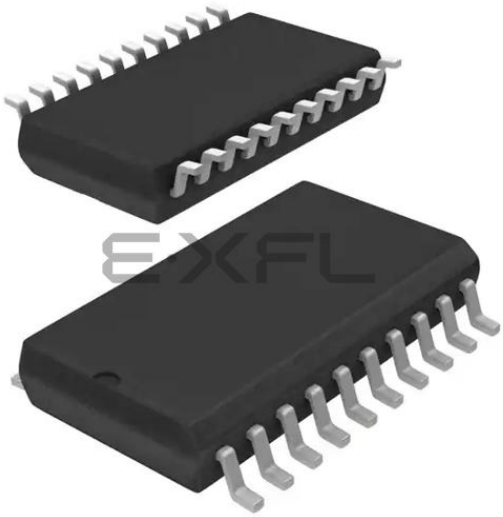


Welcome to E-XFL.COM

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, UART/USART
Peripherals	Brown-out Detect/Reset, LED, LVD, POR, PWM, Temp Sensor, WDT
Number of I/O	17
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 7x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	20-SOIC (0.295", 7.50mm Width)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f2480sh020eg

Table 8. Register File Address Map (Continued)

Address (Hex)	Register Description	Mnemonic	Reset (Hex)[1]	Page #
Multi-Channel Timer				
FA0	MCT High Byte	MCTH	00	130
FA1	MCT Low Byte	MCTL	00	130
FA2	MCT Reload High Byte	MCTRH	FF	131
FA3	MCT Reload Low Byte	MCTRL	FF	131
FA4	MCT Subaddress	MCTSA	XX	132
FA5	MCT Subregister 0	MCTSR0	XX	132
FA6	MCT Subregister 1	MCTSR1	XX	132
FA7	MCT Subregister 2	MCTSR2	XX	132
FA8–FBF	Reserved	—	XX	
Interrupt Controller				
FC0	Interrupt Request 0	IRQ0	00	73
FC1	IRQ0 Enable High Bit	IRQ0ENH	00	76
FC2	IRQ0 Enable Low Bit	IRQ0ENL	00	77
FC3	Interrupt Request 1	IRQ1	00	74
FC4	IRQ1 Enable High Bit	IRQ1ENH	00	78
FC5	IRQ1 Enable Low Bit	IRQ1ENL	00	79
FC6	Interrupt Request 2	IRQ2	00	75
FC7	IRQ2 Enable High Bit	IRQ2ENH	00	80
FC8	IRQ2 Enable Low Bit	IRQ2ENL	00	81
FC9–FCC	Reserved	—	XX	
FCD	Interrupt Edge Select	IRQES	00	82
FCE	Shared Interrupt Select	IRQSS	00	82
FCF	Interrupt Control	IRQCTL	00	83
GPIO Port A				
FD0	Port A Address	PAADDR	00	58
FD1	Port A Control	PACTL	00	60

Notes:
1. XX=Undefined.
2. The Reserved space can be configured as General-Purpose Register File RAM depending on the user option bits (see the User Option Bits chapter on page 277) and the on-chip PRAM size (see the Ordering Information chapter on page 372). If the PRAM is programmed as General-Purpose Register File RAM on Reserved space, the starting address always begins immediately after the end of General-Purpose Register File RAM.

Chapter 5. Reset, Stop Mode Recovery and Low-Voltage Detection

The Reset Controller within the F1680 Series MCU controls Reset and Stop Mode Recovery operations and provides indication of low-voltage supply conditions. During the operation, the following events cause a Reset:

- Power-On Reset (POR)
- Voltage Brown-Out (VBO) protection
- Watchdog Timer (WDT) time-out (when configured by the WDT_RES Flash option bit to initiate a Reset)
- External $\overline{\text{RESET}}$ pin assertion (when the alternate RESET function is enabled by the GPIO register)
- On-Chip Debugger initiated Reset (OCDCTL[0] set to 1)

When the device is in STOP Mode, a Stop Mode Recovery is initiated by each of the following:

- Watchdog Timer time-out
- GPIO Port input pin transition on an enabled Stop Mode Recovery source
- Interrupt from a timer or comparator enabled for STOP Mode operation

The low-voltage detection circuitry on the device features the following:

- The low-voltage detection threshold level is user-defined
- It generates an interrupt when the supply voltage drops below a user-defined level

5.1. Reset Types

The F1680 Series MCU provides various types of Reset operation. Stop Mode Recovery is considered a form of Reset. Table 9 lists the types of Reset and their operating characteristics. The System Reset is longer, if the external crystal oscillator is enabled by the Flash option bits allowing additional time for oscillator start-up.

eZ8 CPU services the Timer interrupt request following the normal Stop Mode Recovery sequence.

5.3.3. Stop Mode Recovery Using Comparator Interrupt

If Comparator enabled for STOP Mode operation interrupts during STOP Mode, the device undergoes a Stop Mode Recovery sequence. In the Reset Status Register, the stop bit is set to 1. If the F1680 Series MCU is configured to respond to interrupts, the eZ8 CPU services the comparator interrupt request following the normal Stop Mode Recovery sequence.

5.3.4. Stop Mode Recovery Using GPIO Port Pin Transition

Each of the GPIO port pins can be configured as a Stop Mode Recovery input source. On any GPIO pin enabled as a Stop Mode Recovery source, a change in the input pin value (from High to Low or from Low to High) initiates Stop Mode Recovery. In the Reset Status Register, the stop bit is set to 1.

