Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	eZ8
Core Size	8-Bit
Speed	20MHz
Connectivity	I²C, IrDA, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LED, LVD, POR, PWM, Temp Sensor, WDT
Number of I/O	37
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	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	44-VFQFN Exposed Pad
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f2480qn020sg

2.4. Pin Characteristics

Table 5 provides detailed information about the characteristics of each pin available on the F1680 Series MCU 20-, 28-, 40- and 44-pin devices. Data provided in Table 5 is sorted alphabetically by the pin symbol mnemonic.

Table 5. Pin Characteristics (20-, 28-, 40- and 44-pin Devices)

Symbol Mnemonic	Direction	Reset Direction	Active Low or Active High	Tristate Output	Internal Pull-up or Pull-down	Schmitt Trigger Input	Open Drain Output	5V Tolerance
AV_{DD}	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
AV_{SS}	N/A	N/A	N/A	N/A	N/A	N/A	N/A	NA
DBG	I/O	I	N/A	Yes	Yes	Yes	Yes	No
PA[7:0]	I/O	I	N/A	Yes	Programmable pull-up	Yes	Yes, programmable	Yes, 5V tolerant inputs unless pull-ups are enabled
PB[5:0]	I/O	I	N/A	Yes	Programmable pull-up	Yes	Yes, programmable	Yes, 5V tolerant inputs unless pull-ups are enabled
PC[7:0]	I/O	I	N/A	Yes	Programmable pull-up	Yes	Yes, programmable	Yes, 5V tolerant inputs unless pull-ups are enabled
PD[7:1]	I/O	I	N/A	Yes	Programmable pull-up	Yes	Yes, programmable	Yes, 5V tolerant inputs unless pull-ups are enabled

Table 6. F1680 Series MCU Program Memory Maps (Continued)

Program Memory Address (Hex)	Function
Z8F1680 Device	
0000–0001	Flash option bits
0002–0003	Reset vector
0004–0005	WDT interrupt vector
0006–0007	Illegal instruction trap
0008–0037	Interrupt vectors*
0038–003D	Oscillator fail traps*
003E–3FFF	Program Flash
E000–E3FF	1KB PRAM
Z8F0880 Device	
0000–0001	Flash option bits
0002–0003	Reset vector
0004–0005	WDT interrupt vector
0006–0007	Illegal instruction trap
0008–0037	Interrupt vectors*
0038–003D	Oscillator fail traps*
003E–1FFF	Program Flash
E000–E3FF	1KB PRAM

Note: *See Table 36 on page 69 for a list of interrupt vectors and traps.

3.3. Data Memory

The F1680 Series MCU does not use the eZ8 CPU's 64KB Data Memory address space.

3.4. Flash Information Area

Table 7 describes the F1680 Series MCU Flash Information Area. This 512-byte Information Area is accessed by setting bit 7 of the Flash Page Select Register to 1. When access is enabled, the Flash Information Area is mapped into the Program Memory and overlays the 512bytes at addresses FE00H to FFFFH. When the Information Area access is enabled, all reads from these Program Memory addresses return the Information Area data rather than the Program Memory data. Access to the Flash Information Area is read-only.

Table 7. F1680 Series MCU Flash Memory Information Area Map

Program Memory Address (Hex)	Function
FE00–FE3F	Zilog option bits
FE40–FE53	Part Number: 20-character ASCII alphanumeric code Left-justified and filled with FH
FE54–FE5F	Reserved
FE60–FE7F	Zilog calibration data (only use the first two bytes FE60 and FE61)
FE80–FFFF	Reserved

7.2. Architecture

Figure 9 displays a simplified block diagram of a GPIO port pin and does not illustrate the ability to accommodate alternate functions and variable port current drive strength.

