



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	I <sup>2</sup> C, IrDA, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	31
Program Memory Size	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 3.6V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	44-LQFP
Supplier Device Package	44-LQFP (10x10)
Purchase URL	<a href="https://www.e-xfl.com/product-detail/zilog/z8f1621an020sg">https://www.e-xfl.com/product-detail/zilog/z8f1621an020sg</a>

Table 141.	Timer 0–3 Reload Low Byte Register (TxRL)	249
Table 142.	Timer 0–3 PWM High Byte Register (TxPWMH)	249
Table 143.	Timer 0–3 PWM Low Byte Register (TxPWML)	250
Table 144.	Timer 0–3 Control 0 Register (TxCTL0)	250
Table 145.	Timer 0–3 Control 1 Register (TxCTL1)	250
Table 146.	Timer 0–3 High Byte Register (TxH)	250
Table 147.	Timer 0–3 Low Byte Register (TxL)	251
Table 148.	Timer 0–3 Reload High Byte Register (TxRH)	251
Table 149.	Timer 0–3 Reload Low Byte Register (TxRL)	251
Table 150.	Timer 0–3 PWM High Byte Register (TxPWMH)	251
Table 151.	Timer 0–3 PWM Low Byte Register (TxPWML)	252
Table 152.	Timer 0–3 Control 0 Register (TxCTL0)	252
Table 153.	Timer 0–3 Control 1 Register (TxCTL1)	252
Table 154.	Timer 0–3 High Byte Register (TxH)	252
Table 155.	Timer 0–3 Low Byte Register (TxL)	253
Table 156.	Timer 0–3 Reload High Byte Register (TxRH)	253
Table 157.	Timer 0–3 Reload Low Byte Register (TxRL)	253
Table 158.	Timer 0–3 PWM High Byte Register (TxPWMH)	253
Table 159.	Timer 0–3 PWM Low Byte Register (TxPWML)	254
Table 160.	Timer 0–3 Control 0 Register (TxCTL0)	254
Table 161.	Timer 0–3 Control 1 Register (TxCTL1)	254
Table 162.	Timer 0–3 High Byte Register (TxH)	254
Table 163.	Timer 0–3 Low Byte Register (TxL)	255
Table 164.	Timer 0–3 Reload High Byte Register (TxRH)	255
Table 165.	Timer 0–3 Reload Low Byte Register (TxRL)	255
Table 166.	Timer 0–3 PWM High Byte Register (TxPWMH)	255
Table 167.	Timer 0–3 PWM Low Byte Register (TxPWML)	256
Table 168.	Timer 0–3 Control 0 Register (TxCTL0)	256
Table 169.	Timer 0–3 Control 1 Register (TxCTL1)	256
Table 170.	UART Transmit Data Register (UxTXD)	257
Table 171.	UART Receive Data Register (UxRXD)	257
Table 172.	UART Status 0 Register (UxSTAT0)	257
Table 173.	UART Control 0 Register (UxCTL0)	257
Table 174.	UART Control 1 Register (UxCTL1)	258
Table 175.	UART Status 1 Register (UxSTAT1)	258
Table 176.	UART Address Compare Register (UxADDR)	258

- 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 on [www.zilog.com](http://www.zilog.com).

## **General-Purpose Input/Output**

The Z8 Encore! XP F64xx Series features seven 8-bit ports (ports A–G) and one 4-bit port (Port H) for general-purpose input/output (GPIO). Each pin is individually programmable. All ports (except B and H) support 5 V-tolerant inputs.

## **Flash Controller**

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

## **10-Bit Analog-to-Digital Converter**

The Analog-to-Digital Converter converts an analog input signal to a 10-bit binary number. The ADC accepts inputs from up to 12 different analog input sources.

## **UARTs**

Each UART is full-duplex and capable of handling asynchronous data transfers. The UARTs support 8- and 9-bit data modes, selectable parity, and an efficient bus transceiver Driver Enable signal for controlling a multitransceiver bus, such as RS-485.

