**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 | I²C, 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 | 8KB (8K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 2K x 8 |
| Voltage - Supply (Vcc/Vdd) | 1.8V ~ 3.6V |
| Data Converters | A/D 8x10b |
| Oscillator Type | Internal |
| Operating Temperature | 0°C ~ 70°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 28-SSOP (0.209", 5.30mm Width) |
| Supplier Device Package | - |
| Purchase URL | https://www.e-xfl.com/product-detail/zilog/z8f0880hj020sg |

## 1.5. Acronyms and Expansions

This document uses the acronyms and expansions listed in Table 2.

**Table 2. F1680 Series MCU Acronyms**

| Abbreviations/ Acronyms | Expansions |
|---|---|
| ADC | Analog-to-Digital Converter |
| NVDS | Non-Volatile Data Storage |
| LPO | Low-Power Operational Amplifier |
| LIN | Local Interconnect Network |
| SPI | Serial Peripheral Interface |
| ESPI | Enhanced Serial Peripheral Interface |
| WDT | Watchdog Timer |
| GPIO | General-Purpose Input/Output |
| OCD | On-Chip Debugger |
| POR | Power-On Reset |
| LVD | Low-Voltage Detection |
| VBO | Voltage Brown-Out |
| IPO | Internal Precision Oscillator |
| UART | Universal Asynchronous Receiver/Transmitter |
| IrDA | Infrared Data Association |
| $I^2C$ | Inter-integrated circuit |
| PDIP | Plastic Dual Inline Package |
| SOIC | Small Outline Integrated Circuit |
| SSOP | Small Shrink Outline Package |
| QFN | Quad Flat No Lead |
| LQFP | Low-Profile Quad Flat Package |
| PRAM | Program RAM |
| PC | Program counter |
| IRQ | Interrupt request |
| ISR | Interrupt service routine |
| MSB | Most-significant byte |

## 7.4. Direct LED Drive

The Port C pins provide a current synchronized output capable of driving an LED without requiring an external resistor. The output synchronizes current at programmable levels of 3mA, 7mA, 13mA and 20mA. This mode is enabled through the Alternate Function sub-register AFS1 and is programmable through the LED control registers. For proper function, the LED anode must be connected to $V_{DD}$ and the cathode to the GPIO pin.

Using all Port C pins in LED drive mode with maximum current can result in excessive total current. For the maximum total current for the applicable package, see the Electrical Characteristics chapter on page 349.

## 7.5. Shared Reset Pin

On all the devices, the Port D0 pin shares function with a bidirectional reset pin. Unlike all other I/O pins, this pin does not default to GPIO pin on power-up. This pin acts as a bidirectional input/open-drain output reset with an internal pull-up until user software reconfigures it as GPIO PD0. The Port D0 pin is output-only when in GPIO Mode, and must be configured as an output. PD0 supports the High Drive feature but not the Stop Mode Recovery feature.

## 7.6. Crystal Oscillator Override

For systems using the crystal oscillator, PA0 and PA1 is used to connect the crystal. When the main crystal oscillator is enabled (see the Oscillator Control1 Register section on page 320), the GPIO settings are overridden and PA0 and PA1 is disabled.

## 7.7. 32kHz Secondary Oscillator Override

For systems using a 32kHz secondary oscillator, PA2 and PA3 is used to connect a watch crystal. When the 32kHz secondary oscillator is enabled (see the Oscillator Control1 Register section on page 320), the GPIO settings are overridden and PA2 and PA3 is disabled.

## 7.8. 5V Tolerance

All GPIO pins, including those that share functionality with an ADC, crystal or comparator signals are 5V-tolerant and can handle inputs higher than $V_{DD}$ even with the pull-ups enabled.

If the TPOL bit in the Timer Control 1 Register is set to 1, the Timer Output signal begins as High (1) and then transitions to Low (0) when the timer value matches the PWM value. The Timer Output signal returns to High (1) after the timer reaches the reload value and is reset to 0001H.

If the TPOL bit in the Timer Control 1 Register is set to 0, the Timer Output signal begins as Low (0) and then transitions to High (1) when the timer value matches the PWM value. The Timer Output signal returns to Low (0) after the timer reaches the reload value and is reset to 0001H.