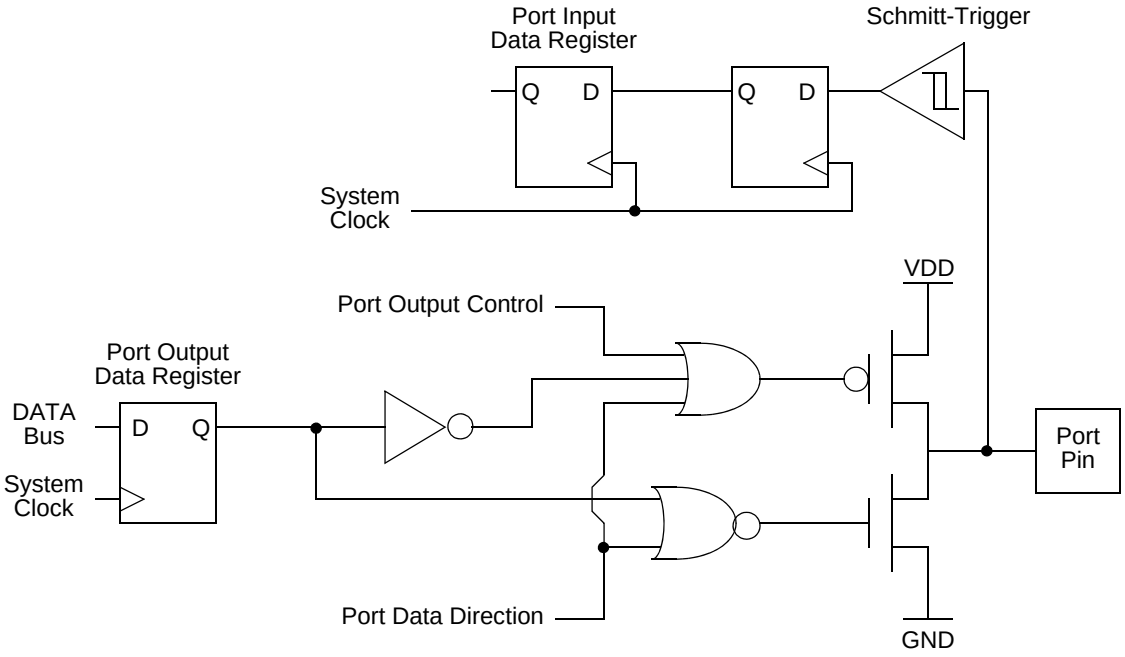


Figure 9. GPIO Port Pin Block Diagram

7.3. GPIO Alternate Functions

Many GPIO port pins are used for GPIO and to access the on-chip peripheral functions like the timers and serial-communication devices. The Port A–E Alternate Function sub-registers configure these pins for either GPIO or alternate function operation. When a pin is configured for alternate function, control of port-pin direction (input/output) is passed from Port A–E Data Direction registers to the alternate functions assigned to this pin. Tables 17 through 19 list the alternate functions possible with each port pin for every package. The alternate function associated at a pin is defined through alternate function sets subregisters AFS1 and AFS2.

The crystal oscillator and the 32 kHz secondary oscillator functionalities are not controlled by the GPIO block. When the crystal oscillator or the 32 kHz secondary oscillator is enabled in the oscillator control block, the GPIO functionality of PA0 and PA1, or PA2 and PA3, is overridden. In such a case, those pins function as input and output for the crystal oscillator.

Table 30. Port A–E Alternate Function Set 2 Subregisters (PxAFS2)

Bits	7	6	5	4	3	2	1	0
Field	PAFS27	PAFS26	PAFS25	PAFS24	PAFS23	PAFS22	PAFS21	PAFS20
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	If 08H in Port A–E Address Register, accessible through the Port A–E Control Register							

Bit	Description
[7:0] PAFS2	**Port Alternate Function Set 2** 0 = Port Alternate Function selected as defined in the GPIO Alternate Functions section on page 47. 1 = Port Alternate Function selected as defined the GPIO Alternate Functions section on page 47.

7.11.11. Port A–E Input Data Registers

Reading from the Port A–E Input Data registers, shown in Table 31, returns the sampled values from the corresponding port pins. The Port A–E Input Data registers are read-only. The value returned for any unused ports is 0. Unused ports include those missing on the 8-pin and 28-pin packages, as well as those missing on the ADC-enabled 28-pin packages.

