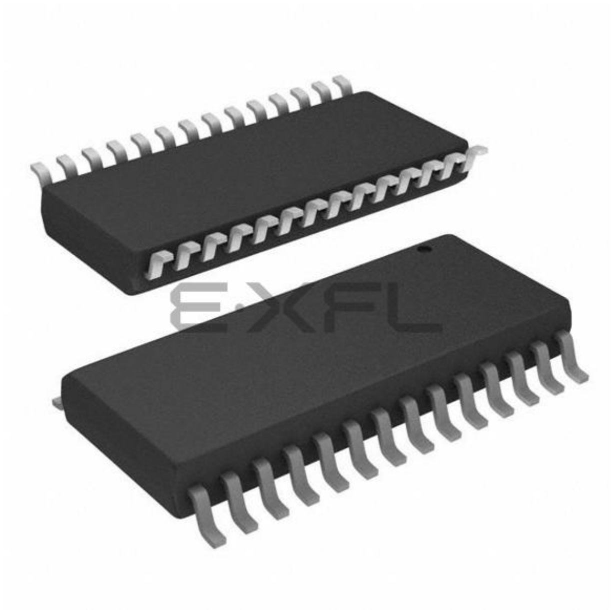


What is "Embedded - Microcontrollers"?

"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	5MHz
Connectivity	IrDA, UART/USART
Peripherals	Brown-out Detect/Reset, LED, POR, PWM, WDT
Number of I/O	22
Program Memory Size	4KB (4K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SOIC (0.295", 7.50mm Width)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f0423sj005ec

Receiving Data using the Polled Method 97
Receiving Data using the Interrupt-Driven Method 98
Clear To Send (CTS) Operation 99
MULTIPROCESSOR (9-Bit) Mode 99
External Driver Enable 101
UART Interrupts 101
UART Baud Rate Generator 103
UART Control Register Definitions 104
UART Transmit Data Register 104
UART Receive Data Register 105
UART Status 0 Register 105
UART Status 1 Register 106
UART Control 0 and Control 1 Registers 107
UART Address Compare Register 109
UART Baud Rate High and Low Byte Registers 110
Infrared Encoder/Decoder 113
Architecture 113
Operation 113
Transmitting IrDA Data 114
Receiving IrDA Data 115
Infrared Encoder/Decoder Control Register Definitions 116
Analog-to-Digital Converter 117
Architecture 117
Operation 118
Data Format 118
Automatic Powerdown 119
Single-Shot Conversion 119
Continuous Conversion 120
Interrupts 121
Calibration and Compensation 121
ADC Control Register Definitions 122
ADC Control Register 0 122
ADC Control/Status Register 1 124
ADC Data High Byte Register 124
ADC Data Low Bits Register 125
Comparator 127
Operation 127
Comparator Control Register Definitions 127
Flash Memory 129

Block Diagram

Figure 1 on page 3 displays the block diagram of the architecture of Z8 Encore! XP F0823 Series devices.

