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	-40°C ~ 105°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/z8f0880hj020eg

7.11.2. Port A–E Control Registers

The Port A–E Control registers set the GPIO port operation. The value in the corresponding Port A–E Address register determines which subregister is read from or written to by a Port A–E Control Register transaction, see Table 22.

Table 22. Port A–E Control Registers (PxCTL)

Bits	7	6	5	4	3	2	1	0
Field	PCTL							
Reset	00H							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Address	FD1H, FD5H, FD9H, FDDH, FE1H							

Bit	Description
[7:0] PCTL	**Port Control** The Port Control Register provides access to all subregisters that configure the GPIO Port operation.

7.11.3. Port A–E Data Direction Subregisters

The Port A–E Data Direction subregister is accessed through the Port A–E Control Register by writing 01H to the Port A–E Address register, as indicated in Table 23.

Table 23. Port A–E Data Direction Subregisters (PxDD)

Bits	7	6	5	4	3	2	1	0
Field	DD7	DD6	DD5	DD4	DD3	DD2	DD1	DD0
Reset	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Address	If 01H in Port A–E Address Register, accessible through the Port A–E Control Register.							

Bit	Description
[7:0] DD	**Data Direction** These bits control the direction of the associated port pin. Port Alternate Function operation overrides the Data Direction register setting. 0 = Output. Data in the Port A–E Output Data Register is driven onto the port pin. 1 = Input. The port pin is sampled and the value written into the Port A–E Input Data Register. The output driver is tristated.

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** 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). 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** These two bytes, {PWM1H[7:0], PWM1L[7:0]}, store the 16-bit captured timer value for DEMODULATION Mode.

10.7.7. Multi-Channel Timer Channel Status 0 and Status 1 Registers

The Multi-Channel Timer Channel Status 0 and Status 1 registers (MCTCHS0, MCTCHS1) indicate channel overruns and channel capture/compare events.

Table 76. Multi-Channel Timer Channel Status 0 Register (MCTCHS0)

Bit	7	6	5	4	3	2	1	0
Field	Reserved				CHDEO	CHCEO	CHBEO	CHAEO
Reset	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R
Address	See note.							
Note: If a 01H is in the Subaddress Register, it is accessible through Subregister 0.								

Bit	Description
[7:4]	Reserved; must be 0.
[3:0] CHyEO	**Channel y Event Flag Overrun** This bit indicates that an overrun error has occurred. An overrun occurs when a new Capture/Compare event occurs before the previous CHyEF bit is cleared. Clearing the associated CHyEF bit in the MCTCHS1 register clears this bit. This bit is cleared when TEN=0 (TEN is the MSB of MCTCTL1). 0 = No Overrun. 1 = Capture/Compare Event Flag Overrun

10.7.9. Multi-Channel Timer Channel-y High and Low Byte Registers

Each channel has a 16-bit capture/compare register defined here as the Channel-y High and Low Byte registers. When the timer is enabled, writes to these registers are buffered and loading of the registers is delayed until the next timer end count, unless CHUE = 1.

Table 79. Multi-Channel Timer Channel-y High Byte Registers (MCTCHyH)*

Bit	7	6	5	4	3	2	1	0
Field	CHyH							
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	See note.							
Note: If 02H, 03H, 04H and 05H are in the Subaddress Register, they are accessible through Subregister 0.								

Bit	7	6	5	4	3	2	1	0
Field	CHyL							
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								
Note: If 02H, 03H, 04H and 05H are in the Subaddress Register, they are accessible through Subregister 1.								

Bit	Description
[7:0] CHyH, CHyL	**Multi-Channel Timer Channel-y High and Low Bytes** During a compare operation, these two bytes, {CHyH[7:0], CHyL[7:0]}, form a 16-bit value that is compared to the current 16-bit timer count. When a match occurs, the Channel Output changes state. The Channel Output value is set by the TPOL bit in the Channel-y Control subregister. During a capture operation, the current Timer Count is recorded in these two bytes when the appropriate Channel Input transition occurs.
Note: *y = A, B, C, D.	