! **Caution:** In STOP Mode, the GPIO Port Input Data registers (PxIN) are disabled. The Port Input Data registers record the Port transition only if the signal stays on the Port pin until the end of the Stop Mode Recovery delay. As a result, short pulses on the Port pin can initiate Stop Mode Recovery without being written to the Port Input Data Register or without initiating an interrupt (if enabled for that pin).

5.3.5. Stop Mode Recovery Using External $\overline{\text{RESET}}$ Pin

When the F1680 Series MCU is in STOP Mode and the external $\overline{\text{RESET}}$ pin is driven Low, a System Reset occurs. Because of a glitch filter operating on the $\overline{\text{RESET}}$ pin, the Low pulse must be greater than the minimum width specified, or it is ignored. For details, see the Electrical Characteristics chapter on page 349.

5.4. Low-Voltage Detection

In addition to the VBO Reset described earlier, it is also possible to generate an interrupt when the supply voltage drops below a user-selected value. For more details about the available Low-Voltage Detection (LVD) threshold levels, see the Trim Option Bits at Address 0000H (TTEMP0) section on page 282.

When the supply voltage drops below the LVD threshold, the LVD bit of the RSTSTAT Register is set to 1. This bit remains 1 until the low-voltage condition elapses. Reading or

Table 18. Port Alternate Function Mapping, 28-Pin Parts[1,2] (Continued)

Port	Pin	Mnemonic	Alternate Function Description	Alternate Function Set Register AFS1
Port C	PC0	Reserved		AFS1[0]: 0
		ANA4/C0INP/LED	ADC or Comparator 0 Input (P), or LED drive	AFS1[0]: 1
	PC1	MISO	SPI Master In/Slave Out	AFS1[1]: 0
		ANA5/C0INN/LED	ADC or Comparator 0 Input (N), or LED Drive	AFS1[1]: 1
	PC2	$\overline{SS}$	SPI Slave Select	AFS1[2]: 0
		ANA6/LED	ADC Analog Input or LED Drive	AFS1[2]: 1
	PC3	C0OUT	Comparator 0 Output	AFS1[3]: 0
		LED	LED drive	AFS1[3]: 1
	PC4	MOSI	SPI Master Out/Slave In	AFS1[4]: 0
		LED	LED Drive	AFS1[4]: 1
	PC5	SCK	SPI Serial Clock	AFS1[5]: 0
		LED	LED Drive	AFS1[5]: 1
	PC6	T2IN/$\overline{T2OUT}$	Timer 2 Input/Timer 2 Output Complement	AFS1[6]: 0
		LED	LED Drive	AFS1[6]: 1
	PC7	T2OUT	Timer 2 Output	AFS1[7]: 0
		LED	LED Drive	AFS1[7]: 1
Port D	PD0	$\overline{RESET}$	External Reset	N/A

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

8.4.7. Interrupt Edge Select Register

The Interrupt Edge Select (IRQES) Register, shown in Table 49, determines whether an interrupt is generated for the rising edge or falling edge on the selected GPIO Port A or Port D input pin.

Table 49. Interrupt Edge Select Register (IRQES)

Bits	7	6	5	4	3	2	1	0
Field	IES7	IES6	IES5	IES4	IES3	IES2	IES1	IES0
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	FCDH							

Bit	Description
[7:0] IES*x*	**Interrupt Edge Select *x*** 0 = An interrupt request is generated on the falling edge of the PA*x* input or PD*x* input. 1 = An interrupt request is generated on the rising edge of the PA*x* input or PD*x* input; *x* indicates the specific GPIO port pin number (0–7).

8.4.8. Shared Interrupt Select Register

The Shared Interrupt Select (IRQSS) Register, shown in Table 50, determines the source of the PAD*x*S interrupts. The Shared Interrupt Select Register selects between Port A and alternate sources for the individual interrupts.

Table 50. Shared Interrupt Select Register (IRQSS)

Bits	7	6	5	4	3	2	1	0
Field	PA7VS	PA6CS	PA5CS	PAD4S	PAD3S	PAD2S	PAD1S	Reserved
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	FCEH							

Bit	Description
[7] PA7VS	**PA7/LVD Selection** 0 = PA7 is used for the interrupt for PA7VS interrupt request. 1 = The LVD is used for the interrupt for PA7VS interrupt request.
[6] PA6CS	**PA6/Comparator 0 Selection** 0 = PA6 is used for the interrupt for PA6CS interrupt request. 1 = The Comparator 0 is used for the interrupt for PA6CS interrupt request.

System clock is only for operation in ACTIVE and HALT modes. System clock is software selectable in Oscillator Control Module as external high-frequency crystal or internal precision oscillator. The TCLKS field in the Timer Control 2 Register selects the timer clock source.

! Caution: When the timer is operating on a peripheral clock, the timer clock is asynchronous to the CPU clock. To ensure error-free operation, disable the timer before modifying its operation (also include changing the timer clock source). Therefore, any write to the timer control registers cannot be performed when the timer is enabled and a peripheral clock is used.

When the timer uses a peripheral clock and the timer is enabled, any read from TxH or TxL is not recommended, because the results can be unpredictable. Disable the timer first, then read it. If the timer works in the CAPTURE, CAPTURE/COMPARE, CAPTURE RESTART or DEMODULATION modes, any read from TxPWM0H, TxPWM0L, TxPWM1H, TxPWM1L or TxSTAT must be performed after a capture interrupt occurs; otherwise, results can be unpredictable. The INPCAP bit of the Timer Control 0 Register is the same as these PWM registers. When the timer uses the main clock, you can write/read all timer registers at any time.

