



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	Discontinued at Digi-Key
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	16
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	20-SSOP (0.209", 5.30mm Width)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f0813hh005sc

 **Warning:** DO NOT USE IN LIFE SUPPORT

LIFE SUPPORT POLICY

ZILOG'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS PRIOR WRITTEN APPROVAL OF THE PRESIDENT AND GENERAL COUNSEL OF ZILOG CORPORATION.

As used herein

Life support devices or systems are devices which (a) are intended for surgical implant into the body, or (b) support or sustain life and whose failure to perform when properly used in accordance with instructions for use provided in the labeling can be reasonably expected to result in a significant injury to the user. A critical component is any component in a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system or to affect its safety or effectiveness.

Document Disclaimer

©2008 by Zilog, Inc. All rights reserved. Information in this publication concerning the devices, applications, or technology described is intended to suggest possible uses and may be superseded. ZILOG, INC. DOES NOT ASSUME LIABILITY FOR OR PROVIDE A REPRESENTATION OF ACCURACY OF THE INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED IN THIS DOCUMENT. ZILOG ALSO DOES NOT ASSUME LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT RELATED IN ANY MANNER TO USE OF INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED HEREIN OR OTHERWISE. The information contained within this document has been verified according to the general principles of electrical and mechanical engineering.

Z8, Z8 Encore!, Z8 Encore! XP, Z8 Encore! MC, Crimzon, eZ80, and ZNEO are trademarks or registered trademarks of Zilog, Inc. All other product or service names are the property of their respective owners.

Internal Precision Oscillator

The internal precision oscillator (IPO) is a trimmable clock source that requires no external components.

10-Bit Analog-to-Digital Converter

The optional analog-to-digital converter (ADC) converts an analog input signal to a 10-bit binary number. The ADC accepts inputs from eight different analog input pins in both single-ended and differential modes.

Analog Comparator

The analog comparator compares the signal at an input pin with either an internal programmable voltage reference or a second input pin. The comparator output can be used to drive either an output pin or to generate an interrupt.

Universal Asynchronous Receiver/Transmitter

The UART is full-duplex and capable of handling asynchronous data transfers. The UART supports 8- and 9-bit data modes and selectable parity. The UART also supports multi-drop address processing in hardware. The UART baud rate generator can be configured and used as a basic 16-bit timer.

Timers

Two enhanced 16-bit reloadable timers can be used for timing/counting events or for motor control operations. These timers provide a 16-bit programmable reload counter and operate in ONE-SHOT, CONTINUOUS, GATED, CAPTURE, CAPTURE RESTART, COMPARE, CAPTURE AND COMPARE, PWM SINGLE OUTPUT, and PWM DUAL OUTPUT modes.

Interrupt Controller

Z8 Encore! XP[®] F0823 Series products support up to 20 interrupts. These interrupts consist of eight internal peripheral interrupts and 12 general-purpose I/O pin interrupt sources. The interrupts have three levels of programmable interrupt priority.

Table 3. Signal Descriptions (Continued)

Signal Mnemonic	I/O	Description
Analog		
ANA[7:0]	I	Analog port. These signals are used as inputs to the ADC. The ANA0, ANA1, and ANA2 pins can also access the inputs and output of the integrated transimpedance amplifier.
VREF	I/O	Analog-to-Digital Converter reference voltage input.
Clock Input		
CLKIN	I	Clock Input Signal. This pin can be used to input a TTL-level signal to be used as the system clock.
LED Drivers		
LED	O	Direct LED drive capability. All port C pins have the capability to drive an LED without any other external components. These pins have programmable drive strengths set by the GPIO block.
On-Chip Debugger		
DBG	I/O	Debug. This signal is the control and data input and output to and from the OCD.
<p>! Caution: <i>The DBG pin is open-drain and requires an external pull-up resistor to ensure proper operation.</i></p>		
Reset		
RESET	I/O	RESET. Generates a reset when asserted (driven Low). Also serves as a reset indicator; the Z8 Encore! XP forces this pin Low when in reset. This pin is open-drain and features an enabled internal pull-up resistor.
Power Supply		
V _{DD}	I	Digital Power Supply.
AV _{DD}	I	Analog Power Supply.
V _{SS}	I	Digital Ground.
AV _{SS}	I	Analog Ground.
<p>Note: The AV_{DD} and AV_{SS} signals are available only in 28-pin packages with ADC. They are replaced by PB6 and PB7 on 28-pin packages without ADC.</p>		

Pin Characteristics

Table 4 provides detailed information about the characteristics for each pin available on Z8 Encore! XP F0823 Series 20- and 28-pin devices. Data in Table 4 is sorted alphabetically by the pin symbol mnemonic.