Table 31. Port A–E Input Data Registers (PxIN)

Bits	7	6	5	4	3	2	1	0
Field	PIN7	PIN6	PIN5	PIN4	PIN3	PIN2	PIN1	PIN0
Reset	X	X	X	X	X	X	X	X
R/W	R	R	R	R	R	R	R	R
Address	FD2H, FD6H, FDAH, FDEH, FE2H							

Bit	Description
[7:0] PIN	**Port Input Data** Sampled data from the corresponding port pin input. 0 = Input data is logical 0 (Low). 1 = Input data is logical 1 (High).

If system clock is chosen as the clock source, the timer ceases to operate as a system clock and is put into STOP Mode. In this case the registers are not reset and operation will resume after Stop Mode Recovery occurs.

9.2.2.3. Power Reduction During Operation

Removal of the TEN bit will inhibit clocking of the entire timer block. The CPU can still read/write registers when the enable bit(s) are taken out.

9.2.3. Timer Operating Modes

The timers can be configured to operate in the following modes, each of which is described in this section where indicated in Table 52.

Table 52. Timer Operating Modes

Mode	Page Number
TRIGGERED ONE-SHOT Mode	88
CONTINUOUS Mode	90
COUNTER Mode	91
COMPARATOR COUNTER Mode	92
PWM SINGLE OUTPUT Mode	93
PWM DUAL Output Mode	95
CAPTURE Mode	97
CAPTURE RESTART Mode	98
COMPARE Mode	100
GATED Mode	100
CAPTURE/COMPARE Mode	102
DEMODULATION Mode	103

9.2.3.1. ONE-SHOT Mode

In ONE-SHOT Mode, the timer counts up to the 16-bit reload value stored in the Timer Reload High and Low Byte registers. The Timer counts timer clocks up to the 16-bit reload value. Upon reaching the reload value, the timer generates an interrupt, and the count value in the Timer High and Low Byte registers is reset to 0001H. Then, the timer is automatically disabled and stops counting.

Additionally, if the Timer Output alternate function is enabled, the Timer Output pin changes state for one clock cycle (from Low to High or from High to Low) upon timer reload. If it is appropriate to have the Timer Output make a permanent state change on

10.2.3. Multi-Channel Timer Clock Prescaler

The prescaler allows the system clock signal to be decreased by factors of 1, 2, 4, 8, 16, 32, 64 or 128. The PRES[2:0] bit field in the MCTCTL1 Register controls prescaler operation. The PRES field is buffered for the prescale value to change only on a Multi-Channel Timer end-of-cycle count. The prescaler has no effect when the T_{IN} is selected as the clock source.

10.2.4. Multi-Channel Timer Start

The Multi-Channel Timer starts counting when the TEN bit in the MCTCTL1 Register is set and the clock source is active. In Count Modulo or Count Up/Down mode, the timer counting can be stopped without disabling the timer by setting the Reload Register to 0. The timer will then stop when the counter next reaches 0. Writing a nonzero value to the Reload Register restarts the timer counting.

10.2.5. Multi-Channel Timer Mode Control

The Multi-Channel Timer supports two modes of operation: Count Modulo and Count up/down. The operating mode is selected with the TMODE[1:0] field in the MCTCTL1 Register. The timer modes are described below in Table 68.

Table 68. Timer Count Modes

TMODE	Timer Mode	Description
00	Count Modulo	Timer counts up to Reload Register value. Then it is reset to 0000H and counting resumes.
01	Reserved	
10	Count Up/Down	Timer counts up to Reload and then counts down to 0000H. The Count up/down cycle continues.
11	Reserved	

10.2.6. Count Modulo Mode

In the Count Modulo Mode, the Timer counts up to the Reload Register value (max value = FFFFH). Then it is reset to 0000H and counting resumes. As shown in Figure 15, the counting cycle continues with Reload + 1 as the period. A timer count interrupt request is generated when the timer count resets from Reload to 0000H. If Count Modulo is selected when the timer count is greater than Reload, the timer immediately restarts counting from zero.

