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, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	11
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/z8f0811hh020sg

CPU and Peripheral Overview

Zilog's latest eZ8 8-bit CPU meets the continuing demand for faster and more code-efficient microcontrollers. The eZ8 CPU executes a superset of the original Z8® instruction set.

The eZ8 CPU features:

- 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 4KB
- 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 about the eZ8 CPU, refer to the eZ8 CPU Core User Manual (UM0128), which is available for download at www.zilog.com.

General Purpose Input/Output

Z8 Encore! XP® F0822 Series features 11 to 19 port pins (Ports A–C) for General-Purpose Input/Output (GPIO). The number of available GPIO pins is a function of package type. Each pin is individually programmable. Ports A and C support 5V-tolerant inputs.

Flash Controller

The Flash Controller programs and erases the contents of Flash memory.

of Reset, all GPIO pins are configured as inputs. All GPIO programmable pull-ups are disabled.

During Reset, the eZ8 CPU and the on-chip peripherals are idle; however, the on-chip crystal oscillator and WDT oscillator continue to run. The system clock begins operating following the WDT oscillator cycle count. The eZ8 CPU and on-chip peripherals remain idle through all of the 16 cycles of the system clock.

Upon Reset, control registers within the Register File which have a defined Reset value are loaded with their reset values. Other control registers (including the Stack Pointer, Register Pointer, and Flags) and general-purpose RAM are undefined following the Reset. The eZ8 CPU fetches the Reset vector at Program memory addresses `0002H` and `0003H` and loads that value into the Program Counter. Program execution begins at the Reset vector address.

Reset Sources

Table 9 lists the reset sources as a function of the operating mode. The remainder of this section provides more detail about the individual reset sources.

Note: A POR/VBO event always has priority over all other possible reset sources to ensure a full system reset occurs.

Table 9. Reset Sources and Resulting Reset Type

Operating Mode	Reset Source	Reset Type
NORMAL or HALT modes	POR/VBO	System Reset
	WDT time-out when configured for Reset	System Reset
	$\overline{\text{RESET}}$ pin assertion	System Reset
	OCD-initiated Reset (OCDCTL[0] set to 1)	System Reset except the OCD is unaffected by the reset
STOP Mode	POR/ VBO	System Reset
	$\overline{\text{RESET}}$ pin assertion	System Reset
	DBG pin driven Low	System Reset

Power-On Reset

Each device in the Z8 Encore! XP® F0822 Series contains an internal POR circuit. The POR circuit monitors the supply voltage and holds the device in the Reset state until the supply voltage reaches a safe operating level. After the supply voltage exceeds the POR

- System clock is enabled and continues to operate
- eZ8 CPU is stopped
- Program counter stops incrementing
- WDT’s internal RC oscillator continues to operate
- If enabled, the WDT continues to operate
- All other on-chip peripherals continue to operate

The eZ8 CPU can be brought out of HALT Mode by any of the following operations:

- Interrupt
- WDT time-out (interrupt or reset)
- Power-On Reset
- Voltage Brown-Out reset
- External $\overline{\text{RESET}}$ pin assertion

To minimize current in HALT Mode, all GPIO pins which are configured as inputs must be driven to one of the supply rails (V_{CC} or GND).

Port A–C Pull-up Enable Subregisters

The Port A–C Pull-Up Enable Subregister, shown in Table 21, is accessed through the Port A–C Control Register by writing `06H` to the Port A–C Address Register. Setting the bits in the Port A–C Pull-Up Enable subregisters enables a weak internal resistive pull-up on the specified Port pins.

Table 21. Port A–C Pull-Up Enable Subregisters

Bit	7	6	5	4	3	2	1	0
Field	PPUE7	PPUE6	PPUE5	PPUE4	PPUE3	PPUE2	PPUE1	PPUE0
RESET	0							
R/W	R/W							
Address	See footnote.							
Note: If 06H is written to the Port A–C Address Register, then it is accessible through the Port A–C Control Register.								

Bit	Description
[7:0] PPUEx	**Port Pull-up Enabled** 0 = The weak pull-up on the port pin is disabled. 1 = The weak pull-up on the port pin is enabled.