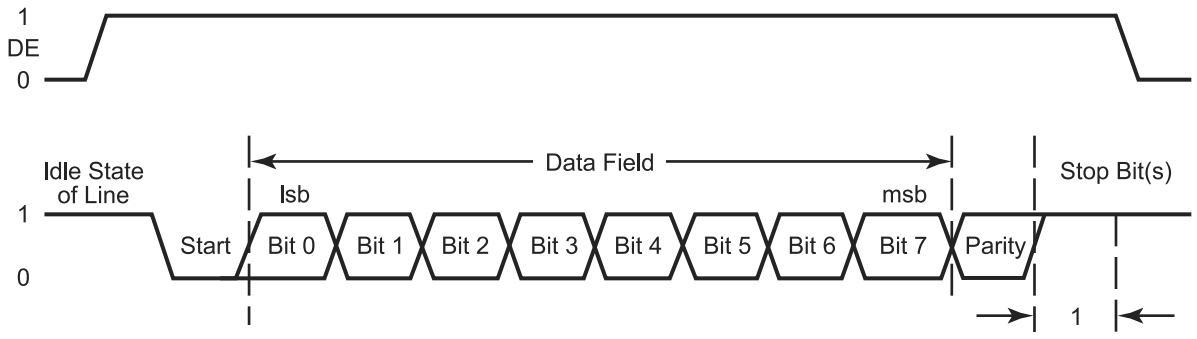

Figure 22. LIN-UART Driver Enable Signal Timing with One Stop Bit and Parity

The Driver Enable to start bit set-up time is calculated as follows:

$$\frac{1}{\text{Baud Rate (Hz)}} \leq \text{DE to Start Bit Set-up Time(s)} \leq \frac{2}{\text{Baud Rate (Hz)}}$$

12.1.8. LIN-UART Special Modes

The special modes of the LIN-UART are:

- MULTIPROCESSOR Mode
- LIN Mode

The LIN-UART features a common control register (Control 0) that has a unique register address and several mode-specific control registers (Multiprocessor Control, Noise Filter Control and LIN Control) that share a common register address (Control 1). When the Control 1 address is read or written, the MSEL[2:0] (Mode Select) field of the Mode Select and Status Register determines which physical register is accessed. Similarly, there are mode-specific status registers, one of which is returned when the Status 0 Register is read depending on the MSEL field.

12.1.9. MULTIPROCESSOR Mode

The LIN-UART features a MULTIPROCESSOR (9-bit) Mode that uses an extra (9^{th}) bit for selective communication when a number of processors share a common UART bus. In MULTIPROCESSOR Mode (also referred to as 9-bit mode), the multiprocessor bit (MP) is transmitted immediately following the 8 bits of data and immediately preceding the stop bit(s) as displayed in Figure 23.

Wake-up message if it requires the master to initiate a LIN message frame. Following the Wake-up message, the master wakes up and initiates a new message. A Wake-up message is accomplished by pulling the bus Low for at least 250 µs but less than 5 ms. Transmitting a 00H character is one way to transmit the Wake-up message.

If the CPU is in STOP Mode, the LIN-UART is not active and the Wake-up message must be detected by a GPIO edge detect Stop Mode Recovery. The duration of the Stop Mode Recovery sequence can preclude making an accurate measurement of the Wake-up message duration.

If the CPU is in HALT or OPERATIONAL mode, the LIN-UART (if enabled) times the duration of the Wake-up and provides an interrupt following the end of the break sequence if the duration is ≥ 3 bit times. The total duration of the Wake-up message in bit times can be obtained by reading the RxBreakLength field in the Mode Select and Status register. After a Wake-up message has been detected, the LIN-UART can be placed (by software) either into LIN Master or LIN Slave Wait for Break states as appropriate. If the break duration exceeds 15 bit times, the RxBreakLength field contains the value Fh. If the LIN-UART is disabled, Wake-up message is detected via a port pin interrupt and timed by software. If the device is in STOP Mode, the High to Low transition on the port pin will bring the device out of STOP Mode.

The LIN Sleep state is selected by software setting LinState[1:0] = 00. The decision to move from an active state to sleep state is based on the LIN messages as interpreted by software.