General-Purpose Input/Output

Z8 Encore! XP[®] F0823 Series products support a maximum of 24 port pins (Ports A–C) for general-purpose input/output (GPIO) operations. Each port contains control and data registers. The GPIO control registers determine data direction, open-drain, output drive current, programmable pull-ups, Stop Mode Recovery functionality, and alternate pin functions. Each port pin is individually programmable. In addition, the Port C pins are capable of direct LED drive at programmable drive strengths.

GPIO Port Availability By Device

Table 14 lists the port pins available with each device and package type.

Table 14. Port Availability by Device and Package Type

Devices	Package	10-Bit ADC	Port A	Port B	Port C	Total I/O
Z8F0823SB, Z8F0823PB Z8F0423SB, Z8F0423PB Z8F0223SB, Z8F0223PB Z8F0123SB, Z8F0123PB	8-pin	Yes	[5:0]	No	No	6
Z8F0813SB, Z8F0813PB Z8F0413SB, Z8F0413PB Z8F0213SB, Z8F0213PB Z8F0113SB, Z8F011vPB	8-pin	No	[5:0]	No	No	6
Z8F0823PH, Z8F0823HH Z8F0423PH, Z8F0423HH Z8F0223PH, Z8F0223HH Z8F0123PH, Z8F0123HH	20-pin	Yes	[7:0]	[3:0]	[3:0]	16
Z8F0813PH, Z8F0813HH Z8F0413PH, Z8F0413HH Z8F0213PH, Z8F0213HH Z8F0113PH, Z8F0113HH	20-pin	No	[7:0]	[3:0]	[3:0]	16
Z8F0823PJ, Z8F0823SJ Z8F0423PJ, Z8F0423SJ Z8F0223PJ, Z8F0223SJ Z8F0123PJ, Z8F0123SJ	28-pin	Yes	[7:0]	[5:0]	[7:0]	22
Z8F0813PJ, Z8F0813SJ Z8F0413PJ, Z8F0413SJ Z8F0213PJ, Z8F0213SJ Z8F0113PJ, Z8F0113SJ	28-pin	No	[7:0]	[7:0]	[7:0]	24

Table 36. Interrupt Request 2 Register (IRQ2)

BITS	7	6	5	4	3	2	1	0
FIELD	Reserved				PC3I	PC2I	PC1I	PC0I
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	FC6H							

Reserved—Must be 0

PCxI—Port C Pin *x* Interrupt Request

0 = No interrupt request is pending for GPIO Port C pin *x*

1 = An interrupt request from GPIO Port C pin *x* is awaiting service

where *x* indicates the specific GPIO Port C pin number (0–3)

IRQ0 Enable High and Low Bit Registers

Table 37 describes the priority control for IRQ0. The IRQ0 Enable High and Low Bit registers (Table 38 and Table 39) form a priority encoded enabling for interrupts in the Interrupt Request 0 register. Priority is generated by setting bits in each register.

Table 37. IRQ0 Enable and Priority Encoding

IRQ0ENH[x]	IRQ0ENL[x]	Priority	Description
0	0	Disabled	Disabled
0	1	Level 1	Low
1	0	Level 2	Nominal
1	1	Level 3	High

where *x* indicates the register bits from 0–7.

Table 38. IRQ0 Enable High Bit Register (IRQ0ENH)

BITS	7	6	5	4	3	2	1	0
FIELD	Reserved	T1ENH	T0ENH	U0RENH	U0TENH	Reserved	Reserved	ADCENH
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	FC1H							

0x = Timer Interrupt occurs on all defined Reload, Compare and Input Events
 10 = Timer Interrupt only on defined Input Capture/Deassertion Events
 11 = Timer Interrupt only on defined Reload/Compare Events

Reserved—Must be 0

PWMD—PWM Delay value

This field is a programmable delay to control the number of system clock cycles delay before the Timer Output and the Timer Output Complement are forced to their active state.

000 = No delay
 001 = 2 cycles delay
 010 = 4 cycles delay
 011 = 8 cycles delay
 100 = 16 cycles delay
 101 = 32 cycles delay
 110 = 64 cycles delay
 111 = 128 cycles delay

INPCAP—Input Capture Event

This bit indicates if the most recent timer interrupt is caused by a Timer Input Capture Event.