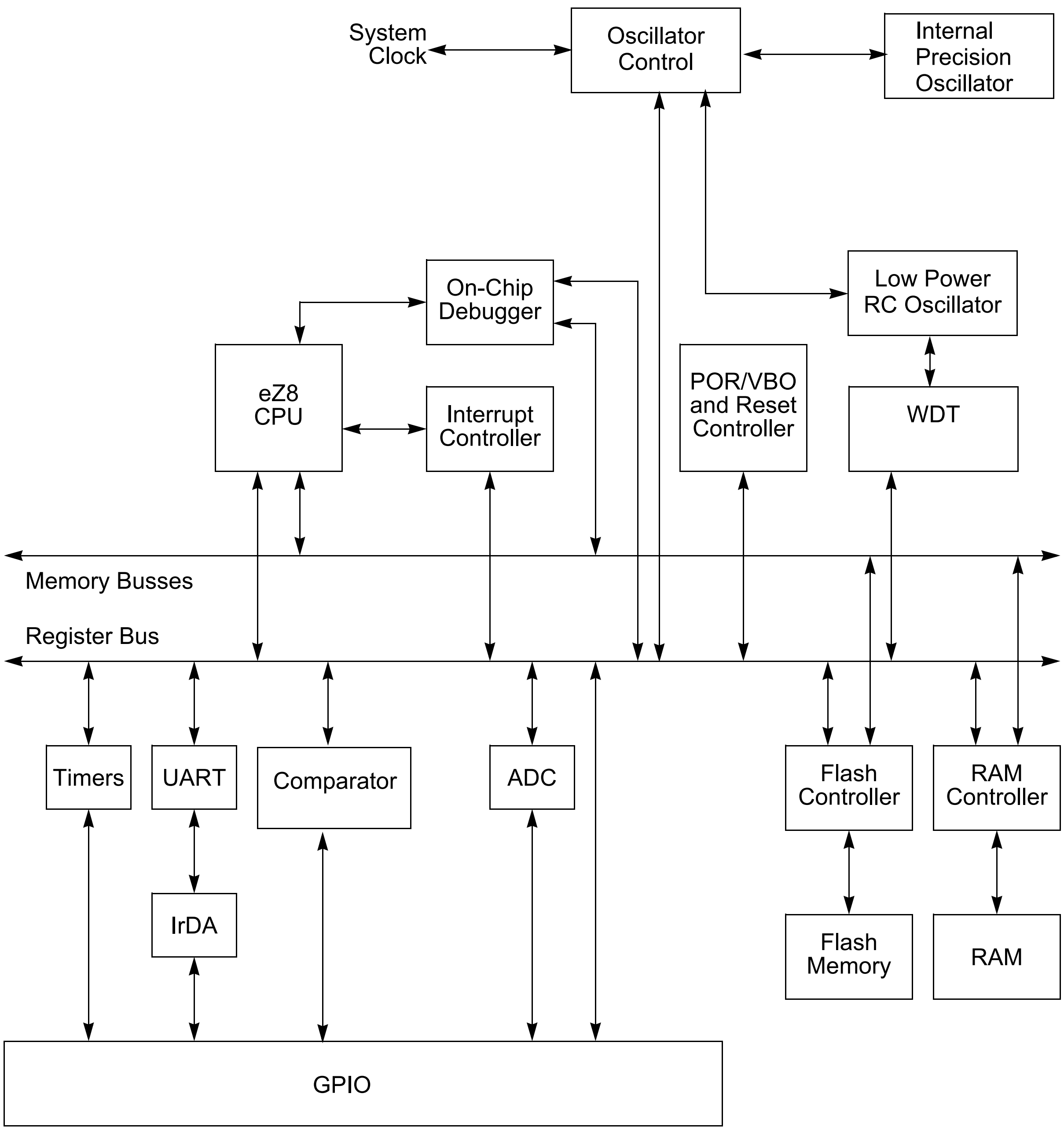


Figure 1. Z8 Encore! XP® F0823 Series Block Diagram

CPU and Peripheral Overview

eZ8 CPU Features

The eZ8 CPU, Zilog's latest 8-bit central processing unit (CPU), meets the continuing demand for faster and code-efficient microcontrollers. The eZ8 CPU executes a superset of the original Z8® instruction set. The eZ8 CPU features include:

- Direct register-to-register architecture allows each register to function as an accumulator, improving execution time and decreasing the required program memory.
- Software stack allows much greater depth in subroutine calls and interrupts than hardware stacks.
- Compatible with existing Z8 code.
- Expanded internal Register File allows access of up to 4 KB.
- New instructions improve execution efficiency for code developed using higher-level programming languages, including C.
- Pipelined instruction fetch and execution.
- New instructions for improved performance including BIT, BSWAP, BTJ, CPC, LDC, LDCI, LEA, MULT, and SRL.
- New instructions support 12-bit linear addressing of the Register file.
- Up to 10 MIPS operation.
- C-Compiler friendly.
- 2 to 9 clock cycles per instruction.

For more information on eZ8 CPU, refer to *eZ8 CPU Core User Manual (UM0128)* available for download at www.zilog.com.

General-Purpose I/O

Z8 Encore! XP F0823 Series features 6 to 24 port pins (Ports A–C) for general-purpose I/O (GPIO). The number of GPIO pins available is a function of package. Each pin is individually programmable. 5 V tolerant input pins are available on all I/Os on 8-pin devices, most I/Os on other package types.

Flash Controller

The Flash Controller programs and erases Flash memory. The Flash Controller supports protection against accidental program and erasure, as well as factory serialization and read protection.

Table 8. Register File Address Map (Continued)

Address (Hex)	Register Description	Mnemonic	Reset (Hex)	Page No
FF2	Watchdog Timer Reload High Byte	WDTH	FF	91
FF3	Watchdog Timer Reload Low Byte	WDTL	FF	91
FF4–FF5	Reserved	—	XX	
Trim Bit Control				
FF6	Trim Bit Address	TRMADR	00	143
FF7	Trim Data	TRMDR	XX	144
Flash Memory Controller				
FF8	Flash Control	FCTL	00	137
FF8	Flash Status	FSTAT	00	137
FF9	Flash Page Select	FPS	00	138
	Flash Sector Protect	FPROT	00	139
FFA	Flash Programming Frequency High Byte	FFREQH	00	140
FFB	Flash Programming Frequency Low Byte	FFREQL	00	140
eZ8 CPU				
FFC	Flags	—	XX	Refer to *eZ8* CPU *Core User Manual (UM0128)*
FFD	Register Pointer	RP	XX	
FFE	Stack Pointer High Byte	SPH	XX	
FFF	Stack Pointer Low Byte	SPL	XX	

XX=Undefined

Table 15. Port Alternate Function Mapping (Non 8-Pin Parts) (Continued)