12.1.10.5. LIN Slave Operation

LIN SLAVE Mode is selected by setting LMST = 0, LSLV = 1, ABEN = 1 or 0 and LinState[1:0] = 01b (Wait for Break state). The LIN slave detects the start of a new message by the break which appears to the Slave as a break of at least 11 bit times in duration. The LIN-UART detects the break and generates an interrupt to the CPU. The duration of the break is observable in the RxBreakLength field of the Mode Select and Status register. A break of less than 11 bit times in duration does not generate a break interrupt when the LIN-UART is in Wait for Break state. If the break duration exceeds 15 bit times, the RxBreakLength field contains the value Fh.

Following the break, the LIN-UART hardware automatically transits to the *Autobaud* state, where it autobauds by timing the duration of the first 8 bit times of the Synch character as defined in the LIN standard. The duration of the autobaud period is measured by the BRG Counter which will update every 8th system clock cycle between the start bit and the beginning of bit 7 of the autobaud sequence. At the end of the autobaud period, the duration measured by the BRG counter (auto baud period divided by 8) is automatically transferred to the Baud Reload High and Low registers if the ABEN bit of the LIN control register is set. If the BRG Counter overflows before reaching the start of bit 7 in the autobaud sequence the Autobaud Overrun Error interrupt occurs, the OE bit in the Status 0 Register is set and the Baud Reload registers are not updated. To autobaud within 2% of the master's baud rate, the slave system clock must be a minimum of 100 times the baud

- Noise Filter Control (NFCTL[2:0]) input selects the width of the up/down saturating counter digital filter; the available width ranges from 4 to 11 bits
- The digital filter output features hysteresis
- Provides an active low *Saturated State* output (FiltSatB) which is used as an indication of the presence of noise

12.2.1. Architecture

Figure 25 displays how the noise filter is integrated with the LIN-UART on a LIN network.

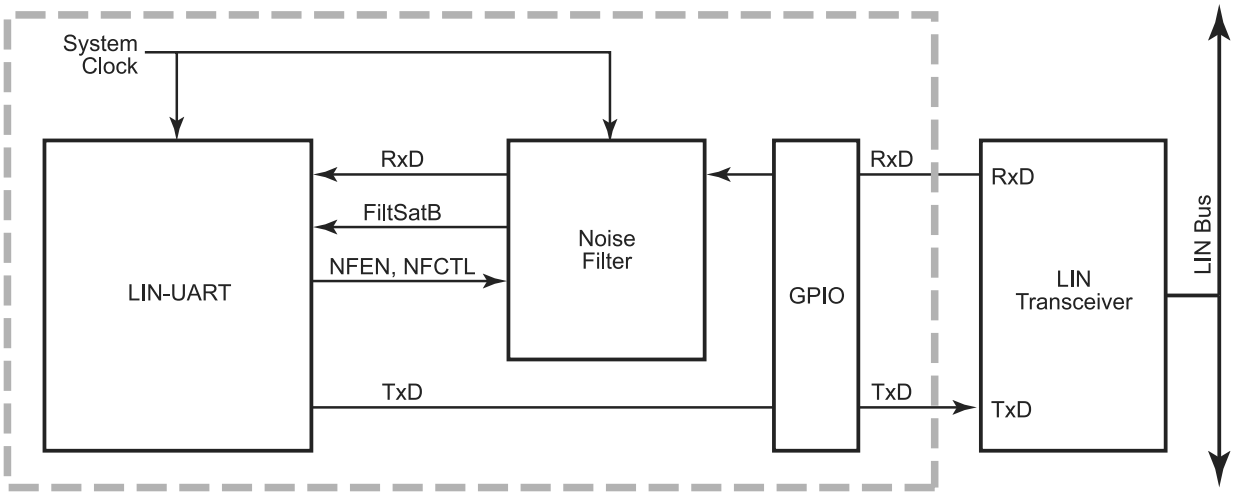

Figure 25. Noise Filter System Block Diagram

12.2.2. Operation

Figure 26 displays the operation of the noise filter both with and without noise. The noise filter in this example is a 2-bit up/down counter which saturates at `00b` and `11b`. A 2-bit counter is shown for convenience, the operation of wider counters is similar. The output of the filter switches from 1 to 0, when the counter counts down from `01b` to `00b`; and switches from 0 to 1, when the counter counts up from `10b` to `11b`. The noise filter delays the receive data by three System Clock cycles.

