be ignored.

8. The I²C controller shifts in the first byte of data from the I²C slave on the SDA signal.
9. The I²C controller asserts the receive interrupt.
10. The software responds by reading the I²C Data Register. If the next data byte is to be the final byte, the software must set the NAK bit of the I²C Control Register.
11. The I²C controller sends a Not Acknowledge to the I²C slave if the next byte is the final byte; otherwise, it sends an Acknowledge.
12. If there are more bytes to transfer, the I²C controller returns to [Step 7](#).
13. A NAK interrupt (NCKI bit in I2CISTAT) is generated by the I²C controller.
14. The software responds by setting the stop bit of the I²C Control Register.
15. A stop condition is sent to the I²C slave.

17.2.5.7. Master Read Transaction with a 10-Bit Address

Figure 46 displays the read transaction format for a 10-bit addressed Slave.

S	Slave Address 1st Byte	W=0	A	Slave Address 2nd Byte	A	S	Slave Address 1st Byte	R=1	A	Data	A	Data	\bar{A}	P
---	---------------------------	-----	---	---------------------------	---	---	---------------------------	-----	---	------	---	------	-----------	---

Figure 46. Data Transfer Format—Master Read Transaction with a 10-Bit Address

The first 7 bits transmitted in the first byte are 11110XX. The two XX bits are the two most-significant bits of the 10-bit address. The lowest bit of the first byte transferred is the write control bit.

Observe the following data transfer procedure for a Read operation to a 10-bit addressed slave:

- d. Set IEN = 1 in the I²C Control Register. Set NAK = 0 in the I²C Control Register.
2. The Master initiates a transfer by sending the address byte. The SLAVE Mode I²C controller finds an address match and detects that the R/W bit = 1 (read by the master from the slave). The I²C controller acknowledges, indicating that it is ready to accept the transaction. The SAM bit in the I2CISTAT Register is set to 1, causing an interrupt. The RD bit is set to 1, indicating a Read from the slave.
3. The software responds to the interrupt by reading the I2CISTAT Register, thereby clearing the SAM bit. Because RD = 1, the software responds by loading the first data byte into the I2CDATA Register. The software sets the TXI bit in the I2CCTL Register to enable transmit interrupts. When the master initiates the data transfer, the I²C controller holds SCL Low until the software has written the first data byte to the I2CDATA Register.
4. SCL is released and the first data byte is shifted out.
5. After the first bit of the first data byte has been transferred, the I²C controller sets the TDRE bit, which asserts the transmit data interrupt.
6. The software responds to the transmit data interrupt (TDRE = 1) by loading the next data byte into the I2CDATA Register, which clears TDRE.
7. After the data byte has been received by the master, the master transmits an Acknowledge instruction (or Not Acknowledge instruction if this byte is the final data byte).
8. The bus cycles through Step 5 to Step 7 until the final byte has been transferred. If the software has not yet loaded the next data byte when the master brings SCL Low to transfer the most significant data bit, the slave I²C controller holds SCL Low until the Data Register has been written. When a Not Acknowledge instruction is received by the slave, the I²C controller sets the NCKI bit in the I2CISTAT Register causing the Not Acknowledge interrupt to be generated.
9. The software responds to the Not Acknowledge interrupt by clearing the TXI bit in the I2CCTL Register and by asserting the FLUSH bit of the I2CCTL Register to *empty* the Data Register.
10. When the Master has completed the final acknowledge cycle, it asserts a stop or restart condition on the bus.
11. The Slave I²C controller asserts the stop/restart interrupt (set SPRS bit in I2CISTAT Register).
12. The software responds to the stop/restart interrupt by reading the I2CISTAT Register, which clears the SPRS bit.

17.2.6.8. Slave Transmit Transaction With 10-Bit Address

The data transfer format for a master reading data from a slave with 10-bit addressing is displayed in Figure 50. The following procedure describes the I²C Master/Slave Controller operating as a slave in 10-bit addressing mode, transmitting data to the bus master.