**Table 5. Z8 Encore! XP F64xx Series Program Memory Maps (Continued)**

Program Memory Address (Hex)	Function
0006-0007	Illegal Instruction Trap
0008-0037	Interrupt Vectors*
0038-BFFF	Program Memory
<b>Z8F642x Products</b>	
0000-0001	Option Bits
0002-0003	Reset Vector
0004-0005	WDT Interrupt Vector
0006-0007	Illegal Instruction Trap
0008-0037	Interrupt Vectors*
0038-FFFF	Program Memory
Note: *See <a href="#">Table 23</a> on page 48 for a list of the interrupt vectors.	

## Data Memory

The Z8 Encore! XP F64xx Series does not use the eZ8 CPU's 64KB data memory address space.

## Information Area

Table 6 describes the Z8 Encore! XP F64xx Series' Information Area. This 512-byte Information Area is accessed by setting bit 7 of the Page Select Register to 1. When access is enabled, the Information Area is mapped into program memory and overlays the 512 bytes at addresses FE00H to FFFFH. When the Information Area access is enabled, execution of the LDC and LDCI instructions from these program memory addresses return the Information Area data rather than the program memory data. Reads of these addresses through the On-Chip Debugger also returns the Information Area data. Execution of code from these addresses continues to correctly use program memory. Access to the Information Area is read-only.

## System Reset

During a system reset, the Z8 Encore! XP F64xx Series devices are held in Reset for 66 cycles of the Watchdog Timer oscillator followed by 16 cycles of the system clock. At the beginning of Reset, all GPIO pins are configured as inputs.

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

Upon Reset, control registers within the Register File that 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 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 text following provides more detailed information about the individual Reset sources. A Power-On Reset/Voltage Brown-Out event always takes 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	Power-On Reset/Voltage Brown-Out	system reset
	Watchdog Timer time-out when configured for Reset	system reset
	RESET pin assertion	system reset
	On-Chip Debugger initiated Reset (OCDCTL[0] set to 1)	system reset except the On-Chip Debugger is unaffected by the reset
STOP Mode	Power-On Reset/Voltage Brown-Out	system reset
	RESET pin assertion	system reset
	DBG pin driven Low	system reset

- Writing a 1 to the IRQE bit in the Interrupt Control Register

Interrupts are globally disabled by any of the following operations:

- Execution of a Disable Interrupt (DI) instruction
- eZ8 CPU acknowledgement of an interrupt service request from the interrupt controller
- Writing a 0 to the IRQE bit in the Interrupt Control Register
- Reset
- Executing a trap instruction
- Illegal instruction trap

## Interrupt Vectors and Priority

The interrupt controller supports three levels of interrupt priority. Level 3 is the highest priority, Level 2 is the second highest priority, and Level 1 is the lowest priority. If all of the interrupts were enabled with identical interrupt priority (all as Level 2 interrupts, for example), then the interrupt priority would be assigned from highest to lowest, as specified in Table 23. Level 3 interrupts always have higher priority than Level 2 interrupts which, in turn, always have higher priority than Level 1 interrupts. Within each interrupt priority level (Level 1, Level 2, or Level 3), priority is assigned as specified in Table 23. Resets, Watchdog Timer interrupts (if enabled), and illegal instruction traps always have highest priority.

## Interrupt Assertion

Interrupt sources assert their interrupt requests for only a single system clock period (single pulse). When the interrupt request is acknowledged by the eZ8 CPU, the corresponding bit in the Interrupt Request Register is cleared until the next interrupt occurs. Writing a 0 to the corresponding bit in the Interrupt Request Register likewise clears the interrupt request.

---

**!** **Caution:** Zilog recommends not using a coding style that clears bits in the Interrupt Request registers. All incoming interrupts received between execution of the first LDX command and the final LDX command are lost. See Example 1, which follows.