The FiltSatB signal is checked when the filtered RxD is sampled in the center of the bit time. The presence of noise (FiltSatB = 1 at center of bit time) does not mean that the sampled data is incorrect, but just that the filter is not in its ‘saturated’ state of all 1s or all 0s. If FiltSatB = 1 then RxD is sampled during a receive character, the NE bit in the

Table 115 defines the valid ESPI states.

Table 115. ESPISTATE Values

ESPISTATE Value	Description
00_0000	Idle
00_0001	Slave Wait For SCK
01_0001	Master Ready
10_1110	Bit 7 Receive
10_1111	Bit 7 Transmit
10_1100	Bit 6 Receive
10_1101	Bit 6 Transmit
10_1010	Bit 5 Receive
10_1011	Bit 5 Transmit
10_1000	Bit 4 Receive
10_1001	Bit 4 Transmit
10_0110	Bit 3 Receive
10_0111	Bit 3 Transmit
10_0100	Bit 2 Receive
10_0101	Bit 2 Transmit
10_0010	Bit 1 Receive
10_0011	Bit 1 Transmit
10_0000	Bit 0 Receive
10_0001	Bit 0 Transmit

16.4.7. ESPI Baud Rate High and Low Byte Registers

The ESPI Baud Rate High and Low Byte registers, shown in Tables 116 and 117, combine to form a 16-bit reload value, BRG[15:0], for the ESPI Baud Rate Generator. The ESPI baud rate is calculated using the following equation:

$$\text{SPI Baud Rate (bits § s)} = \frac{\text{System Clock Frequency (Hz)}}{2 \times \text{BRG[15:0]}}$$

The minimum baud rate is obtained by setting BRG[15:0] to 0000H for a clock divisor value of (2 x 65536 = 131072).

I2CISTAT Register is set to 1, thereby causing an interrupt. The RD bit is cleared to 0, indicating a Write to the slave. The I^2C controller acknowledges, indicating it is available to accept the data.

4. The software responds to the interrupt by reading the I2CISTAT Register, which clears the SAM bit. Because RD = 0, no immediate action is taken by the software until the first byte of data is received. If the software is only able to accept a single byte, it sets the NAK bit in the I2CCTL Register.
5. The Master detects the Acknowledge and sends the first byte of data.
6. The I^2C controller receives the first 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.
7. The software responds by reading the I2CISTAT Register, finding the RDRF bit = 1 and then reading the I2CDATA Register, which clears the RDRF bit. If the software can accept only one more data byte, it sets the NAK bit in the I2CCTL Register.
8. The Master and Slave loops through Step 5 to Step 7 until the Master detects a Not Acknowledge instruction or runs out of data to send.
9. The Master sends the stop or restart signal on the bus. Either of these signals can cause the I^2C controller to assert the 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.

17.2.6.7. Slave Transmit Transaction With 7-bit Address

The data transfer format for a master reading data from a slave in 7-bit address mode is displayed in Figure 49. The procedure that follows describes the I^2C Master/Slave Controller operating as a slave in 7-bit addressing mode and transmitting data to the bus master.

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

Figure 49. Data Transfer Format—Slave Transmit Transaction with 7-bit Address

1. The software configures the controller for operation as a slave in 7-bit addressing mode, as follows:
   a. Initialize the MODE field in the I^2C Mode Register for either SLAVE ONLY mode or MASTER/SLAVE Mode with 7-bit addressing.
   b. Optionally set the GCE bit.
   c. Initialize the SLA[6:0] bits in the I^2C Slave Address Register.

Table 127. I2CSTATE_L (Continued)

State I2CSTATE_H	Substate I2CSTATE_L	Substate Name	State Description
1000–1111	0111	Send/Receive bit 7	Sending/Receiving most significant bit.
	0110	Send/Receive bit 6	
	0101	Send/Receive bit 5	
	0100	Send/Receive bit 4	
	0011	Send/Receive bit 3	
	0010	Send/Receive bit 2	
	0001	Send/Receive bit 1	
	0000	Send/Receive bit 0	Sending/Receiving least significant bit.
	1000	Send/Receive Acknowledge	Sending/Receiving Acknowledge.

17.3.6. I²C Mode Register

The I²C Mode Register, shown in Table 128, provides control over master versus slave operating mode, slave address and diagnostic modes.