Note: x indicates register bits in the range [7:0].

Port A–C Input Data Registers

Reading from the Port A–C Input Data registers, shown in Table 22, returns the sampled values from the corresponding port pins. The Port A–C Input Data registers are read-only.

Table 22. Port A–C Input Data Registers (PxIN)

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

Bit	Description
[7:0] PxIN	**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).

Note: x indicates register bits in the range [7:0].

Table 32. IRQ1 Enable High Bit Register (IRQ1ENH)

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	PA7ENH	PA6ENH	PA5ENH	PA4ENH	PA3ENH	PA2ENH	PA1ENH	PA0ENH
RESET	0							
R/W	R/W							
Address	FC4H							

Bit	**Description**
[7:0] PAxENH	**Port A Bit[*x*] Interrupt Request Enable High Bit**
Note: *x* indicates register bits in the range [7:0].	

Table 33. IRQ1 Enable Low Bit Register (IRQ1ENL)

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	PA7ENL	PA6ENL	PA5ENL	PA4ENL	PA3ENL	PA2ENL	PA1ENL	PA0ENL
RESET	0							
R/W	R/W							
Address	FC5H							

Bit	**Description**
[7:0] PAxENL	**Port A Bit[*x*] Interrupt Request Enable Low Bit**
Note: *x* indicates register bits in the range [7:0].	

Table 39. Timer 0–1 High Byte Register (TxH)

Bit	7	6	5	4	3	2	1	0
Field	TH							
RESET	0							
R/W	R/W							
Address	F00H, F08H							

Table 40. Timer 0–1 Low Byte Register (TxL)

Bit	7	6	5	4	3	2	1	0
Field	TL							
RESET	0							1
R/W	R/W							
Address	F01H, F09H							

Bit	Description
[7:0] TH, TL	**Timer High and Low Bytes** These 2 bytes, {TMRH[7:0], TMRL[7:0]}, contain the current 16-bit timer count value.

Timer Reload High and Low Byte Registers

The Timer 0–1 Reload High and Low Byte (TxRH and TxRL) registers, shown in Tables 41 and 42, store a 16-bit reload value, {TRH[7:0], TRL[7:0]}. Values written to the Timer Reload High Byte Register are stored in a temporary holding register. When a write to the Timer Reload Low Byte Register occurs, the temporary holding register value is written to the Timer High Byte Register. This operation allows simultaneous updates of the 16-bit Timer reload value.

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

Table 41. Timer 0–1 Reload High Byte Register (TxRH)

Bit	7	6	5	4	3	2	1	0
Field	TRH							
RESET	1							
R/W	R/W							
Address	F02H, F0AH							

UART Interrupts

The UART features separate interrupts for the transmitter and the receiver. In addition, when the UART primary functionality is disabled, the BRG also functions as a basic timer with interrupt capability.

Transmitter Interrupts

The transmitter generates a single interrupt when the Transmit Data Register Empty bit (TDRE) is set to 1. This indicates that the transmitter is ready to accept new data for transmission. The TDRE interrupt occurs after the Transmit Shift Register has shifted the first bit of data out. At this point, the Transmit Data Register can be written with the next character to send. This provides 7 bit periods of latency to load the Transmit Data Register before the Transmit Shift Register completes shifting the current character. Writing to the UART Transmit Data Register clears the TDRE bit to 0.

Receiver Interrupts

The receiver generates an interrupt when any of the following occurs:

- A data byte is received and is available in the UART Receive Data Register. This interrupt can be disabled independent of the other receiver interrupt sources. The received data interrupt occurs after the receive character is received and placed in the Receive Data Register. Software must respond to this received data available condition before the next character is completely received to avoid an overrun error. In MULTIPROCESSOR Mode (MPEN = 1), the receive data interrupts are dependent on the multiprocessor configuration and the most recent address byte.
- A break is received.
- An overrun is detected.
- A data framing error is detected.

UART Overrun Errors

When an overrun error condition occurs the UART prevents overwriting of the valid data currently in the Receive Data Register. The break detect and overrun status bits are not displayed until the valid data is read.