9.2.2. Low-Power Modes

Timers can operate in both HALT Mode and STOP Mode.

9.2.2.1. Operation in HALT Mode

When the eZ8 CPU enters HALT Mode, the timer will continue to operate if enabled. To minimize current in HALT Mode, the timer can be disabled by clearing the TEN control bit. The noise filter, if enabled, will also continue to operate in HALT Mode and rejects any noise on the timer input pin.

9.2.2.2. Operation in STOP Mode

When the eZ8 CPU enters STOP Mode, the timer continues to operate if enabled and peripheral clock is chosen as the clock source. In STOP Mode, the timer interrupt (if enabled) automatically initiates a Stop Mode Recovery and generates an interrupt request. In the Reset Status Register, the stop bit is set to 1. Also, timer interrupt request bit in Interrupt Request 0 register is set. Following completion of the Stop Mode Recovery, if interrupts are enabled, the CPU responds to the interrupt request by fetching the timer interrupt vector. The noise filter, if enabled, will also continue to operate in STOP Mode and rejects any noise on the timer input pin.

In COMPARE Mode, the Timer Reload High and Low Byte registers store the 16-bit Compare value.

Table 57. Timer 0–2 Reload High Byte Register (TxRH)

Bit	7	6	5	4	3	2	1	0
Field	TRH							
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	F02H, F0AH, F12H							

Table 58. Timer 0–2 Reload Low Byte Register (TxRL)

Bit	7	6	5	4	3	2	1	0
Field	TRL							
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	F03H, F0BH, F13H							

Bit	Description
[7:0] TRH, TRL	**Timer Reload Register High and Low** These two bytes form the 16-bit reload value, {TRH[7:0], TRL[7:0]}. This value is used to set the maximum count value which initiates a timer reload to 0001H. In COMPARE Mode, these two bytes form the 16-bit Compare value.

9.3.3. Timer 0–2 PWM0 High and Low Byte Registers

The Timer 0–2 PWM0 High and Low Byte (TxPWM0H and TxPWM0L) registers, shown in Tables 59 and 60, are used for Pulse Width Modulator (PWM) operations. These registers also store the Capture values for the CAPTURE, CAPTURE/COMPARE and DEMODULATION Modes. When the timer is enabled, writes to these registers are buffered, and loading of the registers is delayed until a timer reload to `0001H` occurs – that is, unless PWM0UE = 1.

Table 59. Timer 0–2 PWM0 High Byte Register (TxPWM0H)

Bit	7	6	5	4	3	2	1	0
Field	PWM0H							
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	F04H, F0CH, F14H							

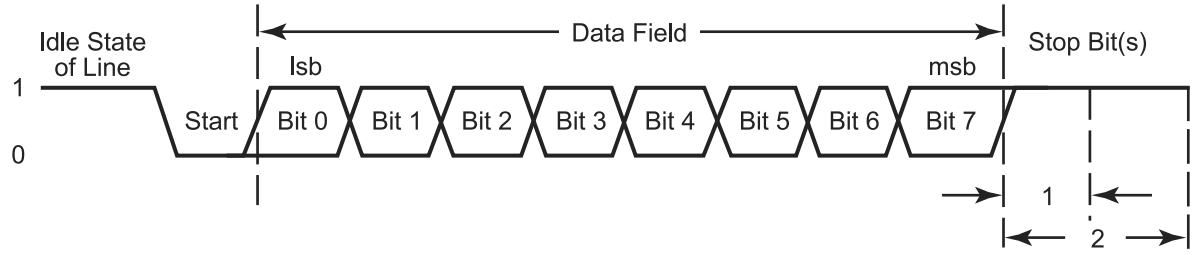


Figure 20. LIN-UART Asynchronous Data Format without Parity

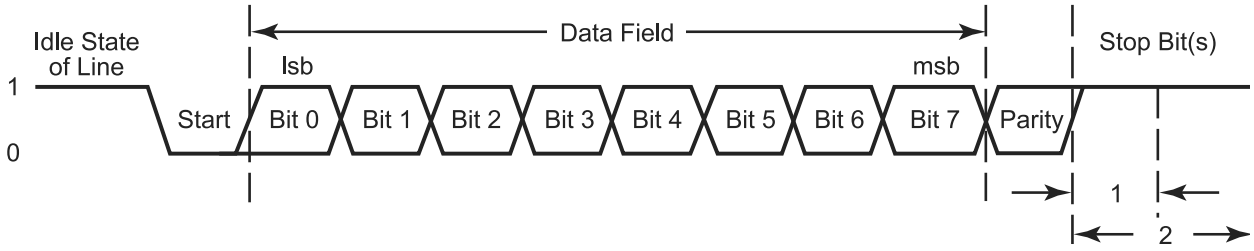


Figure 21. LIN-UART Asynchronous Data Format with Parity

12.1.2. Transmitting Data using the Polled Method

Observe the following steps to transmit data using the polled-operating method:

1. Write to the LIN-UART Baud Rate High and Low Byte registers to set the appropriate baud rate.
2. Enable the LIN-UART pin functions by configuring the associated GPIO port pins for alternate-function operation.
3. If MULTIPROCESSOR Mode is appropriate, write to the LIN-UART Control 1 Register to enable MULTIPROCESSOR (9-bit) Mode functions.
4. Set the MULTIPROCESSOR Mode Select bit (MPEN) to enable MULTIPROCESSOR Mode.
5. Write to the LIN-UART Control 0 Register to:
   a. Set the Transmit Enable bit (TEN) to enable the LIN-UART for data transmission.
   b. If parity is appropriate and MULTIPROCESSOR Mode is not enabled, set the parity enable bit (PEN) and select either even-or-odd parity (PSEL).
   c. Set or clear the CTSE bit to enable or disable control from the remote receiver using the $\overline{\text{CTS}}$ pin.