Table 128. I²C Mode Register (I2C Mode = F56H)

Bits	7	6	5	4	3	2	1	0
Field	Reserved	MODE[1:0]		IRM	GCE	SLA[9:8]		DIAG
Reset	0	0		0	0	0		0
R/W	R	R/W		R/W	R/W	R/W		R/W
Address	F56H							

Bit	Description
[7]	Reserved; must be 0.
[6:5] MODE[1:0]	**Selects the I²C Controller Operational Mode** 00 = MASTER/SLAVE capable (supports multi-master arbitration) with 7-bit Slave address. 01 = MASTER/SLAVE capable (supports multi-master arbitration) with 10-bit slave address. 10 = Slave Only capable with 7-bit address. 11 = Slave Only capable with 10-bit address.

Bit	Description (Continued)
[5:2] REFLVL	**Comparator 0 Internal Reference Voltage Level** This reference is independent of the ADC voltage reference. 0000 = 0.0 V 0001 = 0.2 V 0010 = 0.4 V 0011 = 0.6 V 0100 = 0.8 V 0101 = 1.0 V (Default) 0110 = 1.2 V 0111 = 1.4 V 1000 = 1.6 V 1001 = 1.8 V 1010–1111 = Reserved
[1:0] TIMTRG	**Timer Trigger (COMPARATOR COUNTER MODE)** 00 = Disable Timer Trigger. 01 = Comparator 0 output works as Timer 0 Trigger. 10 = Comparator 0 output works as Timer 1 Trigger. 11 = Comparator 0 output works as Timer 2 Trigger.

18.2.2. Comparator 1 Control Register

The Comparator 1 Control Register (CMP1), shown in Table 131, configures the comparator 1 inputs and sets the value of the internal voltage reference.

Table 131. Comparator 1 Control Register (CMP1)

Bits	7	6	5	4	3	2	1	0
Field	INPSEL	INNSEL	REFLVL				TIMTRG	
Reset	0	0	0	1	0	1	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Address	F91H							

Bit	Description
[7] INPSEL	**Signal Select for Positive Input** 0 = GPIO pin used as positive comparator 1 input. 1 = Temperature sensor used as positive comparator 1 input.
[6] INNSEL	**Signal Select for Negative Input** 0 = Internal reference disabled, GPIO pin used as negative comparator 1 input. 1 = Internal reference enabled as negative comparator 1 input.

Table 133. Flash Code Protection Using the Flash Option Bit

FWP	Flash Code Protection Description
0	Programming and erasing disabled for all of Flash Program Memory. In user code programming, Page Erase and Mass Erase are all disabled. Mass Erase is available through the On-Chip Debugger.
1	Programming, Page Erase and Mass Erase are enabled for all of Flash Program Memory.

20.2.3.2. Flash Code Protection Using the Flash Controller

At Reset, the Flash Controller locks to prevent accidental program or erasure of the contents of Flash memory. Follow the steps below to unlock the Flash Controller from user code:

1. Write the Page Select Register with the target page.
2. Write the first unlock command 73H to the Flash Control Register.
3. Write the second unlock command 8CH to the Flash Control Register.
4. Rewrite the Page Select Register with the same page previously stored there.

If the two Page Select writes do not match, the controller reverts to a locked state. If the two writes match, the selected page becomes active. For details, see the flowchart in Figure 54 on page 266.

Note: Byte Programming, Page Erase and Mass Erase will not be allowed if the FWP bit is cleared or if the page resides in a protected block.

After unlocking a specific page, Byte Programing or Page Erase can be performed. At the conclusion of a Page Erase, the Flash Controller is automatically locked. To lock the Flash Controller after Byte Programming, write to the Flash Control Register with any value other than the Page Erase or Mass Erase comands.

20.2.3.3. Sector Based Flash Protection

The final protection mechanism is implemented on a per-sector basis. The Flash memories of Z8 Encore! devices are divided into a maximum number of 8 sectors. A sector is 1/8 of the total size of Flash memory unless this value is smaller than the page size, in which case the sector and page sizes are equal. On the Z8 Encore! XP F1680 Series devices, the sector size is 3KB, 2KB or 1KB depending on available on-chip Flash size of 24KB, 16KB and 8KB.