---

**Example 1.** A poor coding style that can result in lost interrupt requests:

```
LDX r0, IRQ0
AND r0, MASK
LDX IRQ0, r0
```

## Interrupt Edge Select Register

The Interrupt Edge Select (IRQES) Register, shown in Table 36, determines whether an interrupt is generated for the rising edge or falling edge on the selected GPIO port input pin. The Interrupt Port Select Register selects between Port A and Port D for the individual interrupts.

**Table 36. Interrupt Edge Select Register (IRQES)**

Bit	7	6	5	4	3	2	1	0
Field	IES7	IES6	IES5	IES4	IES3	IES2	IES1	IES0
RESET	0							
R/W	R/W							
Address	FCDH							

Bit	Description
[7:0] IESx	<b>Interrupt Edge Select x</b> The minimum pulse width should be greater than 1 system clock to guarantee capture of the edge triggered interrupt. Shorter pulses may be captured but not guaranteed. 0 = An interrupt request is generated on the falling edge of the PAX/PDx input. 1 = An interrupt request is generated on the rising edge of the PAX/PDx input.

Note: x indicates specific GPIO port pins in the range [7:0].

## Interrupt Port Select Register

The Port Select (IRQPS) Register, shown in Table 37, determines the port pin that generates the PAX/PDx interrupts. This register allows either Port A or Port D pins to be used as interrupts. The Interrupt Edge Select Register controls the active interrupt edge.

**Table 37. Interrupt Port Select Register (IRQPS)**

Bit	7	6	5	4	3	2	1	0
Field	PAD7S	PAD6S	PAD5S	PAD4S	PAD3S	PAD2S	PAD1S	PAD0S
RESET	0							
R/W	R/W							
Address	FCEH							

Bit	Description
[7:0] PADxS	<b>PAX/PDx Selection</b> 0 = PAX is used for the interrupt for PAX/PDx interrupt request. 1 = PDx is used for the interrupt for PAX/PDx interrupt request.

Note: x indicates specific GPIO port pins in the range [7:0].

6. Read data from the UART Receive Data Register. If operating in MULTIPROCESSOR (9-Bit) Mode, further actions may be required depending on the MULTIPROCESSOR Mode bits MPMD[1:0].
7. Return to Step 5 to receive additional data.

## Receiving Data using the Interrupt-Driven Method

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

1. Write to the UART Baud Rate High and Low Byte registers to set the appropriate 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 appropriate 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 appropriate 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.



## UART Baud Rate Generator

The UART Baud Rate Generator creates a lower frequency baud rate clock for data transmission. The input to the Baud Rate Generator is the system clock. The UART Baud Rate High and Low Byte registers combine to create a 16-bit baud rate divisor value (BRG[15:0]) that sets the data transmission rate (baud rate) of the UART. The UART data rate is calculated using the following equation:

$$\text{UART Data Rate (bits/s)} = \frac{\text{System Clock Frequency (Hz)}}{16 \times \text{UART Baud Rate Divisor Value}}$$

When the UART is disabled, the Baud Rate Generator can function as a basic 16-bit timer with interrupt on time-out. To configure the Baud Rate Generator as a timer with interrupt on time-out, complete the following procedure:

1. Disable the UART by clearing the REN and TEN bits in the UART Control 0 Register to 0.
2. Load the appropriate 16-bit count value into the UART Baud Rate High and Low Byte registers.
3. Enable the Baud Rate Generator timer function and associated interrupt by setting the BRGCTL bit in the UART Control 1 Register to 1.

When configured as a general purpose timer, the interrupt interval is calculated using the following equation:

$$\text{Interrupt Interval(s)} = \text{System Clock Period (s)} \times \text{BRG[15:0]}$$

## UART Control Register Definitions

The UART control registers support the UART and the associated Infrared Encoder/Decoders. For more information about the infrared operation, see the [Infrared Encoder/Decoder](#) chapter on page 109.