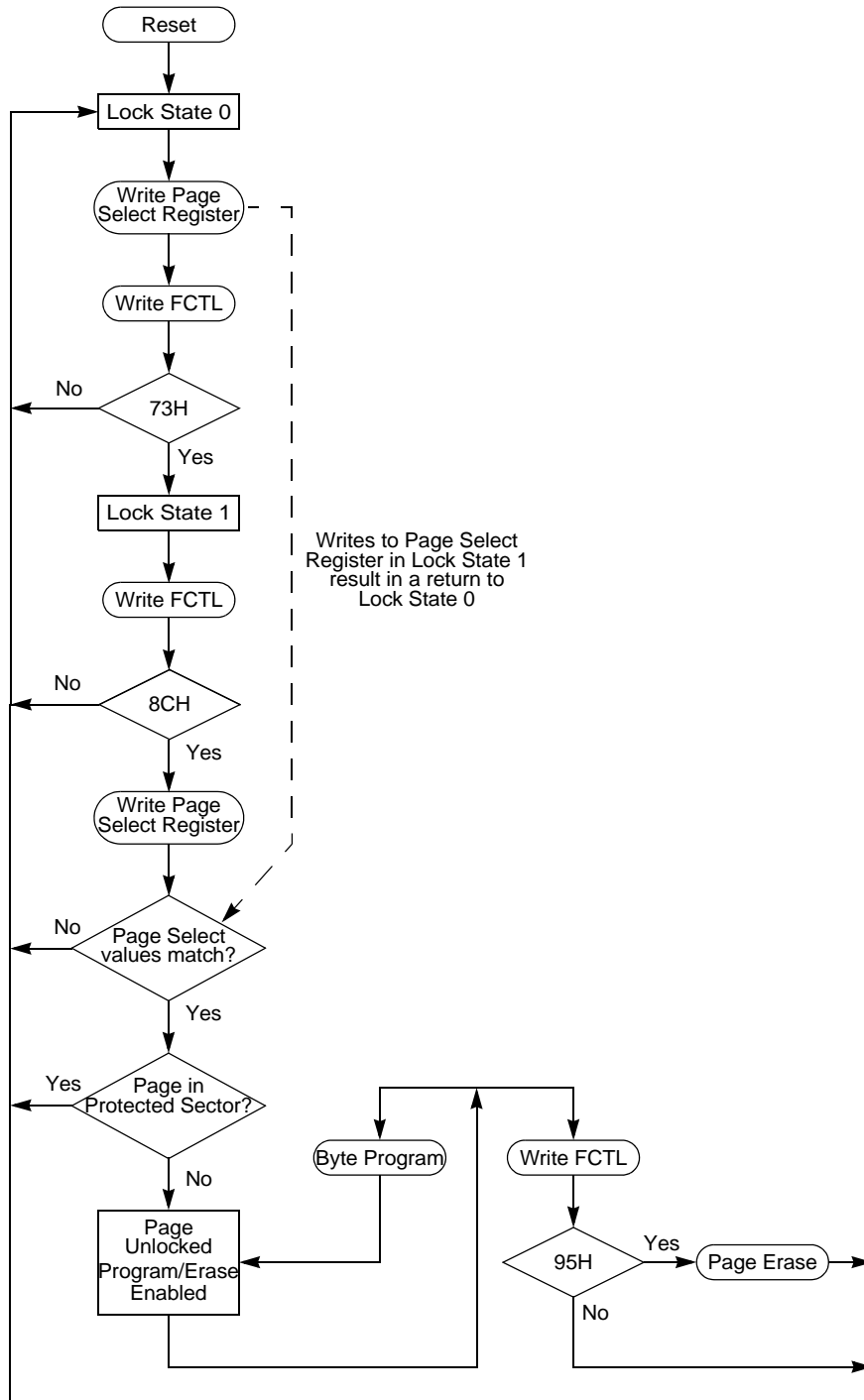


Figure 54. Flowchart: Flash Controller Operation

20.3.4. Flash Sector Protect Register

The Flash Sector Protect Register is shared with the Flash Page Select Register. When the Flash Control Register (see page 271) is written with 5EH, the next write to this address targets the Flash Sector Protect Register. In all other cases, it targets the Flash Page Select Register.