The Flash Sector Protect Register can be configured to prevent sectors from being programmed or erased. After a sector is protected, it cannot be unprotected by user code. The Flash Sector Protect Register is cleared after reset and any previously written protection values is lost. User code must write this register in their initialization routine if they want to enable sector protection.

The Flash Sector Protect Register shares its Register File address with the Page Select Register. The Flash Sector Protect Register is accessed by writing the Flash Control Register with 5EH. After the Flash Sector Protect Register is selected, it can be accessed at the Page Select Register address. When user code writes the Flash Sector Protect Register, bits can only be set to 1. Thus, sectors can be protected, but not unprotected, via register write operations. Writing a value other than 5EH to the Flash Control Register deselects the Flash Sector Protect Register and reenables access to the Page Select Register. 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.

The Sector Protect Register is initialized to 0 on Reset, putting each sector into an unprotected state. When a bit in the Sector Protect Register is written to 1, the corresponding sector can no longer be written or erased. After a bit of the Sector Protect Register has been set, it can not be cleared except by a System Reset.

20.2.4. Byte Programming

Flash memory is enabled for byte programming on the active page after unlocking the Flash Controller. Erase the address(es) to be programmed using either the Page Erase or Mass Erase command prior to performing byte programming. An erased Flash byte contains all 1s (FFH). The programming operation can only be used to change bits from 1 to 0. To change a Flash bit (or multiple bits) from 0 to 1 requires execution of either the Page Erase or Mass Erase command.

Byte programming can be accomplished using the On-Chip Debugger's Write Memory command or eZ8 CPU execution of the LDC or LDCI instructions. For a description of the LDC and LDCI instructions, refer to the eZ8 CPU Core User Manual (UM0128), available for download at www.zilog.com. While the Flash Controller programs the contents of Flash memory, the eZ8 CPU idles but the system clock and on-chip peripherals continue to operate.

After a byte is written, the page remains unlocked, allowing for subsequent writes to other bytes on the same page. To exit programming mode and lock Flash memory, write any value to the Flash Control Register except the Mass Erase or Page Erase commands.

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** 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. 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** TEMPCALL is the low eight bits of 10-bit calibration data. The entire 10-bit calibration data field is {TEMPCALH[1:0], TEMPCALL}.

23.2. Operation of the On-Chip Debugger Interface

The On-Chip Debugger (OCD) uses the DBG pin for communication with an external host. This one-pin interface is a bidirectional open-drain interface that transmits and receives data. Data transmission is half-duplex, in that transmit and receive cannot occur simultaneously. The serial data on the DBG pin is sent using the standard asynchronous data format defined in RS-232. This pin interfaces the Z8 Encore! XP F1680 Series device to the serial port of a host PC using minimal external hardware. Figure 56 displays the connections between the debug connector and the Z8 Encore! microcontroller. Two different methods for connecting the DBG pin to an RS-232 interface are depicted in Figures 57 and and 58.

! Caution: For operation of the Z8 Encore! XP F1680 Series device, all power pins (V_{DD} and AV_{DD}) must be supplied with power and all ground pins (V_{SS} and AV_{SS}) must be properly grounded. The DBG pin should always be connected to V_{DD} through an external pull-up resistor.

The Serial Smart Cable (SSC) does not work with the F1680 device series because it does not fully support the silicon OCD. During external clock switching, the OCD sends a break command to the SSC. This causes the SSC to disconnect from the target and terminate the debug session. You must then reconnect to the target again. Use the Opto-Isolated USB, USB, or Ethernet Smart Cables when using in conjunction with ZDSII.

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.

23.4.2. OCD Status Register

The OCD Status Register, shown in Table 165, reports status information about the current state of the debugger and the system.