Chapter 13. Infrared Encoder/Decoder

The Z8 Encore! XP F1680 Series products contain a fully-functional, high-performance UART to infrared encoder/decoder (endec). The infrared endec is integrated with an on-chip UART to allow easy communication between the Z8 Encore! and IrDA Physical Layer Specification and version 1.3-compliant infrared transceivers. Infrared communication provides secure, reliable, low-cost, point-to-point communication between PCs, PDAs, cell phones, printers and other infrared enabled devices.

13.1. Architecture

Figure 27 displays the architecture of the infrared endec.

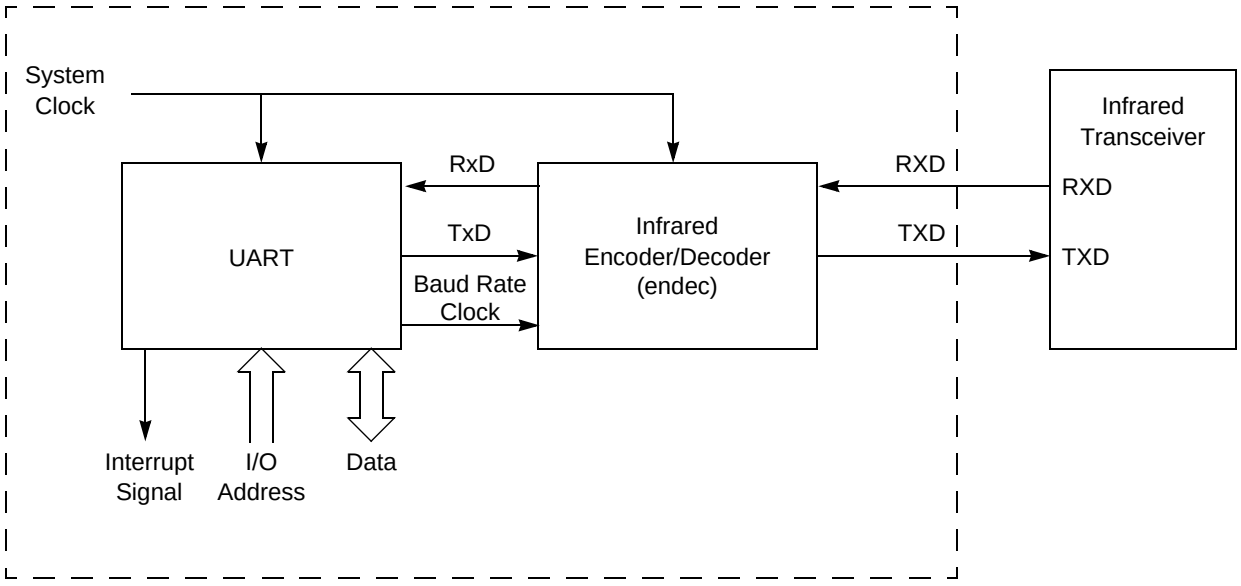


Figure 27. Infrared Data Communication System Block Diagram

13.2. Operation

When the infrared endec is enabled, the transmit data from the associated on-chip UART is encoded as digital signals in accordance with the IrDA standard and output to the infrared transceiver through the TXD pin. Likewise, data received from the infrared transceiver is passed to the infrared endec through the RXD pin, decoded by the infrared

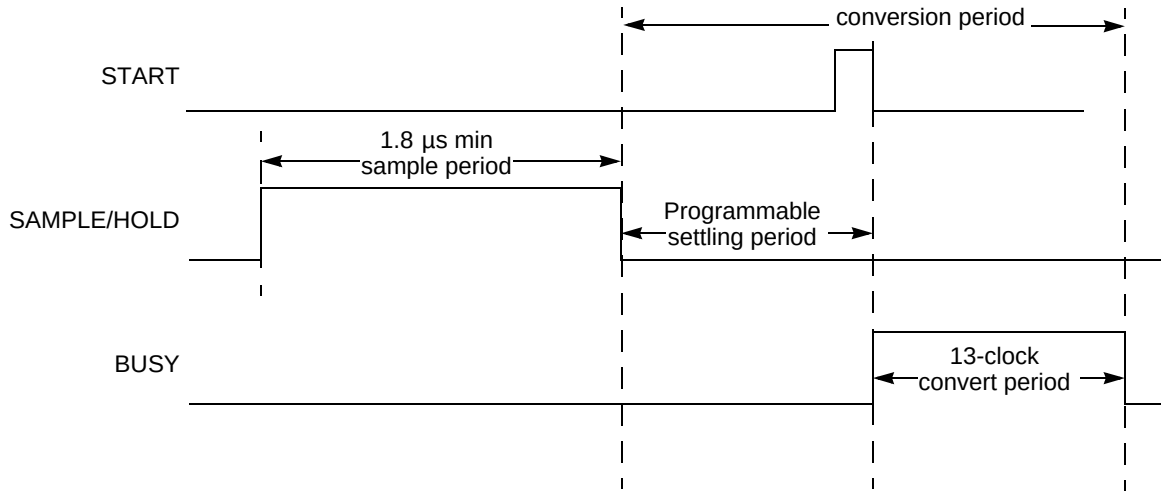


Figure 31. ADC Timing Diagram

14.2.2. ADC Interrupt

The ADC can generate an interrupt request when a conversion is completed. An interrupt request that is pending when the ADC is disabled is not automatically cleared. See Figure 32.