This register selects one of the eight available Flash memory sectors to be protected. The reset state of each Sector Protect bit is an unprotected state. After a sector is protected by setting its corresponding register bit, it can only be unprotected (the register bit can only be cleared) by a System Reset. Please refer to Table 132 on page 262 and to Figures 51 through 53 to review how Flash memory is arranged by sector.

Table 137. Flash Sector Protect Register (FPROT)

Bits	7	6	5	4	3	2	1	0
Field	SPROT7	SPROT6	SPROT5	SPROT4	SPROT3	SPROT2	SPROT1	SPROT0
Reset	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Address	FF9H							

Bit	Description
[7:0]	Sector Protection
SPROT _x	<ul style="list-style-type: none"> On Z8F2480 devices, each bit corresponds to a 3KB Flash sector. On Z8F1680 devices, each bit corresponds to a 2KB Flash sector. On Z8F0880 devices, each bit corresponds to a 1KB Flash sector.

20.3.5. Flash Frequency High and Low Byte Registers

The Flash Frequency High and Low Byte registers, shown in Tables 138 and 139, combine to form a 16-bit value, FFREQ, to control timing for Flash program and erase operations. The 16-bit binary Flash Frequency value must contain the system clock frequency (in kHz) and is calculated using the following equation:

$$FFREQ[15:0] = \{FFREQH[7:0], FFREQL[7:0]\} = \frac{\text{System Clock Frequency}}{1000}$$

! Caution: Flash programming and erasure is not supported for system clock frequencies below 32 kHz or above 20MHz. The Flash Frequency High and Low Byte registers must be loaded with the correct values to ensure proper operation of the device.

Table 147. Trim Option Bits at 0001H (TTEMP1)

Bits	7	6	5	4	3	2	1	0
Field	TS_NEG				TS_COARSE			
Reset	1	0	1	0	0	1	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Address	Information Page Memory 0021H							

Note: U = Unchanged by Reset. R/W = Read/Write.

Bit	Description
[7:4] TS_NEG	Temperature Sensor Negative Control Trim Bits Negative control offset trimming bits for the Temperature Sensor.
[3:0] TS_COARSE	Temperature Sensor Coarse Control Trim Bits Contains coarse control offset trimming bits for the Temperature Sensor.

21.2.4.2. Trim Bit Address 0002H

The Trim Option Bits Register at address 0002H, shown in Table 148, governs control of the Internal Precision Oscillator trim bits.

Table 148. Trim Option Bits at 0002H (TIPO)

Bits	7	6	5	4	3	2	1	0
Field	IPO_TRIM							
Reset	U							
R/W	R/W							
Address	Information Page Memory 0022H							

Note: U = Unchanged by Reset. R/W = Read/Write.

Bit	Description
[7:0] IPO_TRIM	Internal Precision Oscillator Trim Byte Contains trimming bits for Internal Precision Oscillator.

the return from the subroutine, the read byte resides in working register R0 and the read status byte resides in working register R1. The bit fields of this status byte are defined in Table 160. Also, the user code should pop the address byte off the stack.

The read routine uses 16 bytes of stack space in addition to the 1 byte of address pushed by you. Sufficient memory must be available for this stack usage. Because of the Flash memory architecture, NVDS reads exhibit a nonuniform execution time. A read operation takes between 71 μs and 258 μs (assuming a 20 MHz system clock). Slower system clock speeds result in proportionally higher execution times.

NVDS byte reads from invalid addresses (those exceeding the NVDS array size) return 0xff. Illegal read operations have a 6 μs execution time. The status byte returned by the NVDS read routine is zero for successful read. If the status byte is nonzero, there is a corrupted value in the NVDS array at the location being read. In this case, the value returned in R0 is the byte most recently written to the array that does not have an error.

Table 160. Read Status Byte

Bits	7	6	5	4	3	2	1	0
Field	Reserved			DE	Reserved	FE	IGADDR	Reserved
Default Value	0	0	0	0	0	0	0	0

Bit	Description
[7:5]	Reserved; must be 0.
[4] DE	Data Error When reading a NVDS address, if an error is found in the latest data corresponding to the NVDS address, this bit is set to 1. NVDS source code steps forward until it finds valid data at this address.
[3]	Reserved; must be 0.
[2] FE	Flash Error If a Flash error is detected, this bit is set to 1.
[1] IGADDR	Illegal Address When NVDS byte reads occur from invalid addresses (those exceeding the NVDS array size), this bit is set to 1. Note: When the NVDS array size is 256 bytes, there is no address exceeding the size: therefore the IGADDR bit cannot be used.
[0]	Reserved; must be 0.