Port	Pin	Mnemonic	Alternate Function Description	Alternate Function Set Register AFS1
Port C	PC0	Reserved		AFS1[0]: 0
		ANA4/CINP/LED Drive	ADC or Comparator Input, or LED drive	AFS1[0]: 1
	PC1	Reserved		AFS1[1]: 0
		ANA5/CINN/ LED Drive	ADC or Comparator Input, or LED drive	AFS1[1]: 1
	PC2	Reserved		AFS1[2]: 0
		ANA6/LED/ VREF*	ADC Analog Input or LED Drive or ADC Voltage Reference	AFS1[2]: 1
	PC3	COUT	Comparator Output	AFS1[3]: 0
		LED	LED drive	AFS1[3]: 1
	PC4	Reserved		AFS1[4]: 0
		LED	LED Drive	AFS1[4]: 1
	PC5	Reserved		AFS1[5]: 0
		LED	LED Drive	AFS1[5]: 1
	PC6	Reserved		AFS1[6]: 0
		LED	LED Drive	AFS1[6]: 1
	PC7	Reserved		AFS1[7]: 0
		LED	LED Drive	AFS1[7]: 1

Note: Because there are at most two choices of alternate function for any pin of Port C, the Alternate Function Set register AFS2 is implemented but not used to select the function. Also, Alternate Function selection as described in Port A–C Alternate Function Sub-Registers must also be enabled.
*VREF is available on PC2 in 20-pin parts only.

Interrupt Control Register Definitions

For all interrupts other than the Watchdog Timer interrupt, the Primary Oscillator Fail Trap, and the Watchdog Timer Oscillator Fail Trap, the interrupt control registers enable individual interrupts, set interrupt priorities, and indicate interrupt requests.

Interrupt Request 0 Register

The Interrupt Request 0 (IRQ0) register (Table 34) stores the interrupt requests for both vectored and polled interrupts. When a request is presented to the interrupt controller, the corresponding bit in the IRQ0 register becomes 1. If interrupts are globally enabled (vectored interrupts), the interrupt controller passes an interrupt request to the eZ8 CPU. If interrupts are globally disabled (polled interrupts), the eZ8 CPU reads the Interrupt Request 0 register to determine if any interrupt requests are pending.

Table 34. Interrupt Request 0 Register (IRQ0)

BITS	7	6	5	4	3	2	1	0
FIELD	Reserved	T1I	T0I	U0RXI	U0TXI	Reserved	Reserved	ADCI
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
ADDR	FC0H							

Reserved—Must be 0

T1I—Timer 1 Interrupt Request
0 = No interrupt request is pending for Timer 1
1 = An interrupt request from Timer 1 is awaiting service

T0I—Timer 0 Interrupt Request
0 = No interrupt request is pending for Timer 0
1 = An interrupt request from Timer 0 is awaiting service

U0RXI—UART 0 Receiver Interrupt Request
0 = No interrupt request is pending for the UART 0 receiver
1 = An interrupt request from the UART 0 receiver is awaiting service

U0TXI—UART 0 Transmitter Interrupt Request
0 = No interrupt request is pending for the UART 0 transmitter
1 = An interrupt request from the UART 0 transmitter is awaiting service

ADCI—ADC Interrupt Request
0 = No interrupt request is pending for the ADC
1 = An interrupt request from the ADC is awaiting service

Table 48. Interrupt Control Register (IRQCTL)

BITS	7	6	5	4	3	2	1	0
FIELD	IRQE	Reserved						
RESET	0	0	0	0	0	0	0	0
R/W	R/W	R	R	R	R	R	R	R
ADDR	FCFH							

IRQE—Interrupt Request Enable
This bit is set to 1 by executing an `EI` (Enable Interrupts) or `IRET` (Interrupt Return) instruction, or by a direct register write of a 1 to this bit. It is reset to 0 by executing a DI instruction, eZ8 CPU acknowledgement of an interrupt request, reset or by a direct register write of a 0 to this bit.
0 = Interrupts are disabled
1 = Interrupts are enabled

Reserved—0 when read

timer value is not reset to 0001H). Also, if the Timer Output alternate function is enabled, the Timer Output pin changes state (from Low to High or from High to Low) upon Compare.

If the Timer reaches FFFFH, the timer rolls over to 0000H and continue counting. Follow the steps below to configure a timer for COMPARE mode and to initiate the count:

1. Write to the Timer Control register to:
   - Disable the timer.
   - Configure the timer for Compare mode.
   - Set the prescale value.
   - Set the initial logic level (High or Low) for the Timer Output alternate function, if appropriate.
2. Write to the Timer High and Low Byte registers to set the starting count value.
3. Write to the Timer Reload High and Low Byte registers to set the Compare value.
4. Enable the timer interrupt, if appropriate, and set the timer interrupt priority by writing to the relevant interrupt registers.
5. If using the Timer Output function, configure the associated GPIO port pin for the Timer Output alternate function.
6. Write to the Timer Control register to enable the timer and initiate counting.

In COMPARE mode, the system clock always provides the timer input. The Compare time can be calculated by the following equation:

$$\text{COMPARE Mode Time (s)} = \frac{(\text{Compare Value} - \text{Start Value}) \times \text{Prescale}}{\text{System Clock Frequency (Hz)}}$$

GATED Mode

In GATED mode, the timer counts only when the Timer Input signal is in its active state (asserted), as determined by the TPOL bit in the Timer Control register. When the Timer Input signal is asserted, counting begins. A timer interrupt is generated when the Timer Input signal is deasserted or a timer Reload occurs. To determine if a Timer Input signal deassertion generated the interrupt, read the associated GPIO input value and compare to the value stored in the TPOL bit.