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).

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.

- 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.

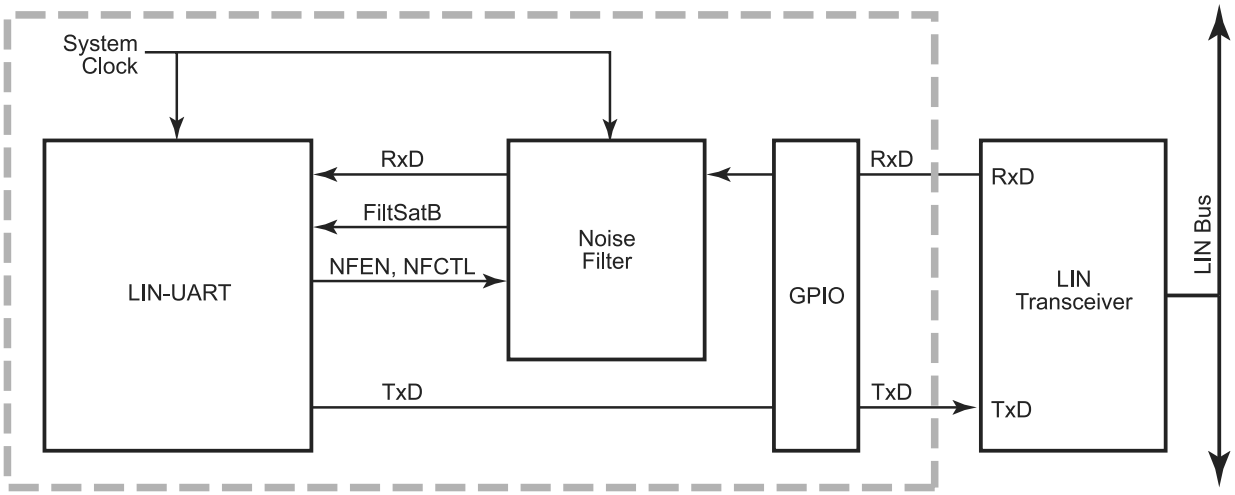


Figure 25. Noise Filter System Block Diagram

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

14.3.2. ADC Raw Data High Byte Register

The ADC Raw Data High Byte Register, shown in Table 102, contains the upper 8 bits of raw data from the ADC output. Access to the ADC Raw Data High Byte register is read-only. This register is used for test only.

Table 102. ADC Raw Data High Byte Register (ADCRD_H)

Bits	7	6	5	4	3	2	1	0
Field	ADCRDH							
Reset	X							
R/W	R							
Address	F71H							

Bit Position	Value (H)	Description
[7:0]	00–FF	**ADC Raw Data High Byte** The data in this register is the raw data coming from the SAR Block. It will change as the conversion is in progress. This register is used for testing only.

14.3.3. ADC Data High Byte Register

The ADC Data High Byte Register, shown in Table 103, contains the upper eight bits of the ADC output. Access to the ADC Data High Byte Register is read-only. Reading the ADC Data High Byte Register latches data in the ADC Low Bits Register.

Table 103. ADC Data High Byte Register (ADCD_H)

Bits	7	6	5	4	3	2	1	0
Field	ADCDH							
Reset	X							
R/W	R							
Address	F72H							

Bit Position	Value (H)	Description
[7:0]	00–FF	**ADC High Byte** The last conversion output is held in the data registers until the next ADC conversion has completed.

Bit	Description
[1] TEOF	**Transmit End of Frame** This bit is used in MASTER Mode to indicate that the data in the Transmit Data Register is the last byte of the transfer or frame. When the last byte has been sent $\overline{SS}$ (and SSV) will change state and TEOF will automatically clear. 0 = The data in the Transmit Data Register is not the last character in the message. 1 = The data in the Transmit Data Register is the last character in the message.
[0] SSV	**Slave Select Value** When SSIO = 1, writes to this register will control the value output on the $\overline{SS}$ pin. For more details, see the SSMD field of the ESPI Mode Register on page 217.