22.2.3. Power Failure Protection

The NVDS routines employ error checking mechanisms to ensure a power failure endangers only the most recently written byte. Bytes previously written to the array are not perturbed. For this protection to function, the VBO must be enabled (see the [Low-Power](#)

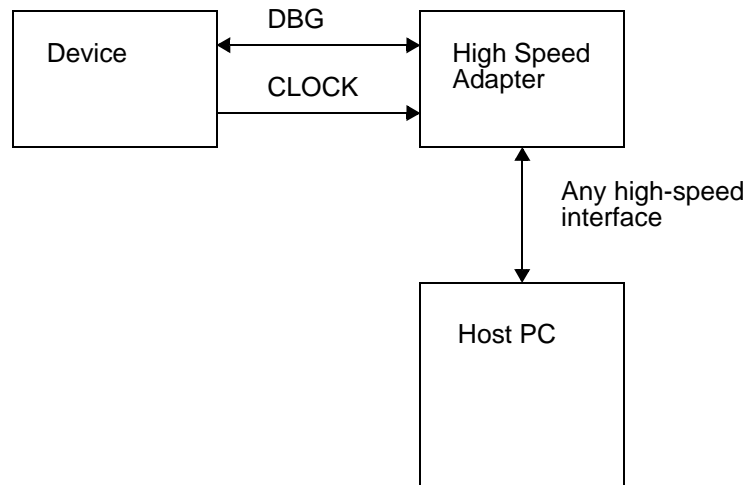


Figure 60. Synchronous Operation

23.2.5. OCD Serial Errors

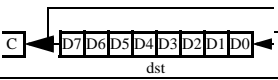
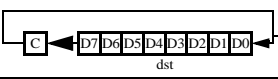

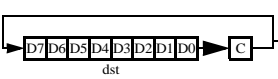
The On-Chip Debugger can detect any of the following error conditions on the DBG pin:

- Serial Break (a minimum of ten continuous bits Low)
- Framing Error (received stop bit is Low)
- Transmit Collision (OCD and host simultaneous transmission detected by the OCD)

When the OCD detects one of these errors, it aborts any command currently in progress, transmits a Serial Break 4096 system clock cycles long back to the host and resets the Autobaud Detector/Generator. A Framing Error or Transmit Collision can be caused by the host sending a Serial Break to the OCD. Because of the open-drain nature of the interface, returning a Serial Break back to the host only extends the length of the Serial Break if the host releases the Serial Break early.

The host transmits a Serial Break on the DBG pin when first connecting to the Z8 Encore! XP F1680 Series device or when recovering from an error. A Serial Break from the host resets the Autobaud Generator/Detector but does not reset the OCD Control Register. A Serial Break leaves the device in DEBUG mode if that is the current mode. The OCD is held in Reset until the end of the Serial Break when the DBG pin returns High. Because of the open-drain nature of the DBG pin, the host can send a Serial Break to the OCD even if the OCD is transmitting a character.

Table 186. eZ8 CPU Instruction Summary (Continued)

Assembly Mnemonic	Symbolic Operation	Address Mode		Op Code(s) (Hex)	Flags						Fetch Cycles	Instr. Cycles
		dst	src		C	Z	S	V	D	H		
POPX dst	dst ← @SP SP ← SP + 1	ER		D8	-	-	-	-	-	-	3	2
PUSH src	SP ← SP - 1 @SP ← src	R		70	-	-	-	-	-	-	2	2
		IR		71							2	3
		IM		IF70							3	2
PUSHX src	SP ← SP - 1 @SP ← src	ER		C8	-	-	-	-	-	-	3	2
RCF	C ← 0			CF	0	-	-	-	-	-	1	2
RET	PC ← @SP SP ← SP + 2			AF	-	-	-	-	-	-	1	4
RL dst		R		90	*	*	*	*	-	-	2	2
		IR		91							2	3
RLC dst		R		10	*	*	*	*	-	-	2	2
		IR		11							2	3
RR dst		R		E0	*	*	*	*	-	-	2	2
		IR		E1							2	3
RRC dst		R		C0	*	*	*	*	-	-	2	2
		IR		C1							2	3
SBC dst, src	dst ← dst - src - C	r	r	32	*	*	*	*	1	*	2	3
		r	lr	33							2	4
		R	R	34							3	3
		R	IR	35							3	4
		R	IM	36							3	3
		IR	IM	37							3	4
SBCX dst, src	dst ← dst - src - C	ER	ER	38	*	*	*	*	1	*	4	3
		ER	IM	39							4	3