The timer counts up to the 16-bit Reload value stored in the Timer Reload High and Low Byte registers. The timer input is the system clock. When reaching the Reload value, the timer generates an interrupt, the count value in the Timer High and Low Byte registers is reset to 0001H and counting resumes (assuming the Timer Input signal remains asserted). Also, if the Timer Output alternate function is enabled, the Timer Output pin changes state (from Low to High or from High to Low) at timer reset.

Receiving Data using the Interrupt-Driven Method

The UART Receiver interrupt indicates the availability of new data (as well as error conditions). Follow the steps below to configure the UART receiver for interrupt-driven operation:

1. Write to the UART Baud Rate High and Low Byte registers to set the acceptable baud rate.
2. Enable the UART pin functions by configuring the associated GPIO port pins for alternate function operation.
3. Execute a `DI` instruction to disable interrupts.
4. Write to the Interrupt control registers to enable the UART Receiver interrupt and set the acceptable priority.
5. Clear the UART Receiver interrupt in the applicable Interrupt Request register.
6. Write to the UART Control 1 Register to enable Multiprocessor (9-bit) mode functions, if appropriate.
   - Set the Multiprocessor Mode Select (`MPEN`) to Enable MULTIPROCESSOR mode
   - Set the Multiprocessor Mode Bits, `MPMD[1:0]`, to select the acceptable address matching scheme
   - Configure the UART to interrupt on received data and errors or errors only (interrupt on errors only is unlikely to be useful for Z8 Encore! XP devices without a DMA block)
7. Write the device address to the Address Compare Register (automatic MULTIPROCESSOR modes only).
8. Write to the UART Control 0 register to:
   - Set the receive enable bit (`REN`) to enable the UART for data reception
   - Enable parity, if appropriate and if multiprocessor mode is not enabled, and select either even or odd parity
9. Execute an `EI` instruction to enable interrupts.

The UART is now configured for interrupt-driven data reception. When the UART Receiver interrupt is detected, the associated interrupt service routine (ISR) performs the following:

1. Checks the UART Status 0 register to determine the source of the interrupt - error, break, or received data.
2. Reads the data from the UART Receive Data register if the interrupt was because of data available. If operating in MULTIPROCESSOR (9-bit) mode, further actions may be required depending on the MULTIPROCESSOR mode bits MPMD[1:0].