0 = Previous timer interrupt is not a result of Timer Input Capture Event
 1 = Previous timer interrupt is a result of Timer Input Capture Event

Timer 0–1 Control Register 1

The Timer 0–1 Control (TxCTL1) registers enable/disable the timers, set the prescaler value, and determine the timer operating mode.

Table 56. Timer 0–1 Control Register 1 (TxCTL1)

BITS	7	6	5	4	3	2	1	0
FIELD	TEN	TPOL	PRES			TMODE		
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	F07H, F0FH							

TEN—Timer Enable

0 = Timer is disabled

1 = Timer enabled to count

TPOL—Timer Input/Output Polarity

Operation of this bit is a function of the current operating mode of the timer

WDT Reset in NORMAL Operation

If configured to generate a Reset when a time-out occurs, the Watchdog Timer forces the device into the System Reset state. The WDT status bit in the Watchdog Timer Control register is set to 1. For more information on System Reset, see Reset and Stop Mode Recovery on page 21.

WDT Reset in STOP Mode

If configured to generate a Reset when a time-out occurs and the device is in STOP mode, the Watchdog Timer initiates a Stop Mode Recovery. Both the WDT status bit and the STOP bit in the Watchdog Timer Control register are set to 1 following WDT time-out in STOP mode. For more information, see Reset and Stop Mode Recovery on page 21.

Watchdog Timer Reload Unlock Sequence

Writing the unlock sequence to the Watchdog Timer Control Register (WDTCTL) address unlocks the three Watchdog Timer Reload Byte Registers (WDTU, WDTL, and WDTL) to allow changes to the time-out period. These write operations to the WDTCTL register address produce no effect on the bits in the WDTCTL register. The locking mechanism prevents spurious writes to the Reload registers. The following sequence is required to unlock the Watchdog Timer Reload Byte Registers (WDTU, WDTL, and WDTL) for write access.

1. Write 55H to the Watchdog Timer Control register (WDTCTL).
2. Write AAH to the Watchdog Timer Control register (WDTCTL).
3. Write the Watchdog Timer Reload Upper Byte register (WDTU).
4. Write the Watchdog Timer Reload High Byte register (WDTL).
5. Write the Watchdog Timer Reload Low Byte register (WDTL).

All three Watchdog Timer Reload registers must be written in the order just listed. There must be no other register writes between each of these operations. If a register write occurs, the lock state machine resets and no further writes can occur unless the sequence is restarted. The value in the Watchdog Timer Reload registers is loaded into the counter when the Watchdog Timer is first enabled and every time a WDT instruction is executed.

Watchdog Timer Control Register Definitions

Watchdog Timer Control Register

The Watchdog Timer Control (WDTCTL) register is a write-only control register. Writing the 55H, AAH unlock sequence to the WDTCTL register address unlocks the three

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]`.

received and the UART Receive Data register has not been read. If the RDA bit is reset to 0, reading the UART Receive Data register clears this bit.

0 = No overrun error occurred

1 = An overrun error occurred

FE—Framing Error

This bit indicates that a framing error (no Stop bit following data reception) was detected. Reading the UART Receive Data register clears this bit.

0 = No framing error occurred

1 = A framing error occurred

BRKD—Break Detect

This bit indicates that a break occurred. If the data bits, parity/multiprocessor bit, and Stop bit(s) are all 0s this bit is set to 1. Reading the UART Receive Data register clears this bit.

0 = No break occurred

1 = A break occurred

TDRE—Transmitter Data Register Empty

This bit indicates that the UART Transmit Data register is empty and ready for additional data. Writing to the UART Transmit Data register resets this bit.

0 = Do not write to the UART Transmit Data register

1 = The UART Transmit Data register is ready to receive an additional byte to be transmitted

TXE—Transmitter Empty

This bit indicates that the transmit shift register is empty and character transmission is finished.

0 = Data is currently transmitting

1 = Transmission is complete

CTS— $\overline{\text{CTS}}$ signal

When this bit is read it returns the level of the $\overline{\text{CTS}}$ signal. This signal is active Low.

UART Status 1 Register

This register contains multiprocessor control and status bits.

Table 65. UART Status 1 Register (U0STAT1)

BITS	7	6	5	4	3	2	1	0
FIELD	Reserved						NEWFRM	MPRX
RESET	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R/W	R/W	R	R
ADDR	F44H							

Table 78. Flash Code Protection Using the Flash Option Bits

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.