Observe the following steps to configure a timer for PWM SINGLE OUTPUT Mode and initiate PWM operation:

1. Write to the Timer Control 1 Register to:
   - Disable the timer
   - Configure the timer for PWM mode
   - Set the prescale value
   - Set the initial logic level (High or Low) and PWM High/Low transition for the Timer Output Alternate Function
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 (typically 0001H). This value only affects the first pass in PWM mode. After the first timer reset in PWM mode, counting always begins at the reset value of 0001H.
5. Write to the Timer PWM0 High and Low Byte registers to set the PWM value.
6. Write to the Timer Reload High and Low Byte registers to set the reload value (PWM period). The reload value must be greater than the PWM value.
7. If appropriate, enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers.
8. Configure the associated GPIO port pin for the Timer Output alternate function.
9. Write to the Timer Control 1 Register to enable the timer and initiate counting.

The PWM period is calculated using the following equation:

If an initial starting value other than 0001H is loaded into the Timer High and Low Byte registers, the ONE-SHOT Mode equation must be used to determine the first PWM time-out period.

**Table 60. Timer 0–2 PWM0 Low Byte Register (TxPWM0L)**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | PWM0L | | | | | | | |
| 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 | F05H, F0DH, F15H | | | | | | | |

| Bit | Description |
|---|---|
| [7:0] PWM0H, PWM0L | **Pulse Width Modulator 0 High and Low Bytes**<br>These two bytes, {PWM0H[7:0], PWM0L[7:0]}, form a 16-bit value that is compared to the current 16-bit timer count. When a match occurs, the PWM output changes state. The PWM output value is set by the TPOL bit in the Timer Control 1 Register (TxCTL1).<br>The TxPWM0H and TxPWM0L registers also store the 16-bit captured timer value when operating in CAPTURE, CAPTURE/COMPARE and DEMODULATION Modes. |

## 9.3.4. Timer 0–2 PWM1 High and Low Byte Registers

The Timer 0–2 PWM1 High and Low Byte (TxPWM1H and TxPWM1L) registers, shown in Tables 61 and 62, store Capture values for DEMODULATION Mode.

**Table 61. Timer 0-2 PWM1 High Byte Register (TxPWM1H)**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | PWM1H | | | | | | | |
| 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 | F20H, F24H, F28H | | | | | | | |

**Table 62. Timer 0–2 PWM1 Low Byte Register (TxPWM1L)**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | PWM1L | | | | | | | |
| 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 | F21H, F25H, F29H | | | | | | | |

| Bit | Description |
|---|---|
| [7:0] PWM1H, PWM1L | **Pulse Width Modulator 1 High and Low Bytes**<br>These two bytes, {PWM1H[7:0], PWM1L[7:0]}, store the 16-bit captured timer value for DEMODULATION Mode. |

# 10.3. Capture/Compare Channel Operation

The Multi-Channel timer supports four Capture/Compare channels: CHA, CHB, CHC and CHD. Each channel has the following features:

- A 16-bit Capture/Compare Register (MCTCHyH and MCTCHyL registers) used to capture input event times or to generate time intervals. Any user software update of the Capture/Compare Register value when the timer is running takes effect only at the end of the counting cycle, not immediately. The end of the counting cycle is when the counter transitions from the reload value to 0 (count modulo mode) or from 1 to 0 (count up/down mode).
- A dedicated bidirectional pin (T4CHA, T4CHB, T4CHC, or T4CHD) that can be configured for the input capture function or to generate an output compare match or one-shot pulse.

Each channel is configured to operate in ONE-SHOT COMPARE, CONTINUOUS COMPARE, PWM OUTPUT, or INPUT CAPTURE mode.

## 10.3.1. One-Shot Compare Operation

In a ONE-SHOT COMPARE operation, a channel interrupt is generated when the channel compare value matches the timer count. The channel event flag (CHyEF) is set in the Channel Status 1 Register (MCTCHS1) to identify the responsible channel. The channel is then automatically disabled. The timer continues counting according to the programmed mode. If the timer channel output alternate function is enabled, the channel output pin (T4CHA, T4CHB, T4CHC, or T4CHD) changes state for one system clock cycle upon match (i.e., from Low to High, then back to Low or High to Low, then back to High as determined by the CHPOL bit).