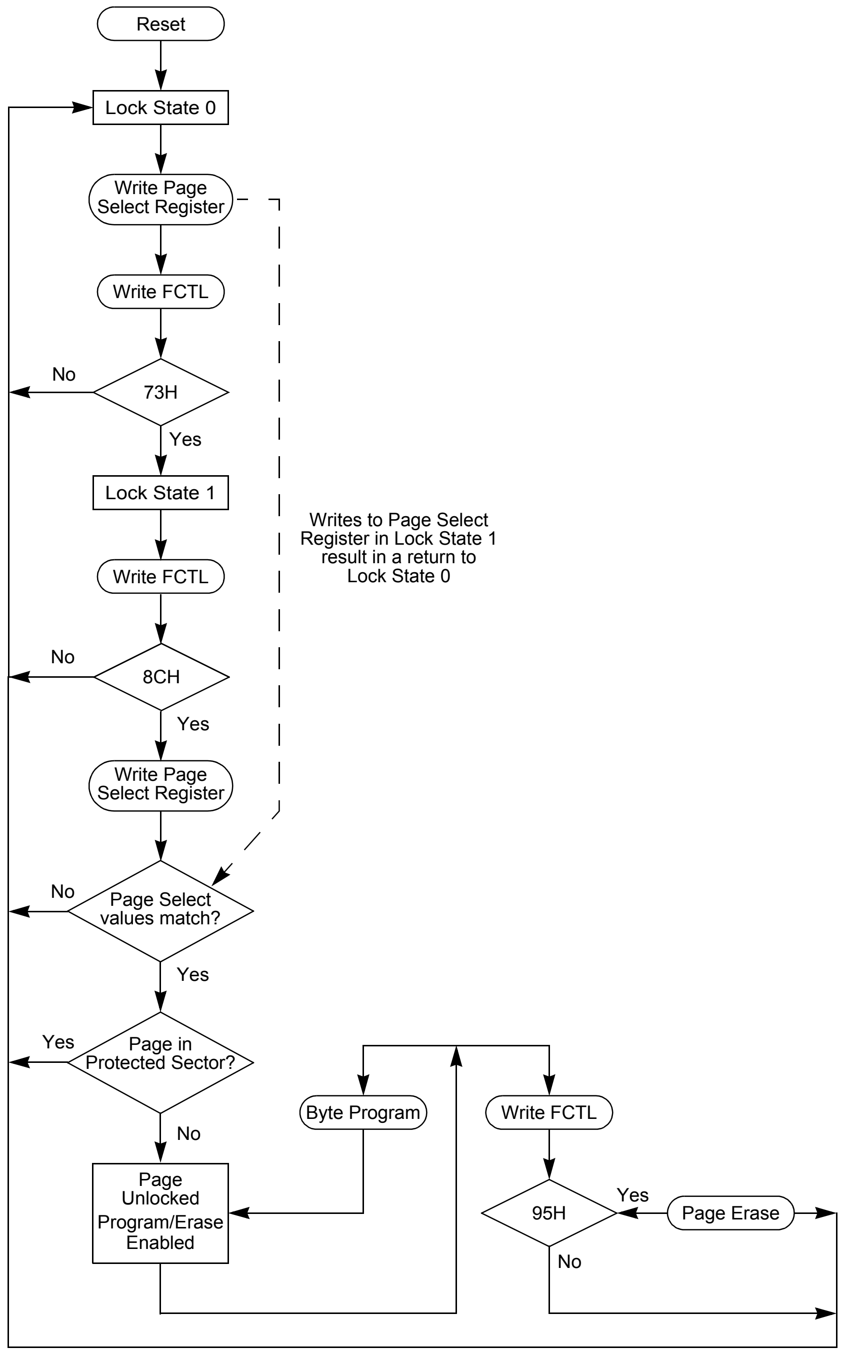


Figure 21. Flash Controller Operation Flowchart

a bit of the Sector Protect Register has been set, it cannot be cleared except by powering down the device.

Byte Programming

The Flash Memory is enabled for byte programming after unlocking the Flash Controller and successfully enabling either Mass Erase or Page Erase. When the Flash Controller is unlocked and Mass Erase is successfully completed, all Program Memory locations are available for byte programming. In contrast, when the Flash Controller is unlocked and Page Erase is successfully enabled, only the locations of the selected page are available for byte programming. An erased Flash byte contains all 1's (`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 commands.

Byte Programming is 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 *eZ8 CPU Core User Manual (UM0128)* available for download at www.zilog.com. While the Flash Controller programs the Flash memory, the eZ8 CPU idles but the system clock and on-chip peripherals continue to operate. To exit programming mode and lock the Flash, write any value to the Flash Control register, except the Mass Erase or Page Erase commands.

! **Caution:** *The byte at each address of the Flash memory cannot be programmed (any bits written to 0) more than twice before an erase cycle occurs. Doing so may result in corrupted data at the target byte.*

Page Erase

The Flash memory can be erased one page (512 bytes) at a time. Page Erasing the Flash memory sets all bytes in that page to the value `FFH`. The Flash Page Select register identifies the page to be erased. Only a page residing in an unprotected sector can be erased. With the Flash Controller unlocked and the active page set, writing the value `95h` to the Flash Control register initiates the Page Erase operation. While the Flash Controller executes the Page Erase operation, the eZ8 CPU idles but the system clock and on-chip peripherals continue to operate. The eZ8 CPU resumes operation after the Page Erase operation completes. If the Page Erase operation is performed using the On-Chip Debugger, poll the Flash Status register to determine when the Page Erase operation is complete. When the Page Erase is complete, the Flash Controller returns to its locked state.

Mass Erase

The Flash memory can also be Mass Erased using the Flash Controller, but only by using the On-Chip Debugger. Mass Erasing the Flash memory sets all bytes to the value `FFH`. With the Flash Controller unlocked and the Mass Erase successfully enabled, writing the point, the PA0/DBG pin can be used to autobaud and cause the device to enter DEBUG mode. For more details, see OCD Unlock Sequence (8-Pin Devices Only) on page 156.

Exiting DEBUG Mode

The device exits DEBUG mode following any of these operations:

- Clearing the DBGMODE bit in the OCD Control Register to 0
- Power-On Reset
- Voltage Brownout reset
- Watchdog Timer reset
- Asserting the $\overline{\text{RESET}}$ pin Low to initiate a Reset
- Driving the DBG pin Low while the device is in STOP mode initiates a system reset

OCD Data Format

The OCD interface uses the asynchronous data format defined for RS-232. Each character is transmitted as 1 Start bit, 8 data bits (least-significant bit first), and 1 Stop bit as displayed in Figure 25.

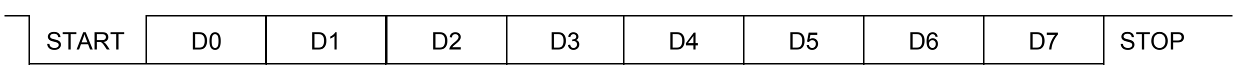


Figure 25. OCD Data Format

Note: *When responding to a request for data, the OCD may commence transmitting immediately after receiving the stop bit of an incoming frame. Therefore, when sending the stop bit, the host must not actively drive the DBG pin High for more than 0.5 bit times. It is recommended that, if possible, the host drives the DBG pin using an open-drain output.*

OCD Auto-Baud Detector/Generator

To run over a range of baud rates (data bits per second) with various system clock frequencies, the OCD contains an auto-baud detector/generator. After a reset, the OCD is idle until it receives data. The OCD requires that the first character sent from the host is the character `80H`. The character `80H` has eight continuous bits Low (one Start bit plus 7 data bits), framed between High bits. The auto-baud detector measures this period and sets the OCD baud rate generator accordingly.