Flash Code Protection Using the Flash Controller

At Reset, the Flash Controller locks to prevent accidental program or erasure of the Flash memory. To program or erase the Flash memory, first write the Page Select Register with the target page. Unlock the Flash Controller by making two consecutive writes to the Flash Control register with the values 73H and 8CH, sequentially. The Page Select Register must be rewritten 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 more details, see Figure 21.

After unlocking a specific page, you can enable either Page Program or Erase. Writing the value 95H causes a Page Erase only if the active page resides in a sector that is not protected. Any other value written to the Flash Control register locks the Flash Controller. Mass Erase is not allowed in the user code but only in through the Debug Port.

After unlocking a specific page, you can also write to any byte on that page. After a byte is written, the page remains unlocked, allowing for subsequent writes to other bytes on the same page. Further writes to the Flash Control Register cause the active page to revert to a locked state.

Sector Based Flash Protection

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

The Sector Protect Register controls the protection state of each Flash sector. This register is shared with the Page Select Register. It is accessed by writing 73H followed by 5EH to the Flash controller. The next write to the Flash Control Register targets the Sector Protect Register.

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 by the CPU. External Flash programming through the OCD or via the Flash Controller Bypass mode are unaffected. After

Oscillator Control

Z8 Encore! XP[®] F0823 Series devices uses three possible clocking schemes, each user-selectable:

- On-chip precision trimmed RC oscillator
- External clock drive
- On-chip low power Watchdog Timer oscillator

In addition, Z8 Encore! XP F0823 Series devices contain clock failure detection and recovery circuitry, allowing continued operation despite a failure of the primary oscillator.

Operation

This chapter discusses the logic used to select the system clock and handle primary oscillator failures. A description of the specific operation of each oscillator is outlined elsewhere in this document.

System Clock Selection

The oscillator control block selects from the available clocks. Table 101 details each clock source and its usage.

Table 101. Oscillator Configuration and Selection

Clock Source	Characteristics	Required Setup
Internal Precision RC Oscillator	<ul style="list-style-type: none">• 32.8 kHz or 5.53 MHz• $\pm 4\%$ accuracy when trimmed• No external components required	<ul style="list-style-type: none">• Unlock and write Oscillator Control Register (OSCCTL) to enable and select oscillator at either 5.53 MHz or 32.8 kHz
External Clock Drive	<ul style="list-style-type: none">• 0 to 20 MHz• Accuracy dependent on external clock source	<ul style="list-style-type: none">• Write GPIO registers to configure PB3 pin for external clock function• Unlock and write OSCCTL to select external system clock• Apply external clock signal to GPIO
Internal Watchdog Timer Oscillator	<ul style="list-style-type: none">• 10 kHz nominal• $\pm 40\%$ accuracy; no external components required• Very Low power consumption	<ul style="list-style-type: none">• Enable WDT if not enabled and wait until WDT Oscillator is operating.• Unlock and write Oscillator Control Register (OSCCTL) to enable and select oscillator

Internal Precision Oscillator

The internal precision oscillator (IPO) is designed for use without external components. You can either manually trim the oscillator for a non-standard frequency or use the automatic factory-trimmed version to achieve a 5.53 MHz frequency. The features of IPO include:

- On-chip RC oscillator that does not require external components
- Output frequency of either 5.53 MHz or 32.8 kHz (contains both a fast and a slow mode)
- Trimming possible through Flash option bits with user override
- Elimination of crystals or ceramic resonators in applications where high timing accuracy is not required

Operation

An 8-bit trimming register, incorporated into the design, compensates for absolute variation of oscillator frequency. Once trimmed the oscillator frequency is stable and does not require subsequent calibration. Trimming is performed during manufacturing and is not necessary for you to repeat unless a frequency other than 5.53 MHz (fast mode) or 32.8 kHz (slow mode) is required. This trimming is done at +30 °C and a supply voltage of 3.3 V, so accuracy of this operating point is optimal.

Power down this block for minimum system power. By default, the oscillator is configured through the Flash Option bits. However, the user code can override these trim values as described in Trim Bit Address Space on page 146.

Select one of the two frequencies for the oscillator: 5.53 MHz and 32.8 kHz, using the OSCSEL bits in the Oscillator Control on page 165.

eZ8 CPU Instruction Set

Assembly Language Programming Introduction

The eZ8 CPU assembly language provides a means for writing an application program without concern for actual memory addresses or machine instruction formats. A program written in assembly language is called a source program. Assembly language allows the use of symbolic addresses to identify memory locations. It also allows mnemonic codes (opcodes and operands) to represent the instructions themselves. The opcodes identify the instruction while the operands represent memory locations, registers, or immediate data values.