## 10.3.2. Continuous Compare Operation

In a CONTINUOUS COMPARE operation, a channel interrupt is generated when the channel compare value matches the timer count. The channel event flag (CHyEF) is set in the Channel Status 1 Register (MCTCHS1) and the channel remains enabled. The timer continues counting according to the programmed mode. If the channel output alternate function is enabled, the channel output pin (T4CHA, T4CHB, T4CHC, or T4CHD) changes state upon match (i.e., from Low to High then back to Low; or High to Low then back to High, as determined by the CHPOL bit).

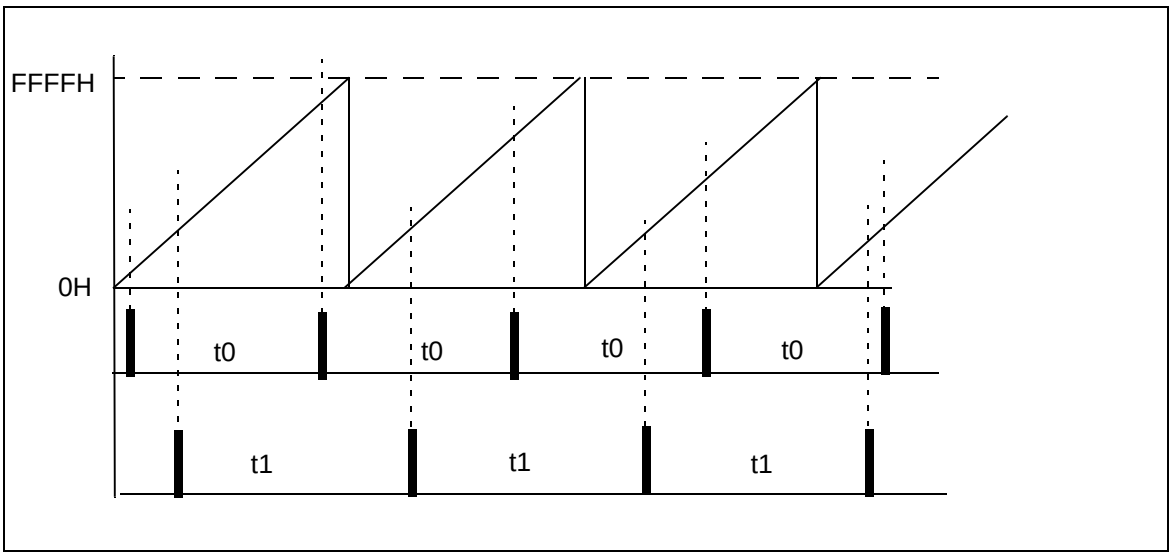


**Figure 18. Count Max Mode with Channel Compare**

# 10.7. Multi-Channel Timer Control Register Definitions

This section defines the features of the following Multi-Channel Timer Control registers.

Multi-Channel Timer High and Low Byte Registers: see page 130

Multi-Channel Timer Reload High and Low Byte Registers: see page 130

Multi-Channel Timer Subaddress Register: see page 131

Multi-Channel Timer Subregister x (0, 1, or 2): see page 132

Multi-Channel Timer Control 0, Control 1 Registers: see page 132

Multi-Channel Timer Channel Status 0 and Status 1 Registers: see page 135

Multi-Channel Timer Channel-y Control Registers: see page 137

Multi-Channel Timer Channel-y High and Low Byte Registers: see page 139

## 10.7.1. Multi-Channel Timer Address Map

Table 69 defines the byte address offsets for the Multi-channel Timer registers. For saving address space, a subaddress is used for the Timer Control 0, Timer Control 1, Channel Status 0, Channel Status 1, Channel-y Control, and Channel-y High and Low byte registers. Only the Timer High and Low Byte registers and the Reload High and Low Byte registers can be directly accessed.