Table 165. OCD Status Register (OCDSTAT)

Bit	7	6	5	4	3	2	1	0
Field	IDLE	HALT	RPEN	Reserved				
Reset	0	0	0	0				
R/W	R	R	R	R				

Bit	Description
[7] IDLE	**CPU Idle** This bit is set if the part is in Debug mode (DBGMODE is 1) or if a BRK instruction has occurred since the last time OCDCTL was written. This can be used to determine if the CPU is running or if it is idle. 0 = The eZ8 CPU is running. 1 = The eZ8 CPU is either stopped or looping on a BRK instruction.
[6] HALT	**HALT Mode** 0 = The device is not in HALT Mode. 1 = The device is in HALT Mode.
[5] RPEN	**Read Protect Option Bit Enable** 0 = The Read Protect option bit is disabled (Flash option bit is 1). 1 = The Read Protect option bit is enabled (Flash option bit is 0), disabling many OCD commands.
[4:0]	Reserved; must be 0.

23.4.4. Baud Reload Register

The Baud Reload Register, shown in Table 167, contains the measured Autobaud value.

Table 167. Baud Reload Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved				RELOAD											
Reset	0H				000H											
R/W	R				R											

Bit	Description
[15:12]	Reserved; must be 000.
[11:0] RELOAD	**Baud Reload Value** This value is the measured Autobaud value. Its value can be calculated using the formula: $RELOAD = \frac{SYSCLK}{BAUDRATE} \times 8$

Table 181. CPU Control Instructions (Continued)

Mnemonic	Operands	Instruction
STOP	—	STOP Mode
WDT	—	Watchdog Timer Refresh

Table 182. Load Instructions

Mnemonic	Operands	Instruction
CLR	dst	Clear
LD	dst, src	Load
LDC	dst, src	Load Constant to/from Program Memory
LDCI	dst, src	Load Constant to/from Program Memory and Auto-Increment Addresses
LDE	dst, src	Load External Data to/from Data Memory
LDEI	dst, src	Load External Data to/from Data Memory and Auto-Increment Addresses
LDWX	dst, src	Load Word using Extended Addressing
LDX	dst, src	Load using Extended Addressing
LEA	dst, X(src)	Load Effective Address
POP	dst	Pop
POPX	dst	Pop using Extended Addressing
PUSH	src	Push
PUSHX	src	Push using Extended Addressing

Table 183. Logical Instructions

Mnemonic	Operands	Instruction
AND	dst, src	Logical AND
ANDX	dst, src	Logical AND using Extended Addressing
COM	dst	Complement
OR	dst, src	Logical OR
ORX	dst, src	Logical OR using Extended Addressing
XOR	dst, src	Logical Exclusive OR
XORX	dst, src	Logical Exclusive OR using Extended Addressing

Table 186. eZ8 CPU Instruction Summary (Continued)

Assembly Mnemonic	Symbolic Operation	Address Mode dst	Address Mode src	Op Code(s) (Hex)	Flags C	Z	S	V	D	H	Fetch Cycles	Instr. Cycles
CLR dst	dst ← 00H	R		B0	–	–	–	–	–	–	2	2
		IR		B1							2	3
COM dst	dst ← ~dst	R		60	–	*	*	0	–	–	2	2
		IR		61							2	3
CP dst, src	dst – src	r	r	A2	*	*	*	*	–	–	2	3
		r	Ir	A3							2	4
		R	R	A4							3	3
		R	IR	A5							3	4
		R	IM	A6							3	3
		IR	IM	A7							3	4
CPC dst, src	dst – src – C	r	r	1F A2	*	*	*	*	–	–	3	3
		r	Ir	1F A3							3	4
		R	R	1F A4							4	3
		R	IR	1F A5							4	4
		R	IM	1F A6							4	3
		IR	IM	1F A7							4	4
CPCX dst, src	dst – src – C	ER	ER	1F A8	*	*	*	*	–	–	5	3
		ER	IM	1F A9							5	3
CPX dst, src	dst – src	ER	ER	A8	*	*	*	*	–	–	4	3
		ER	IM	A9							4	3
DA dst	dst ← DA(dst)	R		40	*	*	*	X	–	–	2	2
		IR		41							2	3
DEC dst	dst ← dst – 1	R		30	–	*	*	*	–	–	2	2
		IR		31							2	3
DECW dst	dst ← dst – 1	RR		80	–	*	*	*	–	–	2	5
		IRR		81							2	6
DI	IRQCTL[7] ← 0			8F	–	–	–	–	–	–	1	2

Flags notation:
* = Value is a function of the result of the operation.
– = Unaffected.
X = Undefined.
0 = Reset to 0.
1 = Set to 1.