16.4.3. ESPI Control Register

The ESPI Control Register, shown in Table 111, configures the ESPI for transmit and receive operations.

Table 111. ESPI Control Register

Bits	7	6	5	4	3	2	1	0
Field	DIRQE	ESPIEN1	BRGCTL	PHASE	CLKPOL	WOR	MMEN	ESPIEN0
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	F62H							

Bit	Description
[7] DIRQE	**Data Interrupt Request Enable** This bit is used to disable or enable data (TDRE and RDRNE) interrupts. Disabling the data interrupts is needed to control data transfer by polling. Error interrupts are not disabled. To block all ESPI interrupt sources, clear the ESPI interrupt enable bit in the Interrupt Controller. 0 = TDRE and RDRNE assertions do not cause an interrupt. Use this setting if controlling data transfer by software polling of TDRE and RDRNE. The TUND, COL, ABT and ROVR bits will cause an interrupt. 1 = TDRE and RDRNE assertions will cause an interrupt. TUND, COL, ABT and ROVR will also cause interrupts. Use this setting when controlling data transfer via interrupt handlers.

Table 125. I²C State Register (I2CSTATE)—Description when DIAG = 0

Bits	7	6	5	4	3	2	1	0
Field	ACKV	ACK	AS	DS	10B	RSTR	SCLOUT	BUSY
Reset	0	0	0	0	0	0	1	0
R/W	R	R	R	R	R	R	R	R
Address	F55H							

Bit	Description
[7] ACKV	**ACK Valid** This bit is set, if sending data (Master or Slave) and the ACK bit in this register is valid for the byte just transmitted. This bit can be monitored if it is appropriate for software to verify the ACK value before writing the next byte to be sent. To operate in this mode, the Data Register must not be written when TDRE asserts; instead, the software waits for ACKV to assert. This bit clears when transmission of the next byte begins or the transaction is ended by a stop or restart condition.
[6] ACK	**Acknowledge** This bit indicates the status of the Acknowledge for the last byte transmitted or received. This bit is set for an Acknowledge and cleared for a Not Acknowledge condition.
[5] AS	**Address State** This bit is active High while the address is being transferred on the I²C bus.
[4] DS	**Data State** This bit is active High while the data is being transferred on the I²C bus.
[3] 10B	**10B** This bit indicates whether a 7-bit or 10-bit address is being transmitted when operating as a Master. After the start bit is set, if the five most-significant bits of the address are 11110B, this bit is set. When set, it is Reset after the address has been sent.
[2] RSTR	**RESTART** This bit is updated each time a stop or restart interrupt occurs (SPRS bit set in I2CISTAT Register). 0 = Stop condition. 1 = Restart condition.
[1] SCLOUT	**Serial Clock Output** Current value of Serial Clock being output onto the bus. The actual values of the SCL and SDA signals on the I²C bus can be observed via the GPIO Input Register.
[0] BUSY	**I²C Bus Busy** 0 = No activity on the I²C Bus. 1 = A transaction is underway on the I²C bus.

Table 126. I2CSTATE_H

State Encoding	State Name	State Description
0000	Idle	I^2C bus is idle or I^2C controller is disabled.
0001	Slave Start	I^2C controller has received a start condition.
0010	Slave Bystander	Address did not match; ignore remainder of transaction.
0011	Slave Wait	Waiting for stop or restart condition after sending a Not Acknowledge instruction.
0100	Master Stop2	Master completing stop condition (SCL = 1, SDA = 1).
0101	Master Start/Restart	MASTER Mode sending start condition (SCL = 1, SDA = 0).
0110	Master Stop1	Master initiating stop condition (SCL = 1, SDA = 0).
0111	Master Wait	Master received a Not Acknowledge instruction, waiting for software to assert stop or start control bits.
1000	Slave Transmit Data	Nine substates, one for each data bit and one for the Acknowledge.
1001	Slave Receive Data	Nine substates, one for each data bit and one for the Acknowledge.
1010	Slave Receive Addr1	Slave receiving first address byte (7- and 10-bit addressing) Nine substates, one for each address bit and one for the Acknowledge.
1011	Slave Receive Addr2	Slave Receiving second address byte (10-bit addressing) nine substates, one for each address bit and one for the Acknowledge.
1100	Master Transmit Data	Nine substates, one for each data bit and one for the Acknowledge.
1101	Master Receive Data	Nine substates, one for each data bit and one for the Acknowledge.
1110	Master Transmit Addr1	Master sending first address byte (7- and 10-bit addressing) nine substates, one for each address bit and one for the Acknowledge.
1111	Master Transmit Addr2	Master sending second address byte (10-bit addressing) nine substates, one for each address bit and one for the Acknowledge.