## 12.3.2. LIN-UART Receive Data Register

Data bytes received through the RxD pin are stored in the LIN-UART Receive Data Register as shown in Table 84. The read-only LIN-UART Receive Data Register shares a Register File address with the write-only LIN-UART Transmit Data Register.

**Table 84. LIN-UART Receive Data Register (U0RXD = F40H)**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | RxD | | | | | | | |
| Reset | X | X | X | X | X | X | X | X |
| R/W | R | R | R | R | R | R | R | R |
| Address | F40H, F48H | | | | | | | |

Note: R = Read.

| Bit | Description |
|---|---|
| [7:0] | **Receive Data** |
| RxD | LIN–UART receiver data byte from the RxD pin. |

## 14.3.5. Sample Settling Time Register

The Sample Settling Time Register, shown in Table 105, is used to program the length of time from the SAMPLE/HOLD signal to the start signal, when the conversion can begin. The number of clock cycles required for settling will vary from system to system depending on the system clock period used. The system designer should program this register to contain the number of clocks required to meet a 0.5µs minimum settling time.

**Table 105. Sample Settling Time (ADCSST)**

| Bits | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Reserved | | | | SST | | | |
| Reset | 0 | | | | 1 | 1 | 1 | 1 |
| R/W | R | | | | R/W | | | |
| Address | F74H | | | | | | | |

| Bit Position | Value (H) | Description |
|---|---|---|
| [7:4] | 0 | Reserved; must be 0. |
| [3:0] SST | 0–F | Sample settling time in number of system clock periods to meet 0.5µs minimum. |

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.

enabled when running in I2C FAST Mode (400 Kbps) and can also be used at lower data rates.

## 17.2.2. I2C Interrupts

The I2C controller contains multiple interrupt sources that are combined into one interrupt request signal to the interrupt controller. If the I2C controller is enabled, the source of the interrupt is determined by which bits are set in the I2CISTAT Register. If the I2C controller is disabled, the BRG controller is used to generate general-purpose timer interrupts.

Each interrupt source, other than the baud rate generator interrupt, features an associated bit in the I2CISTAT Register that clears automatically when software reads the register or performs another task, such as reading/writing the Data Register.

### 17.2.2.1. Transmit Interrupts

Transmit interrupts (TDRE bit = 1 in I2CISTAT) occur under the following conditions, both of which must be true:

- The Transmit Data Register is empty and the TXI bit = 1 in the I2C Control Register.
- The I2C controller is enabled with one of the following elements:
  - The first bit of a 10-bit address is shifted out.
  - The first bit of the final byte of an address is shifted out and the RD bit is deasserted.
  - The first bit of a data byte is shifted out.

Writing to the I2C Data Register always clears the TRDE bit to 0.

### 17.2.2.2. Receive Interrupts

Receive interrupts (RDRF bit = 1 in I2CISTAT) occur when a byte of data has been received by the I2C controller. The RDRF bit is cleared by reading from the I2C Data Register. If the RDRF interrupt is not serviced prior to the completion of the next Receive byte, the I2C controller holds SCL Low during the final data bit of the next byte until RDRF is cleared, to prevent receive overruns. A receive interrupt does not occur when a Slave receives an address byte or for data bytes following a slave address that do not match. An exception is if the Interactive Receive Mode (IRM ) bit is set in the I2CMODE Register, in which case Receive interrupts occur for all Receive address and data bytes in SLAVE Mode.

### 17.2.2.3. Slave Address Match Interrupts

Slave address match interrupts (SAM bit = 1 in I2CISTAT) occur when the I2C controller is in SLAVE Mode and an address received matches the unique slave address. The General Call Address (`0000_0000`) and STARTBYTE (`0000_0001`) are recognized if the

14. The software responds by writing the data to be written out to the $I^2C$ Control Register.
15. The $I^2C$ controller shifts out the remainder of the second byte of the slave address (or ensuring data bytes, if looping) via the SDA signal.
16. The $I^2C$ slave sends an Acknowledge by pulling the SDA signal Low during the next High period of SCL. The $I^2C$ controller sets the ACK bit in the $I^2C$ Status Register. If the slave does not acknowledge, see the second paragraph of Step 11.
17. The $I^2C$ controller shifts the data out by the SDA signal. After the first bit is sent, the transmit interrupt asserts.
18. If more bytes remain to be sent, return to Step 14.
19. The software responds by asserting the stop bit of the $I^2C$ Control Register.
20. The $I^2C$ controller completes transmission of the data on the SDA signal.
21. The $I^2C$ controller sends a stop condition to the $I^2C$ bus.