## UART Transmit Data Register

Data bytes written to the UART Transmit Data Register, shown in Table 53, are shifted out on the TXDx pin. The write-only UART Transmit Data Register shares a Register File address with the read-only UART Receive Data Register.

Bit	Description (Continued)
[2] TDRE	<b>Transmitter Data Register Empty</b> 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.
[1] TXE	<b>Transmitter Empty</b> This bit indicates that the Transmit Shift Register is empty and character transmission is finished. 0 = Data is currently transmitting. 1 = Transmission is complete.
[0] CTS	<b>CTS Signal</b> When this bit is read, it returns the level of the $\overline{\text{CTS}}$ signal.

## UART Status 1 Register

The UART Status 1 Register, shown in Table 56, contains multiprocessor control and UART status bits.

Table 56. UART Status 1 Register (UxSTAT1)

Bit	7	6	5	4	3	2	1	0
Field	Reserved						NEWFRM	MPRX
RESET	0							
R/W	R				R/W		R	
Address	F44H and F4CH							

Bit	Description
[7:2]	<b>Reserved</b> These bits are reserved and must be programmed to 000000.
[1] NEWFRM	<b>New Frame</b> Status bit denoting the start of a new frame. Reading the UART Receive Data Register resets this bit to 0. 0 = The current byte is not the first data byte of a new frame. 1 = The current byte is the first data byte of a new frame.
[0] MPRX	<b>Multiprocessor Receive</b> Returns the value of the last multiprocessor bit received. Reading from the UART Receive Data Register resets this bit to 0.

**Table 60. UART Baud Rate High Byte Register (UxBRH)**

Bit	7	6	5	4	3	2	1	0
Field	BRH							
RESET	1							
R/W	R/W							
Address	F46H and F4EH							

**Table 61. UART Baud Rate Low Byte Register (UxBRL)**

Bit7	7	6	5	4	3	2	1	0
Field	BRL							
RESET	1							
R/W	R/W							
Address	F47H and F4FH							

For a given UART data rate, the integer baud rate divisor value is calculated using the following equation:

$$\text{UART Baud Rate Divisor Value (BRG)} = \text{Round}\left(\frac{\text{System Clock Frequency (Hz)}}{16 \times \text{UART Data Rate (bits/s)}}\right)$$

The baud rate error relative to the appropriate baud rate is calculated using the following equation:

$$\text{UART Baud Rate Error (\%)} = 100 \times \left( \frac{\text{Actual Data Rate} - \text{Desired Data Rate}}{\text{Desired Data Rate}} \right)$$

For reliable communication, the UART baud rate error must never exceed 5 percent. Table 62 lists data rate errors for popular baud rates and commonly used crystal oscillator frequencies.

Table 111 list the Flash memory electrical characteristics and timing.

**Table 111. Flash Memory Electrical Characteristics and Timing**

$V_{DD} = 3.0\text{--}3.6\text{ V}$ $T_A = -40^{\circ}\text{C to } 125^{\circ}\text{C}$					
Parameter	Minimum	Typical	Maximum	Units	Notes
Flash Byte Read Time	50	–	–	ns	
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, –40°C to 105°C	10,000	–	–	cycles	Program/erase cycles
Endurance, 106°C to 125°C	1,000	–	–	cycles	Program/erase cycles

Table 112 lists the Watchdog Timer electrical characteristics and timing.

**Table 112. Watchdog Timer Electrical Characteristics and Timing**

V <sub>DD</sub> = 3.0–3.6V T <sub>A</sub> = –40°C to 125°C						
Symbol	Parameter	Minimum	Typical	Maximum	Units	Conditions
F <sub>WDT</sub>	WDT Oscillator Frequency	5	10	20	kHz	
I <sub>WDT</sub>	WDT Oscillator Current including internal RC Oscillator	–	<1	5	μA	

Figure 57 and Table 122 provide timing information for UART pins for the case where the Clear To Send input signal ( $\overline{\text{CTS}}$ ) is not used for flow control. In this example, it is assumed that the Driver Enable polarity has been configured to be Active Low and is represented 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.