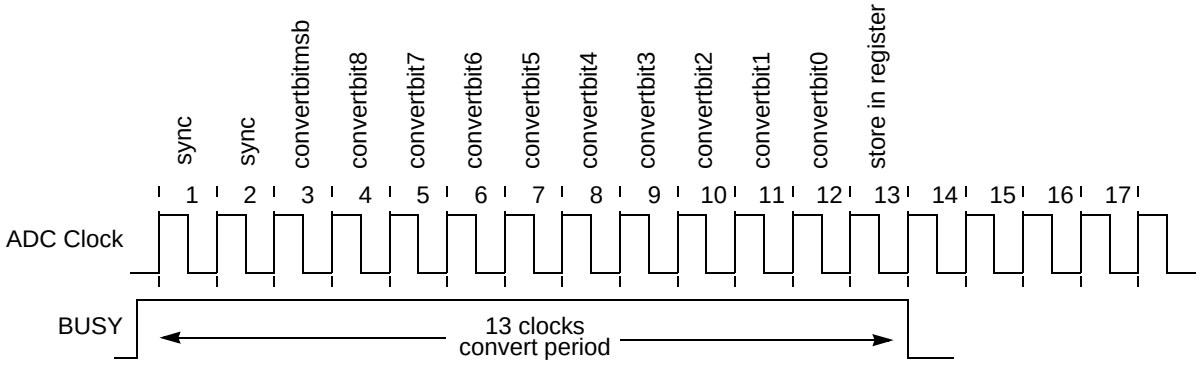


Figure 32. ADC Convert Timing

14.2.3. Reference Buffer

The reference buffer, RBUF, supplies the reference voltage for the ADC. When enabled, the internal voltage reference generator supplies the ADC and this voltage is available on

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.

Bit	Description (Continued)
[2] ARBLST	**Arbitration Lost** This bit is set when the I^2C controller is enabled in MASTER Mode and loses arbitration (outputs a 1 on SDA and receives a 0 on SDA). The ARBLST bit clears when the I2CISTAT Register is read.
[1] SPRS	**Stop/Restart Condition Interrupt** This bit is set when the I^2C controller is enabled in SLAVE Mode and detects a stop or restart condition during a transaction directed to this slave. This bit clears when the I2CISTAT Register is read. Read the RSTR bit of the I2CSTATE Register to determine whether the interrupt was caused by a stop or restart condition.
[0] NCKI	**NAK Interrupt** In MASTER Mode, this bit is set when a Not Acknowledge condition is received or sent and neither the start nor the stop bit is active. In MASTER Mode, this bit can only be cleared by setting the start or stop bits. In SLAVE Mode, this bit is set when a Not Acknowledge condition is received (Master reading data from Slave), indicating the master is finished reading. A stop or restart condition follows. In SLAVE Mode this bit clears when the I2CISTAT Register is read.

23.2.6. Automatic Reset

The Z8 Encore! XP F1680 Series devices have the capability to switch clock sources during operation. If the Autobaud is set and the clock source is switched, the Autobaud value becomes invalid. A new Autobaud value must be configured with the new clock frequency.

The oscillator control logic has clock switch detection. If a clock switch is detected and the Autobaud is set, the device will automatically send a Serial Break for 4096 clocks. This will reset the Autobaud and indicate to the host that a new Autobaud character should be sent.

23.2.7. Transmit Flow Control

Transmit flow control is implemented by the use of a remote start bit. When transmit flow control is enabled, the transmitter will wait for the remote host to send the start bit. Transmit flow control is useful in applications where receive overruns can occur.

The remote host can transmit a remote start bit by sending the character FFH. The transmitter will append its data after the start bit. Due to the *wire-and* nature of the open drain bus, the start bit sent by the remote host and the data bits sent by the Z8 Encore! XP F1680 Series device appear as one character; see Figure 61.

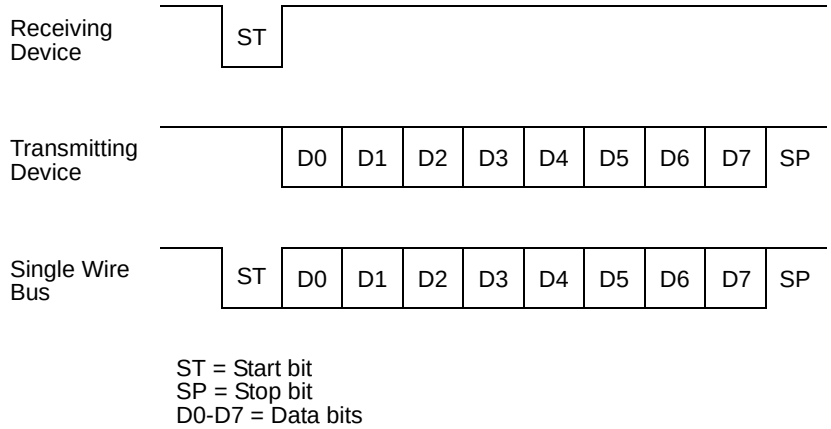


Figure 61. Start Bit Flow Control

23.2.8. Breakpoints

Execution breakpoints are generated using the BRK instruction (op code 00H). When the eZ8 CPU decodes a BRK instruction, it signals the On-Chip Debugger. If breakpoints are enabled, the OCD idles the eZ8 CPU and enters DEBUG mode. If breakpoints are not

enabled, the OCD ignores the BRK signal and the BRK instruction operates as an NOP instruction.