**Note:** If the slave responds with a Not Acknowledge during the transfer, the $I^2C$ controller asserts the NCKI bit, sets the ACKV bit, clears the ACK bit in the $I^2C$ State Register and halts. The software terminates the transaction by setting either the stop bit (end transaction) or the start bit (end this transaction, start a new one). The Transmit Data Register is flushed automatically.

### 17.2.5.6. Master Read Transaction with a 7-Bit Address

Figure 45 displays the data transfer format for a Read operation to a 7-bit addressed slave.

| S | Slave Address | R = 1 | A | Data | A | Data | A | P/S |
|---|---|---|---|---|---|---|---|---|

**Figure 45. Data Transfer Format—Master Read Transaction with a 7-Bit Address**

Observe the following steps for a Master Read operation to a 7-bit addressed slave:

1. The software initializes the MODE field in the $I^2C$ Mode Register for MASTER/SLAVE Mode with 7- or 10-bit addressing (the $I^2C$ bus protocol allows the mixing of slave address types). The MODE field selects the address width for this mode when addressed as a slave (but not for the remote slave). The software asserts the IEN bit in the $I^2C$ Control Register.
2. The software writes the $I^2C$ Data Register with a 7-bit slave address, plus the Read bit (which is set to 1).
3. The software asserts the start bit of the $I^2C$ Control Register.

5. The $I^2C$ controller receives the data byte and responds with Acknowledge or Not Acknowledge depending on the state of the NAK bit in the I2CCTL Register. The $I^2C$ controller generates the receive data interrupt by setting the RDRF bit in the I2CISTAT Register.
6. The software responds by reading the I2CISTAT Register, finding the RDRF bit = 1 and reading the I2CDATA Register clearing the RDRF bit. If software can accept only one more data byte it sets the NAK bit in the I2CCTL Register.
7. The master and slave loops through Step 4 to Step 6 until the master detects a Not Acknowledge instruction or runs out of data to send.
8. The master sends the stop or restart signal on the bus. Either of these signals can cause the $I^2C$ controller to assert a stop interrupt (the stop bit = 1 in the I2CISTAT Register). Because the slave received data from the master, the software takes no action in response to the stop interrupt other than reading the I2CISTAT Register to clear the stop bit in the I2CISTAT Register.

### 17.2.6.6. Slave Receive Transaction with 10-Bit Address

The data transfer format for writing data from a master to a slave with 10-bit addressing is displayed in Figure 48. The procedure that follows describes the $I^2C$ Master/Slave Controller operating as a slave in 10-bit addressing mode and receiving data from the bus master.

| S | Slave Address 1st Byte | W=0 | A | Slave Address 2nd Byte | A | Data | A | Data | A/$\overline{A}$ | P/S |
|---|---|---|---|---|---|---|---|---|---|---|

**Figure 48. Data Transfer Format—Slave Receive Transaction with 10-Bit Address**

1. The software configures the controller for operation as a slave in 10-bit addressing mode, as follows:
   a. Initialize the MODE field in the I2CMODE Register for either SLAVE ONLY mode or MASTER/SLAVE Mode with 10-bit addressing.
   b. Optionally set the GCE bit.
   c. Initialize the SLA[7:0] bits in the I2CSLVAD Register and the SLA[9:8] bits in the I2CMODE Register.
   d. Set IEN = 1 in the I2CCTL Register. Set NAK = 0 in the $I^2C$ Control Register.
2. The Master initiates a transfer, sending the first address byte. The $I^2C$ controller recognizes the start of a 10-bit address with a match to SLA[9:8] and detects R/$\overline{W}$ bit = 0 (a Write from the master to the slave). The $I^2C$ controller acknowledges, indicating it is available to accept the transaction.
3. The Master sends the second address byte. The SLAVE Mode $I^2C$ controller detects an address match between the second address byte and SLA[7:0]. The SAM bit in the

## 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] SPROT*x* | **Sector Protection**<br>• On Z8F2480 devices, each bit corresponds to a 3KB Flash sector.<br>• On Z8F1680 devices, each bit corresponds to a 2KB Flash sector.<br>• 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 20 MHz. The Flash Frequency High and Low Byte registers must be loaded with the correct values to ensure proper operation of the device.