After the valid data has been read, the UART Status 0 Register is updated to indicate the overrun condition (and Break Detect, if applicable). The RDA bit is set to 1 to indicate that the Receive Data Register contains a data byte. However, because the overrun error occurred, this byte cannot contain valid data and should be ignored. The BRKD bit indicates if the overrun was caused by a break condition on the line. After reading the status byte indicating an overrun error, the Receive Data Register must be read again to clear the error bits is the UART Status 0 Register. Updates to the Receive Data Register occur only when the next data word is received.

SPI Control Register Definitions

This section defines the features of the following Serial Peripheral Interface registers.

SPI Data Register: see page 109

SPI Control Register: see page 110

SPI Status Register: see page 111

SPI Mode Register: see page 112

SPI Diagnostic State Register: see page 113

SPI Baud Rate High and Low Byte Registers: see page 114

SPI Data Register

The SPI Data Register, shown in Table 64, stores both the outgoing (transmit) data and the incoming (receive) data. Reads from the SPI Data Register always return the current contents of the 8-bit Shift Register. Data is shifted out starting with bit 7. The last bit received resides in bit position 0.

With the SPI configured as a Master, writing a data byte to this register initiates the data transmission. With the SPI configured as a Slave, writing a data byte to this register loads the shift register in preparation for the next data transfer with the external Master. In either the Master or Slave modes, if a transmission is already in progress, writes to this register are ignored and the Overrun error flag, OVR, is set in the SPI Status Register.

When the character length is less than 8 bits (as set by the NUMBITS field in the SPI Mode Register), the transmit character must be set as *left-justified* in the SPI Data Register. A received character of less than 8 bits is right justified (last bit received is in bit position 0). For example, if the SPI is configured for 4-bit characters, the transmit characters must be written to SPIDATA[7:4] and the received characters are read from SPIDATA[3:0].

Table 64. SPI Data Register (SPIDATA)

Bit	7	6	5	4	3	2	1	0
Field	DATA							
RESET	X							
R/W	R/W							
Address	F60H							

Bit	Description
[7:0] DATA	**SPI Data** Transmit and/or receive data.

SPI Status Register

The SPI Status Register, shown in Table 66, indicates the current state of the SPI. All bits revert to their reset state if the SPIEN bit in the SPICTL Register equals 0.

Table 66. SPI Status Register (SPISTAT)

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	IRQ	OVR	COL	ABT	Reserved		TXST	SLAS
RESET	0							1
R/W	R/W*				R			
Address	F62H							
Note: *R/W = read access; write a 1 to clear the bit to 0.								

Bit	Description
[7] IRQ	**Interrupt Request** If SPIEN = 1, this bit is set if the STR bit in the SPICTL Register is set, or upon completion of an SPI Master or Slave transaction. This bit does not set if SPIEN = 0 and the SPI Baud Rate Generator is used as a timer to generate the SPI interrupt. 0 = No SPI interrupt request pending. 1 = SPI interrupt request is pending.
[6] OVR	**Overrun** 0 = An overrun error has not occurred. 1 = An overrun error has been detected.
[5] COL	**Collision** 0 = A multimaster collision (mode fault) has not occurred. 1 = A multimaster collision (mode fault) has been detected.
[4] ABT	**SLAVE Mode Transaction Abort** This bit is set if the SPI is configured in SLAVE Mode, a transaction is occurring and $\overline{SS}$ deasserts before all bits of a character have been transferred as defined by the NUMBITS field of the SPIMODE Register. The IRQ bit also sets, indicating the transaction has completed. 0 = A SLAVE Mode transaction abort has not occurred. 1 = A SLAVE Mode transaction abort has been detected.
[3:2]	**Reserved** These bits are reserved and must be programmed to 00.
[1] TXST	**Transmit Status** 0 = No data transmission currently in progress. 1 = Data transmission currently in progress.
[0] SLAS	**Slave Select** If SPI is enabled as a Slave, then the following bit settings are true: 0 = $\overline{SS}$ input pin is asserted (Low) 1 = $\overline{SS}$ input is not asserted (High). If SPI is enabled as a Master, this bit is not applicable.