The auto-baud detector/generator is clocked by the system clock. The minimum baud rate is the system clock frequency divided by 512. For optimal operation with asynchronous is not in DEBUG mode or if the Flash Read Protect Option bit is enabled, the data is discarded.

```
DBG ← 0AH
DBG ← Program Memory Address[15:8]
DBG ← Program Memory Address[7:0]
DBG ← Size[15:8]
DBG ← Size[7:0]
DBG ← 1-65536 data bytes
```

- **Read Program Memory (0BH)**—The Read Program Memory command reads data from Program Memory. This command is equivalent to the `LDC` and `LDCI` instructions. Data can be read 1–65536 bytes at a time (65536 bytes can be read by setting size to 0). If the device is not in DEBUG mode or if the Flash Read Protect Option Bit is enabled, this command returns `FFH` for the data.

```
DBG ← 0BH
DBG ← Program Memory Address[15:8]
DBG ← Program Memory Address[7:0]
DBG ← Size[15:8]
DBG ← Size[7:0]
DBG → 1-65536 data bytes
```

- **Write Data Memory (0CH)**—The Write Data Memory command writes data to Data Memory. This command is equivalent to the `LDE` and `LDEI` instructions. Data can be written 1–65536 bytes at a time (65536 bytes can be written by setting size to 0). If the device is not in DEBUG mode or if the Flash Read Protect Option Bit is enabled, the data is discarded.

```
DBG ← 0CH
DBG ← Data Memory Address[15:8]
DBG ← Data Memory Address[7:0]
DBG ← Size[15:8]
DBG ← Size[7:0]
DBG ← 1-65536 data bytes
```

- **Read Data Memory (0DH)**—The Read Data Memory command reads from Data Memory. This command is equivalent to the `LDE` and `LDEI` instructions. Data can be read 1 to 65536 bytes at a time (65536 bytes can be read by setting size to 0). If the device is not in DEBUG mode, this command returns `FFH` for the data.

```
DBG ← 0DH
DBG ← Data Memory Address[15:8]
DBG ← Data Memory Address[7:0]
DBG ← Size[15:8]
DBG ← Size[7:0]
DBG → 1-65536 data bytes
```

- **Read Program Memory CRC (0EH)**—The Read Program Memory Cyclic Redundancy Check (CRC) command computes and returns the CRC of Program Memory using the 16-bit CRC-CCITT polynomial. If the device is not in DEBUG mode, this command returns `FFFFH` for the CRC value. Unlike most other OCD Read commands, there is a delay from issuing of the command until the OCD returns the data. The OCD reads the Program Memory, calculates the CRC value, and returns the result. The delay is a function of the Program Memory size and is approximately equal to the system clock period multiplied by the number of bytes in the Program Memory.

```
DBG ← 0EH
DBG → CRC[15:8]
DBG → CRC[7:0]
```

- **Step Instruction (10H)**—The Step Instruction command steps one assembly instruction at the current Program Counter (PC) location. If the device is not in DEBUG mode or the Flash Read Protect Option bit is enabled, the OCD ignores this command.

```
DBG ← 10H
```

- **Stuff Instruction (11H)**—The Stuff Instruction command steps one assembly instruction and allows specification of the first byte of the instruction. The remaining 0-4 bytes of the instruction are read from Program Memory. This command is useful for stepping over instructions where the first byte of the instruction has been overwritten by a Breakpoint. If the device is not in DEBUG mode or the Flash Read Protect Option bit is enabled, the OCD ignores this command.

```
DBG ← 11H
DBG ← opcode[7:0]
```

- **Execute Instruction (12H)**—The Execute Instruction command allows sending an entire instruction to be executed to the eZ8 CPU. This command can also step over breakpoints. The number of bytes to send for the instruction depends on the opcode. If the device is not in DEBUG mode or the Flash Read Protect Option bit is enabled, this command reads and discards one byte.

```
DBG ← 12H
DBG ← 1-5 byte opcode
```

On-Chip Debugger Control Register Definitions

OCD Control Register

The OCD Control register controls the state of the OCD. This register is used to enter or exit DEBUG mode and to enable the `BRK` instruction. It also resets Z8 Encore! XP® F0823 Series device.

Table 106 lists additional symbols that are used throughout the Instruction Summary and Instruction Set Description sections.

Table 106. Additional Symbols

Symbol	Definition
dst	Destination Operand
src	Source Operand
@	Indirect Address Prefix
SP	Stack Pointer
PC	Program Counter
FLAGS	Flags Register
RP	Register Pointer
#	Immediate Operand Prefix
B	Binary Number Suffix
%	Hexadecimal Number Prefix
H	Hexadecimal Number Suffix

Assignment of a value is indicated by an arrow. For example,

dst ← dst + src

indicates the source data is added to the destination data and the result is stored in the destination location.

eZ8 CPU Instruction Classes

eZ8 CPU instructions are divided functionally into the following groups:

- Arithmetic
- Bit Manipulation
- Block Transfer
- CPU Control
- Load
- Logical
- Program Control

Table 117. Absolute Maximum Ratings (Continued)

Parameter	Minimum	Maximum	Units	Notes
Maximum current into V_{DD} or out of V_{SS}		125	mA	

Operating temperature is specified in DC Characteristics.