## 21.2.5. Zilog Calibration Option Bits

This section describes the calibration of the temperature sensor's Low and High bytes.

### 21.2.5.1. Temperature Sensor Calibration High and Low Byte Registers

Tables 157 and 158 present the Temperature Sensor Calibration High and Low Byte registers.

**Table 157. Temperature Sensor Calibration High Byte at FE60H (TEMPCALH)**

| Bits | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | TEMPCALH | | | | | | | |
| Reset | U | U | U | U | U | U | U | U |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Address | Information Page Memory FE60H | | | | | | | |

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

| Bit | Description |
|---|---|
| [7:0] TEMPCALH | **Temperature Sensor Calibration High Byte**<br>Bits [7:3] of this register are not used. Bit 2 indicates whether the calibration data is added to or subtracted from the measured ADC data. If bit 2 is 0, the calibration data is added; if bit 2 is 1, the calibration data is subtracted.<br>Bits 1 and 0 are the High two bits of the 10-bit calibration data. |

**Table 158. Temperature Sensor Calibration Low Byte at FE61H (TEMPCALL)**

| Bits | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | TEMPCALL | | | | | | | |
| Reset | U | U | U | U | U | U | U | U |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Address | Information Page Memory FE61H | | | | | | | |

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

| Bit | Description |
|---|---|
| [7:0] TEMPCALL | **Temperature Sensor Calibration Low Byte**<br>TEMPCALL is the low eight bits of 10-bit calibration data. The entire 10-bit calibration data field is {TEMPCALH[1:0], TEMPCALL}. |

**Table 162. OCD Baud-Rate Limits**

| System Clock Frequency | Maximum Asynchronous Baud Rate (bits/s) | Minimum Baud Rate (bits/s) |
|---|---|---|
| 20.0 MHz | 2.5 M | 39.1 k |
| 1.0MHz | 125 k | 1.96K |
| 32kHz | 4096 | 64 |

If the OCD receives a Serial Break (ten or more continuous bits Low), the Autobaud Detector/Generator resets. The Autobaud Detector/Generator can then be reconfigured by sending 80H. If the Autobaud Detector overflows while measuring the Autobaud character, the Autobaud Detector will remain reset.

## 23.2.4. High Speed Synchronous

It is possible to operate the serial On-Chip Debugger at high speeds. To operate at high speeds, data must be synchronized with an external clock. High speed synchronous communication will only work when using an external clock source. To operate in high-speed synchronous mode, simply Autobaud to the appropriate speed. The Autobaud generator will automatically run at the appropriate baud rate.

Slow bus rise times due to the pullup resistor become a limiting factor when operating at high speeds. To compensate for slow rise times, the output driver can be configured to drive the line High. If the TXD (Transmit Drive) bit is set, the line will be driven both High and Low during transmission. The line starts being driven at the beginning of the start bit and stops being driven at the middle of the stop bit. If the TXDH (Transmit Drive High) bit is set, the line will be driven High until the input is High or the center of the bit occurs, whichever is first. If both TXD and TXDH are set, the pin will be driven High for one clock period at the beginning of each 0 to 1 transition. An example of a high-speed synchronous interface is displayed in Figure 60.

Table 163 contains a summary of the On-Chip Debugger commands; each is further described following the table. The table indicates those commands that operate when the device is not in DEBUG mode (normal operation) and those commands that are disabled by programming the Flash Read Protect option bit.