If breakpoints are enabled, the OCD can be configured to automatically enter DEBUG mode, or to loop on the break instruction. If the OCD is configured to loop on the BRK instruction, then the CPU remains able to service interrupt requests.

The loop on a BRK instruction can service interrupts in the background. For interrupts to be serviced in the background, there cannot be any breakpoints in the interrupt service routine. Otherwise, the CPU stops on the breakpoint in the interrupt routine. For interrupts to be serviced in the background, interrupts must also be enabled. Interrupts are typically disabled during critical sections of code where interrupts do not occur (such as adjusting the stack pointer or modifying shared data).

Through the OCD, host debugger software can poll the IDLE bit of the OCDSTAT Register to determine if the OCD is looping on a BRK instruction. When the host must stop the CPU on the BRK instruction on which it is looping, the host must not set the DBGMODE bit of the OCDCTL register. The CPU may have vectored to an interrupt service routine. Instead, the host clears the BRKLOOP bit, thereby allowing the CPU to finish the interrupt service routine and return to the BRK instruction. When the CPU returns to the BRK instruction on which it was previously looping, it automatically sets the DBGMODE bit and enters DEBUG mode.

The majority of the OCD commands remain disabled when the eZ8 CPU is looping on a BRK instruction. The eZ8 CPU must be in DEBUG mode before these commands can be issued.

23.2.8.1. Breakpoints in Flash Memory

The BRK instruction is op code `00H`, which corresponds to the fully programmed state of a byte in Flash memory. To implement a breakpoint, write `00H` to the appropriate address, overwriting the current instruction. To remove a breakpoint, erase the corresponding page of Flash memory and reprogram with the original data.

23.2.9. OCDCNTR Register

The On-Chip Debugger contains a multipurpose 16-bit Counter Register. It can be used for the following:

- Count system clock cycles between breakpoints
- Generate a BRK when it counts down to 0
- Generate a BRK when its value matches the Program Counter

When configured as a counter, the OCDCNTR Register starts counting when the On-Chip Debugger exits DEBUG mode and stops counting when it enters DEBUG mode again or

! **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.

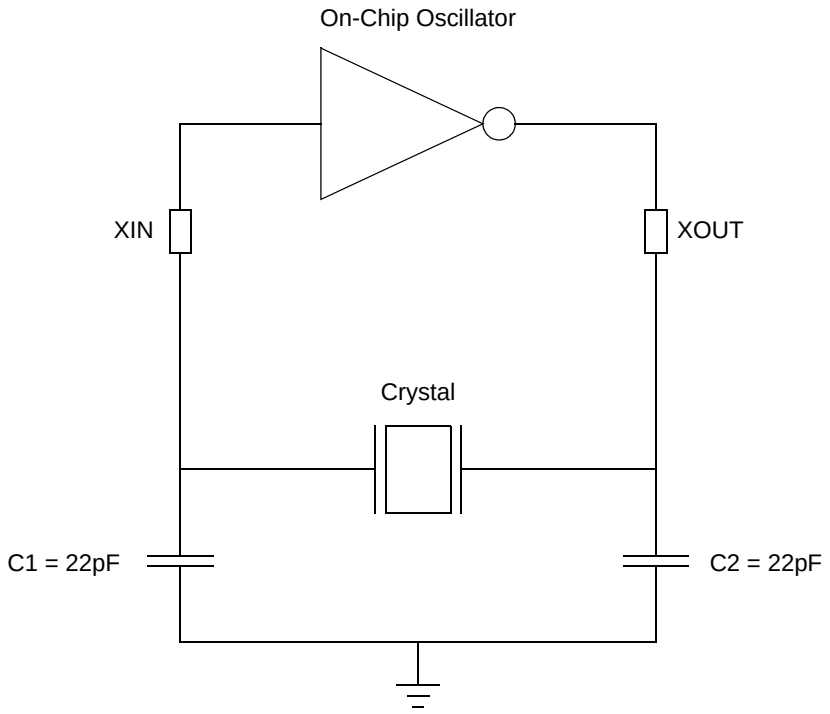


Figure 65. Recommended 32kHz Crystal Oscillator Configuration

Figures 67 and 68 provide information about each of the eZ8 CPUinstructions.