Each assembly language program consists of a series of symbolic commands called statements. Each statement can contain labels, operations, operands, and comments.

Labels are assigned to a particular instruction step in a source program. The label identifies that step in the program as an entry point for use by other instructions.

The assembly language also includes assembler directives that supplement the machine instruction. The assembler directives, or pseudo-ops, are not translated into a machine instruction. Rather, the pseudo-ops are interpreted as directives that control or assist the assembly process.

The source program is processed (assembled) by the assembler to obtain a machine language program called the object code. The object code is executed by the eZ8 CPU. An example segment of an assembly language program is detailed in the following example.

Assembly Language Source Program Example

```
JP  START          ; Everything after the semicolon is a comment.

START:             ; A label called 'START'. The first instruction (JP  START) in this
                   ; example causes program execution to jump to the point within the
                   ; program where the START label occurs.

LD  R4, R7         ; A Load (LD) instruction with two operands. The first operand,
                   ; Working Register R4, is the destination. The second operand,
                   ; Working Register R7, is the source. The contents of R7 is
                   ; written into R4.

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.
```


Table 115. eZ8 CPU Instruction Summary (Continued)

Assembly Mnemonic	Symbolic Operation	Address Mode		Opcode(s) (Hex)	Flags						Fetch Cycles	Instr. Cycles
		dst	src		C	Z	S	V	D	H		
COM dst	dst ← ~dst	R		60	–	*	*	0	–	–	2	2
		IR		61							2	3
CP dst, src	dst - src	r	r	A2	*	*	*	*	–	–	2	3
		r	lr	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	lr	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
DJNZ dst, RA	dst ← dst - 1 if dst ≠ 0 PC ← PC + X	r		0A-FA	–	–	–	–	–	–	2	3
EI	IRQCTL[7] ← 1			9F	–	–	–	–	–	–	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							

Table 115. eZ8 CPU Instruction Summary (Continued)

Assembly Mnemonic	Symbolic Operation	Address Mode		Opcode(s) (Hex)	Flags						Fetch Cycles	Instr. Cycles
		dst	src		C	Z	S	V	D	H		
HALT	HALT Mode			7F	–	–	–	–	–	–	1	2
INC dst	dst ← dst + 1	R		20	–	*	*	–	–	–	2	2
		IR		21							2	3
		r		0E-FE							1	2
INCW dst	dst ← dst + 1	RR		A0	–	*	*	*	–	–	2	5
		IRR		A1							2	6
IRET	FLAGS ← @SP SP ← SP + 1 PC ← @SP SP ← SP + 2 IRQCTL[7] ← 1			BF	*	*	*	*	*	*	1	5
JP dst	PC ← dst	DA		8D	–	–	–	–	–	–	3	2
		IRR		C4							2	3
JP cc, dst	if cc is true PC ← dst	DA		0D-FD	–	–	–	–	–	–	3	2
JR dst	PC ← PC + X	DA		8B	–	–	–	–	–	–	2	2
JR cc, dst	if cc is true PC ← PC + X	DA		0B-FB	–	–	–	–	–	–	2	2
LD dst, rc	dst ← src	r	IM	0C-FC	–	–	–	–	–	–	2	2
		r	X(r)	C7							3	3
		X(r)	r	D7							3	4
		r	lr	E3							2	3
		R	R	E4							3	2
		R	IR	E5							3	4
		R	IM	E6							3	2
		IR	IM	E7							3	3
		lr	r	F3							2	3
		IR	R	F5							3	3
Flags Notation:		* = Value is a function of the result of the operation. – = Unaffected X = Undefined				0 = Reset to 0 1 = Set to 1						

UART Timing

Figure 32 and Table 130 provide timing information for UART pins for the case where CTS is used for flow control. The CTS to DE assertion delay (T1) assumes the transmit data register has been loaded with data prior to CTS assertion.