**Table 163. On-Chip Debugger Commands**

| Debug Command | Command Byte | Enabled when not in DEBUG mode? | Disabled by Read Protect Option Bit |
|---|---|---|---|
| Read Revision | 00H | Yes | – |
| Write OCD Counter Register | 01H | – | – |
| Read OCD Status Register | 02H | Yes | – |
| Read OCD Counter Register | 03H | – | – |
| Write OCD Control Register | 04H | Yes | – |
| Read OCD Control Register | 05H | Yes | – |
| Write Program Counter | 06H | – | Disabled. |
| Read Program Counter | 07H | – | Disabled. |
| Write Register | 08H | – | Writes to on-chip peripheral registers are enabled. Writes to the on-chip RAM are disabled. |
| Read Register | 09H | – | Reads from on-chip peripheral registers are enabled. Reads from the on-chip RAM are disabled. |
| Write Program Memory | 0AH | – | Disabled. |
| Read Program Memory | 0BH | – | Disabled. |
| Write Data Memory | 0CH | – | Disabled. |
| Read Data Memory | 0DH | – | Disabled. |
| Read Program Memory CRC | 0EH | – | – |
| Reserved | 0FH | – | – |
| Step Instruction | 10H | – | Disabled. |
| Stuff Instruction | 11H | – | Disabled. |
| Execute Instruction | 12H | – | Disabled. |
| Write Line Control Register | 18H | – | – |
| Read Line Control Register | 19H | – | – |
| Read Baud Reload Register | 1BH | – | – |

Note: Unlisted command byte values are reserved.

**Table 168. Oscillator Configuration and Selection**

| Clock Source | Characteristics | Required Setup |
|---|---|---|
| Internal Precision RC Oscillator | • Selectable frequency 0.0432MHz, 0.0864MHz, 0.3456MHz, 0.6912MHz, 1.3824MHz, 2.7648MHz, 5.5296MHz and 11.0592MHz<br>• ± 4% accuracy when trimmed<br>• No external components required | • Unlock and write Oscillator Control Register (OSCCTL0) to enable and select oscillator frequency. |
| External Crystal/ Resonator | • 32kHz to 20MHz<br>• Very high accuracy (dependent on crystal or resonator used)<br>• External components required | • Configure Flash option bits for correct external oscillator mode<br>• Unlock and write OSCCTL0 to enable crystal oscillator, wait for it to stabilize and select as system clock (if the EXTL_AO option bit has been deasserted, no waiting is required) |
| External RC Oscillator | • 32kHz to 4MHz<br>• Accuracy dependent on external components | • Configure Flash option bits for correct external oscillator mode<br>• Unlock and write OSCCTL0 to enable crystal oscillator and select as system clock |
| External Clock Drive | • 0 to 20MHz<br>• Accuracy dependent on external clock source | • Write GPIO registers to configure PB3 pin for external clock function<br>• Unlock and write OSCCTL0 to select external system clock<br>• Apply external clock signal to GPIO |
| Internal WDT Oscillator | • 10kHz nominal<br>• ± 40% accuracy; no external components required<br>• Low power consumption | • Enable WDT if not enabled and wait until WDT Oscillator is operating.<br>• Unlock and write Oscillator Control Register (OSCCTL0) to enable and select oscillator |

**! Caution:** Unintentional accesses to the Oscillator Control Register can actually stop the chip by switching to a nonfunctioning oscillator. To prevent this condition, the oscillator control block employs a register unlocking/locking scheme.

### 24.1.1.1. OSC Control Register Unlocking/Locking

To write to the Oscillator Control Register (OSCCTL0 and OSCCTL1), unlock it by making two writes to the OSCCTLx Register with the value E7H followed by the value 18H. A third write to the OSCCTLx Register changes the value of the actual register and returns it to a locked state. Any other sequence of oscillator control register writes has no effect. The values written to unlock the register must be ordered correctly, but are not necessarily con-

The Watchdog Timer oscillator failure-detection circuit counts system clocks while looking for a Watchdog Timer clock. The logic counts 8004 system clock cycles before determining that a failure has occurred. The system clock rate determines the speed at which the Watchdog Timer failure can be detected. A very slow system clock results in very slow detection times.

**!** **Caution:** It is possible to disable the clock failure detection circuitry as well as all functioning clock sources. In this case, the Z8 Encore! XP F1680 Series device ceases functioning and can only be recovered by Power-on reset.