Table 127. I2CSTATE_L

State I2CSTATE_H	Substate I2CSTATE_L	Substate Name	State Description
0000–0100	0000	—	There are no substates for these I2CSTATE_H values.
0110–0111	0000	—	There are no substates for these I2CSTATE_H values.
0101	0000	Master Start	Initiating a new transaction
	0001	Master Restart	Master is ending one transaction and starting a new one without letting the bus go idle.

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.

Read Baud Reload Register (1BH). The Read Baud Reload Register command returns the current value in the Baud Reload Register.

```
DBG ← 1BH
DBG → BAUD[15:8]
DBG → BAUD[7:0]
```

Write Test Mode Register (F0H). The Write Test Mode Register command writes the data that follows to the Test Mode register.

```
DBG ← F0H
DBG ← TESTMODE[7:0]
```

Read Test Mode Register (F1H). The Read Test Mode Register command returns the current value in the Test Mode register.

```
DBG ← F1H
DBG → TESTMODE[7:0]
```

Write Option Bit Registers (F2H). The Write Option Bit Registers command is used to write to the option bit configuration registers. The option bit configuration registers store the device configuration and are loaded from Flash every time the Z8 Encore! XP F1680 Series is reset. The registers may be temporarily written using this OCD command to test peripherals without having to program the Flash Information Area and resetting the Z8 Encore! XP F1680 Series. The ZilogUserSel bit selects between Zilog option bits (1) and user option bits (0).

```
DBG ← F2H
DBG ← {ZilogUserSel,1'b0,OptAddr[4:0]}
DBG ← OptData[7:0]
```

Read Option Bit Registers (F3H). The Read Option Bit Registers command is used to read the option bit registers that store the device configuration that is read out of flash when the Z8 Encore! XP F1680 Series is reset. The ZilogUserSel bit selects between reading Zilog option bits (1) or user option bits (0).

```
DBG ← F1H
DBG ← {ZilogUserSel,1'b0,OptAddr[4:0]}
DBG → OptData[7:0]
```

23.4. On-Chip Debugger Control Register Definitions

This section defines the features of the following On-Chip Debugger Control registers.

OCD Control Register: see page 310

OCD Status Register: see page 312

Line Control Register: see page 313

Baud Reload Register: see page 314

Chapter 25. Crystal Oscillator

The products in the Z8 Encore! XP F1680 Series contain a primary on-chip crystal oscillator for use with external crystals with 1MHz to 20MHz frequencies, plus a secondary 32K crystal oscillator. In addition, the external oscillator supports external RC networks with oscillation frequencies up to 4MHz. The 32K secondary crystal oscillator does not feature an external RC oscillator mode. The on-chip crystal oscillator can be used to generate the primary system clock for the internal eZ8 CPU and the majority of the on-chip peripherals. Additionally, the secondary 32K crystal oscillator can only be used to generate a clock for three timers.

Alternatively, the X_{IN} and $X2_{IN}$ input pins can also accept a CMOS-level clock input signal (for X_{IN}, the frequency range is 32kHz–20MHz; for $X2_{IN}$, the range is below 4MHz). If an external clock generator is used, the X_{OUT} or $X2_{OUT}$ pin must remain unconnected. The Z8 Encore! XP F1680 Series products do not contain an internal clock divider. The frequency of the signal on the X_{IN} input pin determines the frequency of the system clock; the frequency of the signal on the $X2_{IN}$ determines the frequency of the timers.