1. This voltage applies to all pins except the following: V_{DD}, AV_{DD}, pins supporting analog input (Port B[5:0], Port C[2:0]) and pins supporting the crystal oscillator (PA0 and PA1). On the 8-pin packages, this applies to all pins but V_{DD}.
2. This voltage applies to pins on the 20/28 pin packages supporting analog input (Port B[5:0], Port C[2:0]) and pins supporting the crystal oscillator (PA0 and PA1).

DC Characteristics

Table 118 lists the DC characteristics of the Z8 Encore! XP® F0823 Series products. All voltages are referenced to V_{SS}, the primary system ground.

Table 118. DC Characteristics

		T_A = -40 °C to +105 °C (unless otherwise specified)				
Symbol	**Parameter**	**Minimum**	**Typical**	**Maximum**	**Units**	**Conditions**
V_{DD}	Supply Voltage	2.7	–	3.6	V	
V_{IL1}	Low Level Input Voltage	-0.3	–	0.3*V_{DD}	V	
V_{IH1}	High Level Input Voltage	0.7*V_{DD}	–	5.5	V	For all input pins without analog or oscillator function. For all signal pins on the 8-pin devices. Programmable pull-ups must also be disabled.
V_{IH2}	High Level Input Voltage	0.7*V_{DD}	–	V_{DD}+0.3	V	For those pins with analog or oscillator function (20-/28-pin devices only), or when programmable pull-ups are enabled.
V_{OL1}	Low Level Output Voltage	–	–	0.4	V	I_{OL} = 2 mA; V_{DD} = 3.0 V High Output Drive disabled.
V_{OH1}	High Level Output Voltage	2.4	–	–	V	I_{OH} = -2 mA; V_{DD} = 3.0 V High Output Drive disabled.
V_{OL2}	Low Level Output Voltage	–	–	0.6	V	I_{OL} = 20 mA; V_{DD} = 3.3 V High Output Drive enabled.

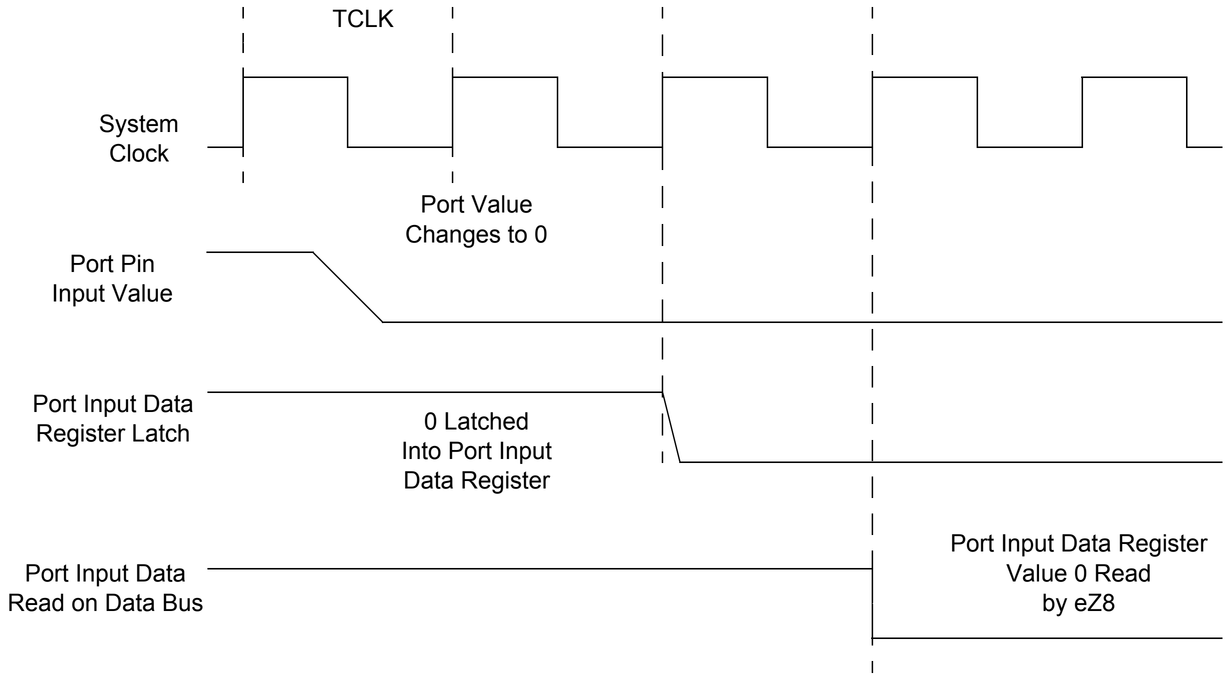


Figure 29. Port Input Sample Timing

Table 127. GPIO Port Input Timing

		Delay (ns)	
Parameter	**Abbreviation**	**Minimum**	**Maximum**
T_{S_PORT}	Port Input Transition to XIN Rise Setup Time (Not pictured)	5	–
T_{H_PORT}	XIN Rise to Port Input Transition Hold Time (Not pictured)	0	–
T_{SMR}	GPIO Port Pin Pulse Width to ensure Stop Mode Recovery (for GPIO Port Pins enabled as SMR sources)	1 µs	

Figure 33 and Table 131 provide timing information for UART pins for the case where CTS is not used for flow control. DE asserts after the transmit data register has been written. DE remains asserted for multiple characters as long as the transmit data register is written with the next character before the current character has completed.