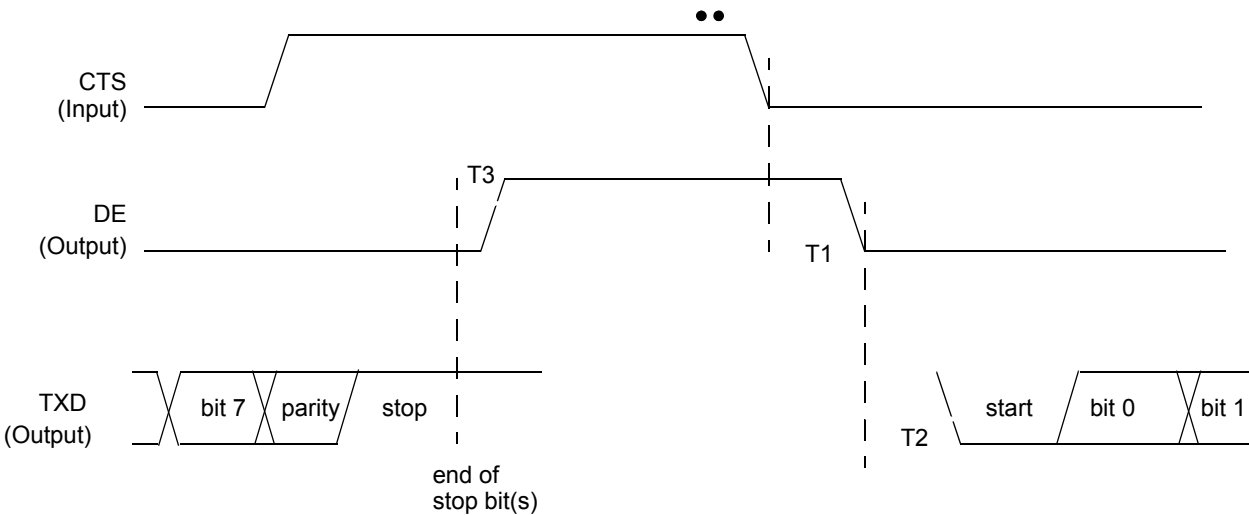


Figure 32. UART Timing With CTS

Table 130. UART Timing With CTS

Parameter	Abbreviation	Delay (ns)	
		Minimum	Maximum
UART			
T ₁	CTS Fall to DE output delay	2 * XIN period	2 * XIN period + 1 bit time
T ₂	DE assertion to TXD falling edge (start bit) delay	± 5	
T ₃	End of Stop Bit(s) to DE deassertion delay	± 5	

Part Number	Flash	RAM	I/O Lines	Interrupts	16-Bit Timers w/PWM	10-Bit A/D Channels	UART with IrDA	Description
Z8 Encore! XP with 4 KB Flash, 10-Bit Analog-to-Digital Converter								
Standard Temperature: 0 °C to 70 °C								
Z8F0423PB005SC	4 KB	1 KB	6	12	2	4	1	PDIP 8-pin package
Z8F0423QB005SC	4 KB	1 KB	6	12	2	4	1	QFN 8-pin package
Z8F0423SB005SC	4 KB	1 KB	6	12	2	4	1	SOIC 8-pin package
Z8F0423SH005SC	4 KB	1 KB	16	18	2	7	1	SOIC 20-pin package
Z8F0423HH005SC	4 KB	1 KB	16	18	2	7	1	SSOP 20-pin package
Z8F0423PH005SC	4 KB	1 KB	16	18	2	7	1	PDIP 20-pin package
Z8F0423SJ005SC	4 KB	1 KB	22	18	2	8	1	SOIC 28-pin package
Z8F0423HJ005SC	4 KB	1 KB	22	18	2	8	1	SSOP 28-pin package
Z8F0423PJ005SC	4 KB	1 KB	22	18	2	8	1	PDIP 28-pin package
Extended Temperature: -40 °C to 105 °C								
Z8F0423PB005EC	4 KB	1 KB	6	12	2	4	1	PDIP 8-pin package
Z8F0423QB005EC	4 KB	1 KB	6	12	2	4	1	QFN 8-pin package
Z8F0423SB005EC	4 KB	1 KB	6	12	2	4	1	SOIC 8-pin package
Z8F0423SH005EC	4 KB	1 KB	16	18	2	7	1	SOIC 20-pin package
Z8F0423HH005EC	4 KB	1 KB	16	18	2	7	1	SSOP 20-pin package
Z8F0423PH005EC	4 KB	1 KB	16	18	2	7	1	PDIP 20-pin package
Z8F0423SJ005EC	4 KB	1 KB	22	18	2	8	1	SOIC 28-pin package
Z8F0423HJ005EC	4 KB	1 KB	22	18	2	8	1	SSOP 28-pin package
Z8F0423PJ005EC	4 KB	1 KB	22	18	2	8	1	PDIP 28-pin package
Replace C with G for Lead-Free Packaging								