SDA and SCL Signals

I^2C sends all addresses, data and acknowledge signals over the SDA line, most-significant bit first. SCL is the common clock for the I^2C Controller. When the SDA and SCL pin alternate functions are selected for their respective GPIO ports, the pins are automatically configured for open-drain operation.

The master (I^2C) is responsible for driving the SCL clock signal, although the clock signal becomes skewed by a slow slave device. During the Low period of the clock, the slave pulls the SCL signal Low to suspend the transaction. The master releases the clock at the end of the Low period and notices that the clock remains Low instead of returning to a High level. When the slave releases the clock, the I^2C Controller continues the transaction. All data is transferred in bytes and there is no limit to the amount of data transferred in one operation. When transmitting data or acknowledging read data from the slave, the SDA signal changes in the middle of the Low period of SCL and is sampled in the middle of the High period of SCL.

I^2C Interrupts

The I^2C Controller contains four sources of interrupts: Transmit, Receive, Not Acknowledge and Baud Rate Generator. These four interrupt sources are combined into a single interrupt request signal to the interrupt controller. The transmit interrupt is enabled by the IEN and TXI bits of the control register. The Receive and Not Acknowledge interrupts are enabled by the IEN bit of the control register. BRG interrupt is enabled by the BIRQ and IEN bits of the control register.

Not Acknowledge interrupts occur when a Not Acknowledge condition is received from the slave or sent by the I^2C Controller and neither the start or stop bit is set. The Not Acknowledge event sets the NCKI bit of the I^2C Status Register and can only be cleared by setting the start or stop bit in the I^2C Control Register. When this interrupt occurs, the I^2C Controller waits until either the stop or start bit is set before performing any action. In an ISR, the NCKI bit should always be checked prior to servicing transmit or receive interrupt conditions because it indicates the transaction is being terminated.

Receive interrupts occur when a byte of data has been received by the I^2C Controller (Master reading data from Slave). This procedure sets the RDRF bit of the I^2C Status Register. The RDRF bit is cleared by reading the I^2C Data Register. The RDRF bit is set during the acknowledge phase. The I^2C Controller pauses after the acknowledge phase until the receive interrupt is cleared before performing any other action.

Transmit interrupts occur when the TDRE bit of the I^2C Status Register sets and the TXI bit in the I^2C Control Register is set. Transmit interrupts occur under the following conditions when the Transmit Data Register is empty:

- The I^2C Controller is enabled

Bit	Description
[7] CEN	**Conversion Enable** 0 = Conversion is complete. Writing a 0 produces no effect. The ADC automatically clears this bit to 0 when a conversion has been completed. 1 = Begin conversion. Writing a 1 to this bit starts a conversion. If a conversion is already in progress, the conversion restarts. This bit remains 1 until the conversion is complete.
[6]	**Reserved** This bit is reserved and must be programmed to 0.
[5] VREF	**Voltage Reference** 0 = Internal reference generator enabled. The V_{REF} pin must remain unconnected or capacitively coupled to analog ground (AV_{SS}). 1 = Internal voltage reference generator disabled. An external voltage reference must be provided through the V_{REF} pin.
[4] CONT	**Conversion** 0 = SINGLE-SHOT conversion. ADC data is output one time at completion of the 5129 system clock cycles. 1 = Continuous conversion. ADC data updated every 256 system clock cycles.
[3] ANAIN[3:0]	**Analog Input Select** These bits select the analog input for conversion. Not all Port pins in this list are available in all packages for Z8 Encore! XP® F0822 Series. See the Signal and Pin Descriptions chapter on page 7 for information regarding the port pins available with each package style. Do not enable unavailable analog inputs. 0000 = ANA0. 0001 = ANA1. 0010 = ANA2. 0011 = ANA3. 0100 = ANA4. 0101 = Reserved. 011X = Reserved. 1XXX = Reserved.

Exiting Debug Mode

The device exits DEBUG Mode following any of the following operations:

- Clearing the DBGMODE bit in the OCD Control Register to 0
- Power-On Reset
- Voltage Brown-Out reset
- Asserting the 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; see Figure 37.