## 24.2. Peripheral Clock

The peripheral clock is based on a low-frequency/low-power 32kHz secondary oscillator that can be used with an external watch crystal. The peripheral clock is only available for driving Timer and associated noise filter operation. It is not supported for other peripherals. The dedicated peripheral clock source allows Timer operation when the device is in STOP Mode.

Table 169 summarizes peripheral clock source features and usage.

**Table 169. Peripheral Clock Source and Usage**

| Peripheral Clock Source | Characteristics | Required Setup |
|---|---|---|
| Secondary Oscillator | • Optimized for use with a 32kHz Watch Crystal<br>• Very high accuracy<br>• Dedicated XTAL pins<br>• No external components | • Unlock and write OSCCTL1 to enable the secondary oscillator<br>• Select the peripheral clock at the timer clock source in the TxCTL2 Register |

## 24.3. Oscillator Control Register Definitions

The Oscillator Control registers enable and disable the various oscillator circuits, enable and disable the failure detection and recovery circuitry, and select the primary oscillator, which becomes the system clock.

### 24.3.1. Oscillator Control 0 Register

The Oscillator Control 0 Register (OSCCTL0) must be unlocked before writing. Unlock the Oscillator Control 0 Register by writing the two-step sequence of E7H followed by

**!** **Caution:** When using the external RC oscillator mode, the oscillator can stop oscillating if the power supply drops below 1.6V but remains above the Voltage Brown-Out threshold. The oscillator resumes oscillation when the supply voltage exceeds 1.6V.

## 25.4. Secondary Crystal Oscillator Operation

Figure 65 displays the recommended configuration for connection with an external fundamental-mode, parallel-resonant crystal operating at 32kHz. The recommended 32kHz crystal specifications are provided in Table 173. Printed circuit board layout must add no more than 4pF of stray capacitance to either the $X_{IN}$ or $X_{OUT}$ pins. If oscillation does not occur, reduce the values of capacitors $C_1$ and $C_2$ to decrease loading.

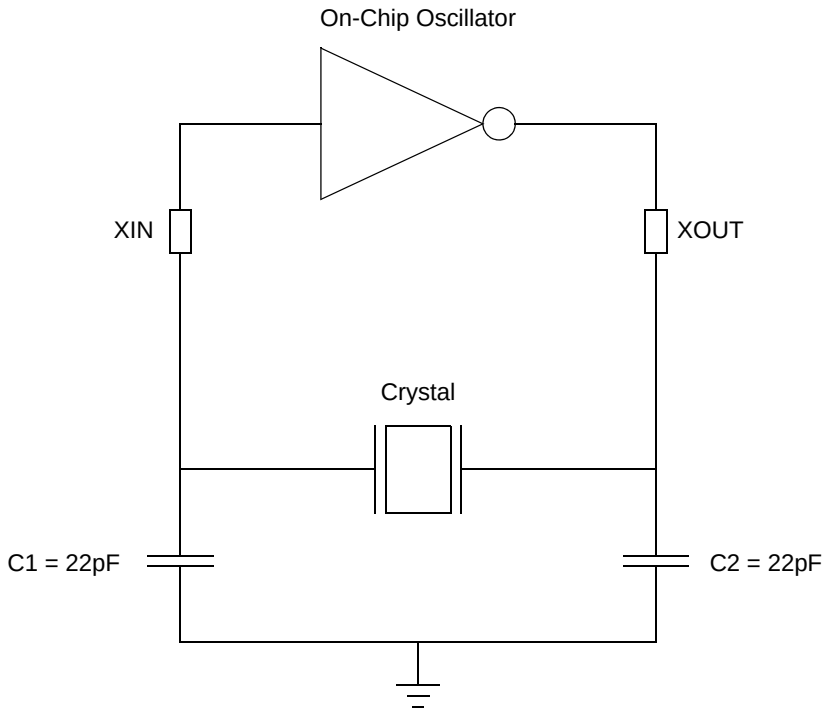


**Figure 65. Recommended 32kHz Crystal Oscillator Configuration**

## T



## U