Upper Nibble (Hex) \ Lower Nibble (Hex)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	1.1 **BRK**	2.2 **SRP** IM	2.3 **ADD** r1,r2	2.4 **ADD** r1,Ir2	3.3 **ADD** R2,R1	3.4 **ADD** IR2,R1	3.3 **ADD** R1,IM	3.4 **ADD** IR1,IM	4.3 **ADDX** ER2,ER1	4.3 **ADDX** IM,ER1	2.3 **DJNZ** r1,X	2.2 **JR** cc,X	2.2 **LD** r1,IM	3.2 **JP** cc,DA	1.2 **INC** r1	1.2 **NOP**
1	2.2 **RLC** R1	2.3 **RLC** IR1	2.3 **ADC** r1,r2	2.4 **ADC** r1,Ir2	3.3 **ADC** R2,R1	3.4 **ADC** IR2,R1	3.3 **ADC** R1,IM	3.4 **ADC** IR1,IM	4.3 **ADCX** ER2,ER1	4.3 **ADCX** IM,ER1						See 2nd Op Code Map
2	2.2 **INC** R1	2.3 **INC** IR1	2.3 **SUB** r1,r2	2.4 **SUB** r1,Ir2	3.3 **SUB** R2,R1	3.4 **SUB** IR2,R1	3.3 **SUB** R1,IM	3.4 **SUB** IR1,IM	4.3 **SUBX** ER2,ER1	4.3 **SUBX** IM,ER1						1, 2 ATM
3	2.2 **DEC** R1	2.3 **DEC** IR1	2.3 **SBC** r1,r2	2.4 **SBC** r1,Ir2	3.3 **SBC** R2,R1	3.4 **SBC** IR2,R1	3.3 **SBC** R1,IM	3.4 **SBC** IR1,IM	4.3 **SBCX** ER2,ER1	4.3 **SBCX** IM,ER1						
4	2.2 **DA** R1	2.3 **DA** IR1	2.3 **OR** r1,r2	2.4 **OR** r1,Ir2	3.3 **OR** R2,R1	3.4 **OR** IR2,R1	3.3 **OR** R1,IM	3.4 **OR** IR1,IM	4.3 **ORX** ER2,ER1	4.3 **ORX** IM,ER1						
5	2.2 **POP** R1	2.3 **POP** IR1	2.3 **AND** r1,r2	2.4 **AND** r1,Ir2	3.3 **AND** R2,R1	3.4 **AND** IR2,R1	3.3 **AND** R1,IM	3.4 **AND** IR1,IM	4.3 **ANDX** ER2,ER1	4.3 **ANDX** IM,ER1						1.2 **WDT**
6	2.2 **COM** R1	2.3 **COM** IR1	2.3 **TCM** r1,r2	2.4 **TCM** r1,Ir2	3.3 **TCM** R2,R1	3.4 **TCM** IR2,R1	3.3 **TCM** R1,IM	3.4 **TCM** IR1,IM	4.3 **TCMX** ER2,ER1	4.3 **TCMX** IM,ER1						1.2 **STOP**
7	2.2 **PUSH** R2	2.3 **PUSH** IR2	2.3 **TM** r1,r2	2.4 **TM** r1,Ir2	3.3 **TM** R2,R1	3.4 **TM** IR2,R1	3.3 **TM** R1,IM	3.4 **TM** IR1,IM	4.3 **TMX** ER2,ER1	4.3 **TMX** IM,ER1						1.2 **HALT**
8	2.5 **DECW** RR1	2.6 **DECW** IRR1	2.5 **LDE** r1,Irr2	2.9 **LDEI** Ir1,Irr2	3.2 **LDX** r1,ER2	3.3 **LDX** Ir1,ER2	3.4 **LDX** IRR2,R1	3.5 **LDX** IRR2,IR1	3.4 **LDX** r1,rr2,X	3.4 **LDX** rr1,r2,X						1.2 **DI**
9	2.2 **RL** R1	2.3 **RL** IR1	2.5 **LDE** r2,Irr1	2.9 **LDEI** Ir2,Irr1	3.2 **LDX** r2,ER1	3.3 **LDX** Ir2,ER1	3.4 **LDX** R2,IRR1	3.5 **LDX** IR2,IRR1	3.3 **LEA** r1,r2,X	3.5 **LEA** rr1,rr2,X						1.2 **EI**
A	2.5 **INCW** RR1	2.6 **INCW** IRR1	2.3 **CP** r1,r2	2.4 **CP** r1,Ir2	3.3 **CP** R2,R1	3.4 **CP** IR2,R1	3.3 **CP** R1,IM	3.4 **CP** IR1,IM	4.3 **CPX** ER2,ER1	4.3 **CPX** IM,ER1						1.4 **RET**
B	2.2 **CLR** R1	2.3 **CLR** IR1	2.3 **XOR** r1,r2	2.4 **XOR** r1,Ir2	3.3 **XOR** R2,R1	3.4 **XOR** IR2,R1	3.3 **XOR** R1,IM	3.4 **XOR** IR1,IM	4.3 **XORX** ER2,ER1	4.3 **XORX** IM,ER1						1.5 **IRET**
C	2.2 **RRC** R1	2.3 **RRC** IR1	2.5 **LDC** r1,Irr2	2.9 **LDCI** Ir1,Irr2	2.3 **JP** IRR1	2.9 **LDC** Ir1,Irr2		3.4 **LD** r1,r2,X	3.2 **PUSHX** ER2							1.2 **RCF**
D	2.2 **SRA** R1	2.3 **SRA** IR1	2.5 **LDC** r2,Irr1	2.9 **LDCI** Ir2,Irr1	2.6 **CALL** IRR1	2.2 **BSWAP** R1	3.3 **CALL** DA	3.4 **LD** r2,r1,X	3.2 **POPX** ER1							1.2 **SCF**
E	2.2 **RR** R1	2.3 **RR** IR1	2.2 **BIT** p,b,r1	2.3 **LD** r1,Ir2	3.2 **LD** R2,R1	3.3 **LD** IR2,R1	3.2 **LD** R1,IM	3.3 **LD** IR1,IM	4.2 **LDX** ER2,ER1	4.2 **LDX** IM,ER1						1.2 **CCF**
F	2.2 **SWAP** R1	2.3 **SWAP** IR1	2.6 **TRAP** Vector	2.3 **LD** Ir1,r2	2.8 **MULT** RR1	3.3 **LD** R2,IR1	3.3 **BTJ** p,b,r1,X	3.4 **BTJ** p,b,Ir1,X								