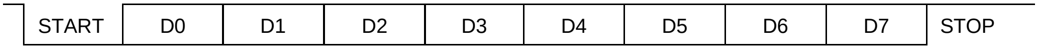

Figure 37. OCD Data Format

OCD Autobaud Detector/Generator

To run over a range of baud rates (bits per second) with various system clock frequencies, the OCD contains an Autobaud 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). The Autobaud Detector measures this period and sets the OCD Baud Rate Generator accordingly.

The Autobaud Detector/Generator is clocked by the system clock. The minimum baud rate is the system clock frequency divided by 512. For optimal operation, the maximum recommended baud rate is the system clock frequency divided by 8. The theoretical maximum baud rate is the system clock frequency divided by 4. This theoretical maximum is possible for low-noise designs with clean signals. Table 92 lists minimum and recommended maximum baud rates for sample crystal frequencies.

OCD Status Register

The OCD Status Register, shown in Table 95, reports status information about the current state of the debugger and the system.

Table 95. OCD Status Register (OCDSTAT)

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

Bit	**Description**
[7] IDLE	**CPU Idling** This bit is set if the part is in DEBUG Mode (DBGMODE is 1), or if a BRK instruction occurred since the last time OCDCTL was written. This can be used to determine if the CPU is running or if it is idling. 0 = The eZ8 CPU is running. 1 = The eZ8 CPU is either stopped or looping on a BRK instruction.
[6] HALT	**HALT Mode** 0 = The device is not in HALT Mode. 1 = The device is in HALT Mode.
[5] RPEN	**Read Protect Option Bit Enabled** 0 = The Read Protect option bit is disabled (1). 1 = The Read Protect option bit is enabled (0), disabling many OCD commands.
[4:0]	**Reserved** These bits are reserved and must be programmed to 00000.

Table 101 provides information about the external RC oscillator electrical characteristics and timing, and Table 102 provides information about the Flash memory electrical characteristics and timing.

Table 101. External RC Oscillator Electrical Characteristics and Timing

		T_A = –40°C to 105°C				
Symbol	**Parameter**	**Minimum**	**Typical***	**Maximum**	**Units**	**Conditions**
V_{DD}	Operating Voltage Range	2.70	–	–	V	
R_{EXT}	External Resistance from X_{IN} to V_{DD}	40	45	200	kΩ	
C_{EXT}	External Capacitance from X_{IN} to V_{SS}	0	20	1000	pF	
F_{OSC}	External RC Oscillation Frequency	–	–	4	MHz	

Note: *When using the external RC oscillator mode, the oscillator can stop oscillating if the power supply drops below 2.7V, but before the power supply drops to the voltage brown-out threshold. The oscillator will resume oscillation as soon as the supply voltage exceeds 2.7V.

Table 102. Flash Memory Electrical Characteristics and Timing

	V_{DD} = 2.7–3.6V T_A = –40°C to 105°C				
Parameter	**Minimum**	**Typical**	**Maximum**	**Units**	**Notes**
Flash Byte Read Time	50	–	–	µs	
Flash Byte Program Time	20	–	40	µs	
Flash Page Erase Time	10	–	–	ms	
Flash Mass Erase Time	200	–	–	ms	
Writes to Single Address Before Next Erase	–	–	2		
Flash Row Program Time	–	–	8	ms	Cumulative program time for single row cannot exceed limit before next erase. This parameter is only an issue when bypassing the Flash Controller.
Data Retention	100	–	–	years	25°C
Endurance	10,000	–	–	cycles	Program/erase cycles

SPI MASTER Mode Timing

Figure 51 and Table 109 provide timing information for SPI MASTER Mode pins. Timing is shown with SCK rising edge used to source MOSI output data, SCK falling edge used to sample MISO input data. Timing on the SS output pin(s) is controlled by software.