Flags notation:

* = Value is a function of the result of the operation.

- = Unaffected.

X = Undefined.

0 = Reset to 0.

1 = Set to 1.

Table 201. Low Voltage Detect Electrical Characteristics (Continued)

		T _A = 0°C to +70°C T _A = -40°C to +105°C				
		V _{DD} = 1.8 to 3.6 V				
Symbol	Parameter	Min	Typ	Max	Units	Conditions
V _{TH_PRO}	Detected Source Voltage for Flash Protection	2.4	2.5	2.6	V	
T _{DELAY}	Delay from source voltage falling lower than V _{TP} to I _{VD_OUT} output logic High	50	1000	–	ns	

Note: ¹ V_{TP} is a user-set threshold voltage to be detected.

Table 202. Crystal Oscillator Characteristics

		T _A = 0°C to +70°C T _A = -40°C to +105°C							
		V _{DD} = 2.7 to 3.6 V			V _{DD} = 1.8 to 2.7 V				
Symbol	Parameter	Min	Typ	Max	Min	Typ	Max	Units	Conditions
I _{DDXTAL}	Crystal Oscillator Active Supply Current	–	–	500	–	–	300	μA	
I _{DDQXTAL}	Crystal Oscillator Quiescent Current	–	5	–	–	5	–	nA	
S _{CLK}	Clk_out State in Crystal Disable	1	1	1	1	1	1		
F _{XTAL}	External Crystal Oscillator Frequency	1	–	20	1	–	20	MHz	See Figure 74.
T _{SET}	Startup Time After Enable	–	10,000	30,000	–	10,000	30,000	Cycle	
	Clk_out Duty Cycle	40	50	60	40	50	60	%	
	Clk_out Jitter	–	1	–	–	1	–	%	

control register, I2C 247
Control Registers 19
CP 332
CPC 332
CPCX 332
CPU and peripheral overview 4
CPU control instructions 333
CPX 332
current measurement
 architecture 186
 operation 186
Customer Feedback Form 387

D

DA 330, 332
data memory 21
data register, I2C 243
DC characteristics 350
debugger, on-chip 294
DEC 332
decimal adjust 332
decrement 332
decrement and jump non-zero 335
decrement word 332
DECW 332
destination operand 331
device, port availability 46
DI 333
direct address 330
disable interrupts 333
DJNZ 335
dst 331

E

EI 333
electrical characteristics 349
 ADC 360
 flash memory and timing 359
 GPIO input data sample timing 366
 watch-dog timer 359, 361
electrical noise 186
enable interrupt 333
ER 330
extended addressing register 330

external pin reset 37
eZ8 CPU features 4
eZ8 CPU instruction classes 331
eZ8 CPU instruction notation 330
eZ8 CPU instruction set 328
eZ8 CPU instruction summary 336

F

FCTL register 272, 281
features, Z8 Encore! 1
first opcode map 347
FLAGS 331
flags register 331
flash
 controller 4
 option bit configuration - reset 276
flash memory 262
 arrangement 263, 264, 265
 byte programming 269
 code protection 267
 configurations 262
 control register definitions 271, 278
 controller bypass 270
 electrical characteristics and timing 359
 flash control register 272, 281
 flash option bits 268
 flash status register 272
 flow chart 266
 frequency high and low byte registers 274
 mass erase 270
 operation 265
 operation timing 267
 page erase 270
 page select register 273, 274
FPS register 273, 274
FSTAT register 272

G

gated mode 114
general-purpose I/O 46
GPIO 4, 46
 alternate functions 47
 architecture 47
 control register definitions 58