Note: Although the X_{IN} pin can be used as a primary system clock input for an external clock generator, the CLKIN pin is better suited for such use. For details, see the System Clock Selection section on page 315.

25.1. Operating Modes

The primary on-chip crystal oscillator supports three oscillator modes:

- Medium power for use with medium frequency crystals or ceramic resonators (1MHz to 8MHz)
- Maximum power for use with high-frequency crystals (8MHz to 20MHz)
- On-chip oscillator configured for use with external RC networks or external clock input (<4MHz)

The primary on-chip crystal oscillator mode is selected using user-programmable Flash option bits. For information, see the Flash Option Bits section on page 276. The secondary 32kHz crystal oscillator supports two oscillator modes:

- NORMAL Mode for use with 32kHz crystals
- On-chip oscillator configured for use with external clock input

```
LD 234H, #%01 ; Another Load (LD) instruction with two operands.
              ; The first operand, Extended Mode Register Address 234H,
              ; identifies the destination. The second operand, Immediate Data
              ; value 01H, is the source. The value 01H is written into the
              ; Register at address 234H.
```

27.2. Assembly Language Syntax

For proper instruction execution, eZ8 CPU assembly language syntax requires that the operands be written as destination and source. After assembly, the object code usually has the operands in the order source, destination, but ordering is opcode-dependent. The following instruction examples illustrate the format of some basic assembly instructions and the resulting object code produced by the assembler. You must follow this binary format if you prefer manual program coding or intend to implement your own assembler.

Example 1. If the contents of registers 43H and 08H are added and the result is stored in 43H, the assembly syntax and resulting object code is as listed in Table 174.

Table 174. Assembly Language Syntax Example 1

Assembly Language Code	ADD	43H,	08H	(ADD dst, src)
Object Code	04	08	43	(OPC src, dst)

Example 2. In general, when an instruction format requires an 8-bit register address, that address can specify any register location in the range 0–255 or, using Escaped Mode Addressing, a Working Register R0–R15. If the contents of Register 43H and Working Register R8 are added and the result is stored in 43H, the assembly syntax and resulting object code is as listed in Table 175.

Table 175. Assembly Language Syntax Example 2

Assembly Language Code	ADD	43H,	R8	(ADD dst, src)
Object Code	04	E8	43	(OPC src, dst)

The register file size varies depending on the device type. See the device-specific product specification to determine the exact register file range available.

Table 181. CPU Control Instructions (Continued)

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

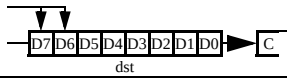
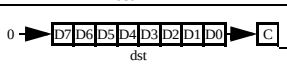
Table 182. Load Instructions

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
SCF	C ← 1			DF	1	–	–	–	–	–	1	2
SRA dst		R		D0	*	*	*	0	–	–	2	2
		IR		D1							2	3
SRL dst		R		1F C0	*	*	0	*	–	–	3	2
		IR		1F C1							3	3
SRP src	RP ← src		IM	01	–	–	–	–	–	–	2	2
STOP	STOP Mode			6F	–	–	–	–	–	–	1	2
SUB dst, src	dst ← dst – src	r	r	22	*	*	*	*	1	*	2	3
		r	Ir	23							2	4
		R	R	24							3	3
		R	IR	25							3	4
		R	IM	26							3	3
		IR	IM	27							3	4
SUBX dst, src	dst ← dst – src	ER	ER	28	*	*	*	*	1	*	4	3
		ER	IM	29							4	3
SWAP dst	dst[7:4] ↔ dst[3:0]	R		F0	X	*	*	X	–	–	2	2
		IR		F1							2	3
TCM dst, src	(NOT dst) AND src	r	r	62	–	*	*	0	–	–	2	3
		r	Ir	63							2	4
		R	R	64							3	3
		R	IR	65							3	4
		R	IM	66							3	3
		IR	IM	67							3	4
TCMX dst, src	(NOT dst) AND src	ER	ER	68	–	*	*	0	–	–	4	3
		ER	IM	69							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.