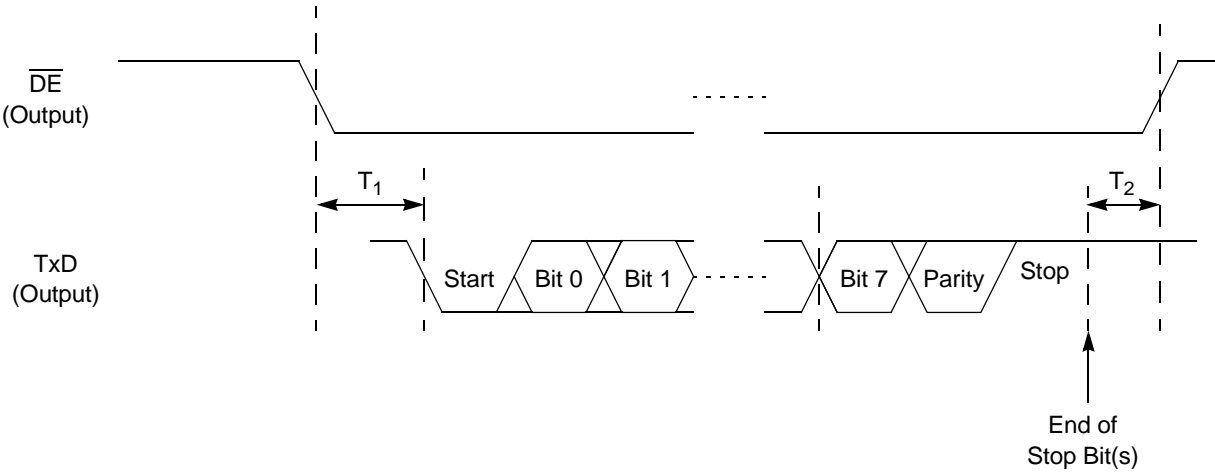


Figure 57. UART Timing without  $\overline{\text{CTS}}$

Table 122. UART Timing without  $\overline{\text{CTS}}$

Parameter	Abbreviation	Delay (ns)	
		Minimum	Maximum
T <sub>1</sub>	$\overline{\text{DE}}$ Assertion to TxD Falling Edge (Start) Delay	1 bit period	1 bit period + 1 * X <sub>IN</sub> period
T <sub>2</sub>	End of stop bit(s) to $\overline{\text{DE}}$ Deassertion Delay	1 * X <sub>IN</sub> period	2 * X <sub>IN</sub> period

**Table 124. Assembly Language Syntax Example 2**

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

Refer to the device-specific Product Specification to determine the exact register file range available. The register file size varies, depending on the device type.

## eZ8 CPU Instruction Notation

In the eZ8 CPU Instruction Summary and Description sections, the operands, condition codes, status flags, and address modes are represented by a notational shorthand that is described in Table 125.

**Table 125. Notational Shorthand**

Notation	Description	Operand	Range
b	Bit	b	b represents a value from 0 to 7 (000B to 111B).
cc	Condition Code	—	Refer to Condition Codes overview in the eZ8 CPU User Manual.
DA	Direct Address	Addr	Addr. represents a number in the range of 0000H to FFFFH.
ER	Extended Addressing Register	Reg	Reg. represents a number in the range of 000H to FFH.
IM	Immediate Data	#Data	Data is a number between 00H to FFH.
Ir	Indirect Working Register	@Rn	n = 0 – 15.
IR	Indirect Register	@Reg	Reg. represents a number in the range of 00H to FFH.
Irr	Indirect Working Register Pair	@RRp	p = 0, 2, 4, 6, 8, 10, 12, or 14.
IRR	Indirect Register Pair	@Reg	Reg. represents an even number in the range 00H to FEH.
p	Polarity	p	Polarity is a single bit binary value of either 0B or 1B.
r	Working Register	Rn	n = 0 – 15.
R	Register	Reg	Reg. represents a number in the range of 00H to FFH.