Figure 67. First Op Code Map

Table 201. Low Voltage Detect Electrical Characteristics (Continued)

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

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

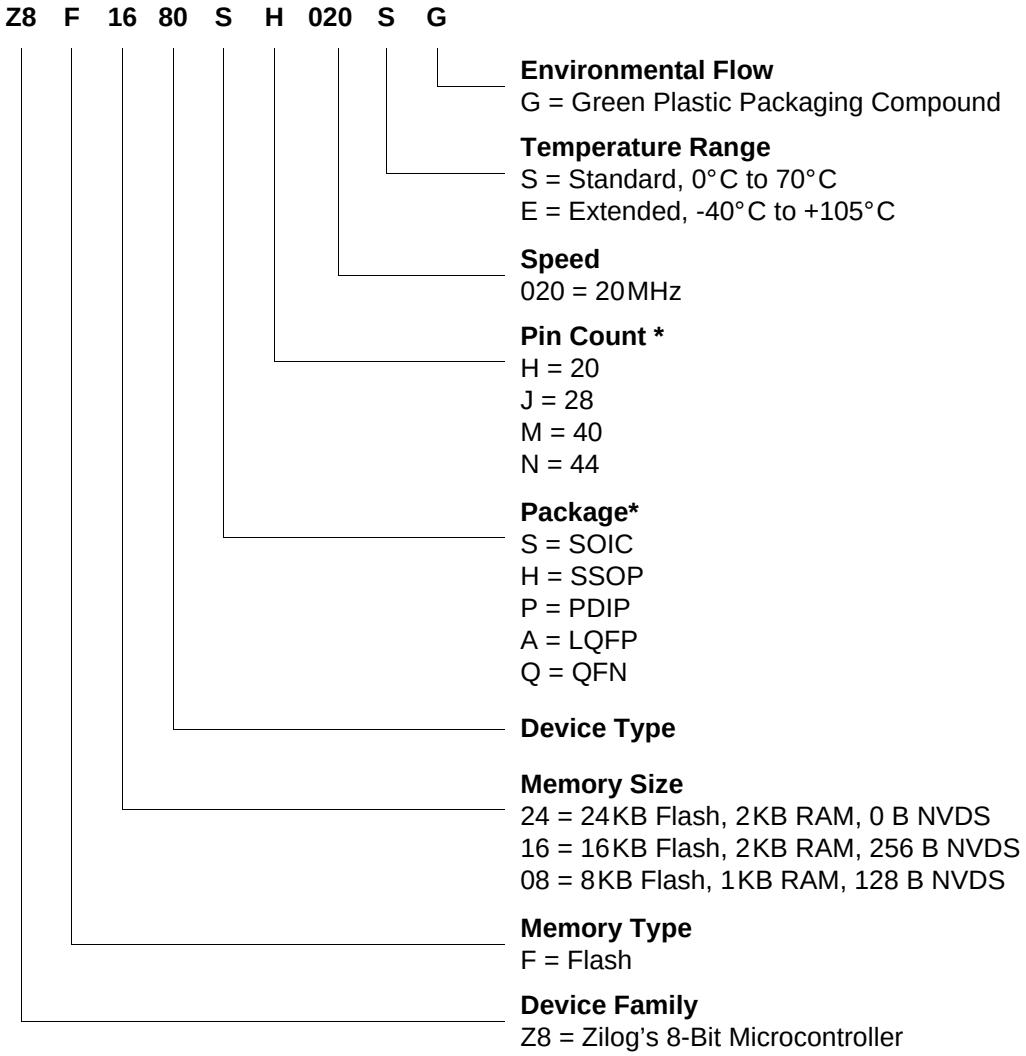
Table 202. Crystal Oscillator Characteristics

		T_A = 0°C to +70°C T_A = –40°C to +105°C							
		V_{DD} = 2.7 to 3.6V			V_{DD} = 1.8 to 2.7V				
Symbol	**Parameter**	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**	**Units**	**Conditions**
I_{DD}XTAL	Crystal Oscillator Active Supply Current	–	–	500	–	–	300	µA	
I_{DDQ}XTAL	Crystal Oscillator Quiescent Current	–	5	–	–	5	–	nA	
S_{CLK}	Clk_out State in Crystal Disable	1	1	1	1	1	1		
F_{XTAL}	External Crystal Oscillator Frequency	1	–	20	1	–	20	MHz	See Figure 74.
T_{SET}	Startup Time After Enable	–	10,000	30,000	–	10,000	30,000	Cycle	
	Clk_out Duty Cycle	40	50	60	40	50	60	%	
	Clk_out Jitter	–	1	–	–	1	–	%	

31.1. Part Number Suffix Designations

Zilog part numbers consist of a number of components, as indicated in the following example.

Example. Part number Z8F1680SH020SG is an 8-bit, 20 MHz Flash Motor Controller with 16 KB of Program memory in a 20-pin SOIC package, operating within a 0ºC to +70ºC temperature range and built using lead-free solder.



S