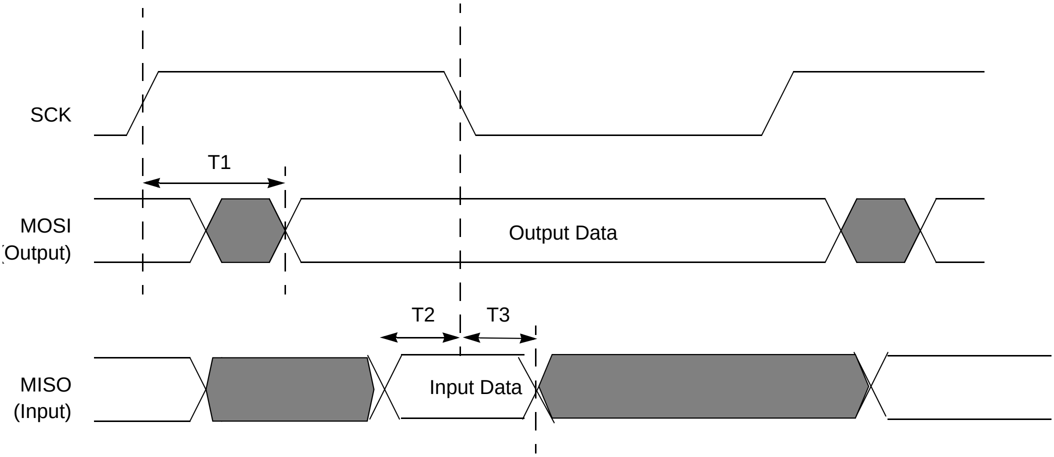

Figure 51. SPI MASTER Mode Timing

Table 109. SPI MASTER Mode Timing

		Delay (ns)	
Parameter	**Abbreviation**	**Minimum**	**Maximum**
SPI MASTER			
T_1	SCK Rise to MOSI output Valid Delay	–5	+5
T_2	MISO input to SCK (receive edge) Setup Time	20	
T_3	MISO input to SCK (receive edge) Hold Time	0	

resented here by $\overline{\text{DE}}$. $\overline{\text{DE}}$ asserts after the UART Transmit Data Register has been written. $\overline{\text{DE}}$ remains asserted for multiple characters as long as the Transmit Data Register is written with the next character before the current character has completed.

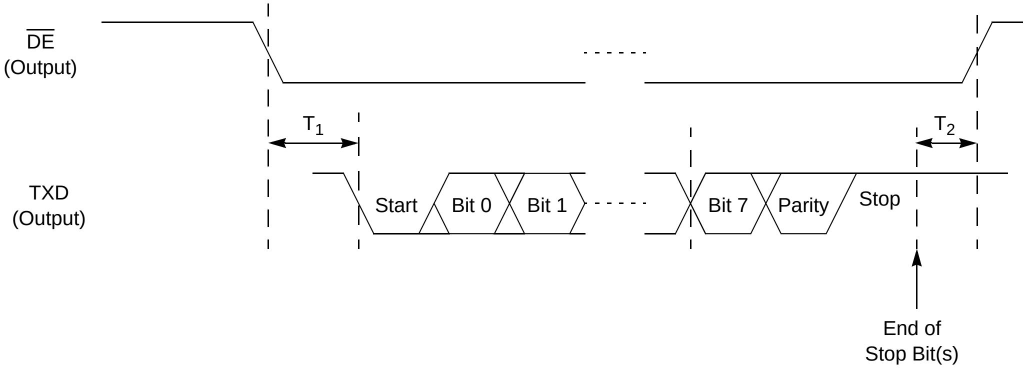

Figure 55. UART Timing without $\overline{\text{CTS}}$

Table 113. UART Timing without $\overline{\text{CTS}}$

		Delay (ns)	
Parameter	**Abbreviation**	**Minimum**	**Maximum**
T_1	$\overline{\text{DE}}$ Assertion to TXD Falling Edge (Start) Delay	1 Bit period	1 Bit period + 1 * X_{IN} period
T_2	End of Stop Bit(s) to $\overline{\text{DE}}$ Deassertion Delay	1 * X_{IN} period	2 * X_{IN} period

Table 127. eZ8 CPU Instruction Summary (Continued)

		Address Mode		Op	Flags							
Assembly Mnemonic	**Symbolic Operation**	**dst**	**src**	**Code(s) (Hex)**	**C**	**Z**	**S**	**V**	**D**	**H**	**Fetch Cycles**	**Instr. Cycles**
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
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

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

Index