Table 135. Rotate and Shift Instructions

Mnemonic	Operands	Instruction
BSWAP	dst	Bit Swap
RL	dst	Rotate Left
RLC	dst	Rotate Left through Carry
RR	dst	Rotate Right
RRC	dst	Rotate Right through Carry
SRA	dst	Shift Right Arithmetic
SRL	dst	Shift Right Logical
SWAP	dst	Swap Nibbles

## eZ8 CPU Instruction Summary

Table 136 summarizes the eZ8 CPU instructions. The table identifies the addressing modes employed by the instruction, the effect upon the Flags Register, the number of CPU clock cycles required for the instruction fetch, and the number of CPU clock cycles required for the instruction execution.

Table 136. eZ8 CPU Instruction Summary

Assembly Mnemonic	Symbolic Operation	Address Mode		Opcode(s) (Hex)	Flags						Fetch Cycles	Instr. Cycles
		dst	src		C	Z	S	V	D	H		
ADC dst, src	$\text{dst} \leftarrow \text{dst} + \text{src} + \text{C}$	r	r	12	*	*	*	*	0	*	2	3
		r	lr	13							2	4
		R	R	14							3	3
		R	IR	15							3	4
		R	IM	16							3	3
		IR	IM	17							3	4
ADCX dst, src	$\text{dst} \leftarrow \text{dst} + \text{src} + \text{C}$	ER	ER	18	*	*	*	*	0	*	4	3
		ER	IM	19							4	3

Note: Flags Notation:

\* = Value is a function of the result of the operation.

– = Unaffected.

X = Undefined.

0 = Reset to 0.

1 = Set to 1.






		Lower Nibble (Hex)															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Upper Nibble (Hex)	0	1.2 BRK	2.2 SRP IM	2.3 ADD r1,r2	2.4 ADD r1,lr2	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,lr2	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 Opcode Map
	2	2.2 INC R1	2.3 INC IR1	2.3 SUB r1,r2	2.4 SUB r1,lr2	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,lr2	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,lr2	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,lr2	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,lr2	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,lr2	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,lr2	2.9 LDEI lr1,lr2	3.2 LDX r1,ER2	3.3 LDX lr1,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	2.3 LDE r2,lr1	2.9 LDEI lr2,lr1	3.2 LDX r2,ER1	3.3 LDX lr2,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,lr2	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,lr2	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,lr2	2.9 LDCI lr1,lr2	2.3 JP IRR1	2.9 LDC lr1,lr2		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,lr1	2.9 LDCI lr2,lr1	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,lr2	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 lr1,r2	2.8 MULT RR1	3.3 LD R2,IR1	3.3 BTJ p,b,r1,X	3.4 BTJ p,b,lr1,X								

Figure 60. First Op Code Map



**Hex Address: F09**

**Table 147. Timer 0–3 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, F11H, F19H							

**Hex Address: F0A**

**Table 148. Timer 0–3 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, F12H, F1AH							

**Hex Address: F0B**

**Table 149. Timer 0–3 Reload Low Byte Register (TxRL)**

Bit	7	6	5	4	3	2	1	0
Field	TRL							
RESET	1							
R/W	R/W							
Address	F03H, F0BH, F13H, F1BH							

**Hex Address: F0C**

**Table 150. Timer 0–3 PWM High Byte Register (TxPWMH)**

Bit	7	6	5	4	3	2	1	0
Field	PWMH							
RESET	0							
R/W	R/W							
Address	F04H, F0CH, F14H, F1CH							

**Hex Address: F64**

**Table 199. SPI Diagnostic State Register (SPIDST)**

Bit	7	6	5	4	3	2	1	0
Field	SCKEN	TCKEN	SPISTATE					
RESET	0							
R/W	R							
Address	F64H							

**Hex Address: F65**

This address is reserved.