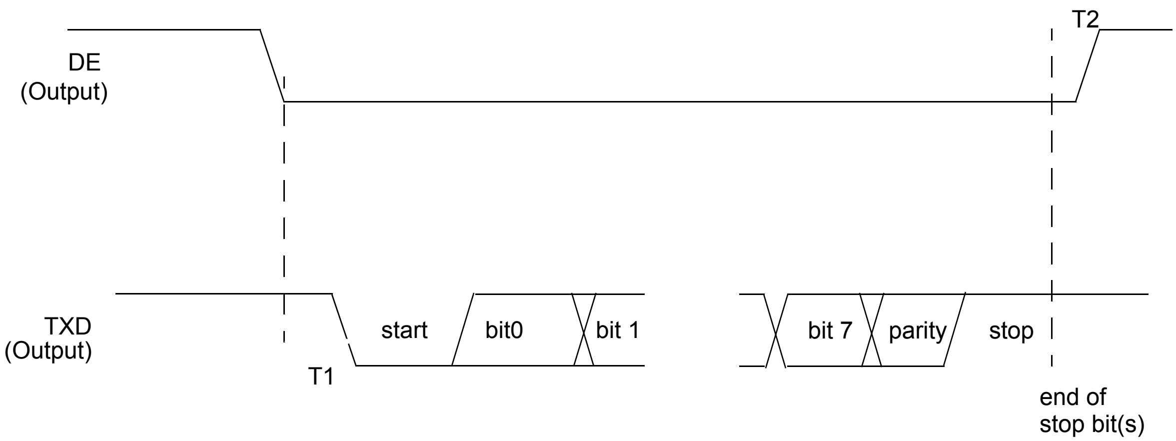
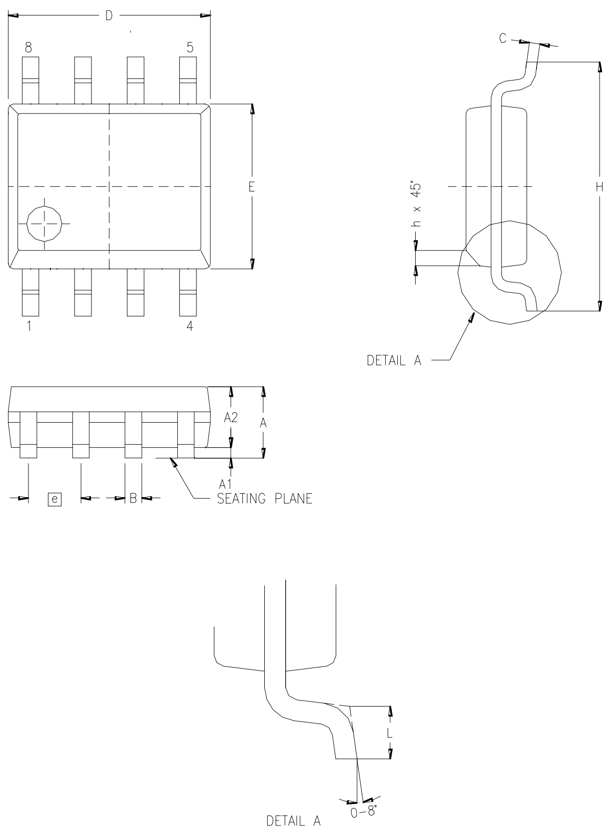


Figure 33. UART Timing Without CTS

Table 131. UART Timing Without CTS

Parameter	Abbreviation	Delay (ns)	
		Minimum	**Maximum**
UART			
T_1	DE assertion to TXD falling edge (start bit) delay	1 * XIN period	1 bit time
T_2	End of Stop Bit(s) to DE deassertion delay (Tx data register is empty)	± 5	

Figure 35 displays the 8-pin Small Outline Integrated Circuit package (SOIC) available for the Z8 Encore! XP F0823 Series devices.



SYMBOL	MILLIMETER		INCH	
	MIN	MAX	MIN	MAX
A	1.55	1.73	0.061	0.068
A1	0.10	0.25	0.004	0.010
A2	1.40	1.55	0.055	0.061
B	0.36	0.48	0.014	0.019
C	0.18	0.25	0.007	0.010
D	4.80	4.98	0.189	0.196
E	3.81	3.99	0.150	0.157
e	1.27 BSC		.050 BSC	
H	5.84	6.15	0.230	0.242
h	0.25	0.40	0.010	0.016
L	0.46	0.81	0.018	0.032

CONTROLLING DIMENSIONS : MM
LEADS ARE COPLANAR WITHIN .004 INCH.

Figure 35. 8-Pin Small Outline Integrated Circuit Package (SOIC)