**Hex Address: F66**

**Table 200. SPI Baud Rate High Byte Register (SPIBRH)**

Bit	7	6	5	4	3	2	1	0
Field	BRH							
RESET	1							
R/W	R/W							
Address	F66H							

**Hex Address: F67**

**Table 201. SPI Baud Rate Low Byte Register (SPIBRL)**

Bit	7	6	5	4	3	2	1	0
Field	BRL							
RESET	1							
R/W	R/W							
Address	F67H							

**Hex Addresses: F68–F6F**

This address range is reserved.

**Hex Address: FC1**

**Table 218. IRQ0 Enable High Bit Register (IRQ0ENH)**

Bit	7	6	5	4	3	2	1	0
Field	T2ENH	T1ENH	T0ENH	U0RENH	U0TENH	I2CENH	SPIENH	ADCENH
RESET	0							
R/W	R/W							
Address	FC1H							

**Hex Address: FC2**

**Table 219. IRQ0 Enable Low Bit Register (IRQ0ENL)**

Bit	7	6	5	4	3	2	1	0
Field	T2ENL	T1ENL	T0ENL	U0RENL	U0TENL	I2CENL	SPIENL	ADCENL
RESET	0							
R/W	R/W							
Address	FC2H							

**Hex Address: FC3**

**Table 220. Interrupt Request 1 Register (IRQ1)**

Bit	7	6	5	4	3	2	1	0
Field	PAD7I	PAD6I	PAD5I	PAD4I	PAD3I	PAD2I	PAD1I	PAD0I
RESET	0							
R/W	R/W							
Address	FC3H							

**Hex Address: FC4**

**Table 221. IRQ1 Enable High Bit Register (IRQ1ENH)**

Bit	7	6	5	4	3	2	1	0
Field	PAD7ENH	PAD6ENH	PAD5ENH	PAD4ENH	PAD3ENH	PAD2ENH	PAD1ENH	PAD0ENH
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	FC4H							

**Hex Address: FD3****Table 232. Port A–H Output Data Register (PxOUT)**

Bit	7	6	5	4	3	2	1	0
Field	POUT7	POUT6	POUT5	POUT4	POUT3	POUT2	POUT1	POUT0
RESET	0							
R/W	R/W							
Address	FD3H, FD7H, FDBH, FDFH, FE3H, FE7H, FEBH, FEFH							

**Hex Address: FD4****Table 233. Port A–H GPIO Address Registers (PxADDR)**

Bit	7	6	5	4	3	2	1	0
Field	PADDR[7:0]							
RESET	00H							
R/W	R/W							
Address	FD0H, FD4H, FD8H, FDCH, FE0H, FE4H, FE8H, FECH							

**Hex Address: FD5****Table 234. Port A–H Control Registers (PxCTL)**

Bit	7	6	5	4	3	2	1	0
Field	PCTL							
RESET	00H							
R/W	R/W							
Address	FD1H, FD5H, FD9H, FDDH, FE1H, FE5H, FE9H, FEDH							

**Hex Address: FD6****Table 235. Port A–H 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, FDEH, FE2H, FE6H, FEAH, FEEH							

## ***Customer Support***

To share comments, get your technical questions answered or report issues you may be experiencing with our products, please visit Zilog's Technical Support page at <http://support.zilog.com>.

To learn more about this product, find additional documentation or to discover other facets about Zilog product offerings, please visit the Zilog Knowledge Base at <http://zilog.com/kb> or consider participating in the Zilog Forum at <http://zilog.com/forum>.

This publication is subject to replacement by a later edition. To determine whether a later edition exists, please visit the Zilog website at <http://www.zilog.com>.