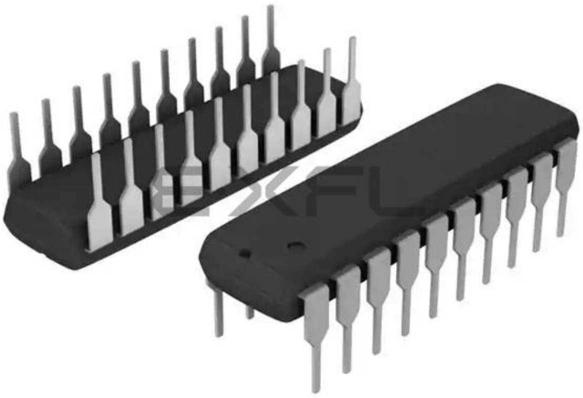
Zilog - Z8F083APH020SG Datasheet



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	-
Peripherals	Brown-out Detect/Reset, LED, POR, PWM, WDT
Number of I/O	17
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	A/D 7x10b
Oscillator Type	Internal
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Through Hole
Package / Case	20-DIP (0.300", 7.62mm)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f083aph020sg

List of Tables

Pin Description

The Z8 Encore! F083A Series products are available in variety of package styles and pin configurations. This chapter describes the signals and the pin configurations for each of the package styles. For information about the physical package specifications, see the Packaging chapter on page 198.

Available Packages

Table 3 lists the package styles that are available for each device in the Z8 Encore! F083A Series product line.

Table 3. Z8 Encore! F083A Series Package Options

Part Number	ADC	20-pin QFN	20-pin SOIC	20-pin SSOP	20-pin PDIP	28-pin QFN	28-pin SOIC	28-pin SSOP
Z8F083A	Yes	X	X	X	X	X	X	X
Z8F043A	Yes	X	X	X	X	X	X	X

Pin Configurations

Figures 2 through 5 display the pin configurations of all of the packages available in the Z8 Encore! F083A Series. For the description of the signals, see Table 4 on page 11.

The pin configurations listed are preliminary and subject to change based on manufacturing limitations.

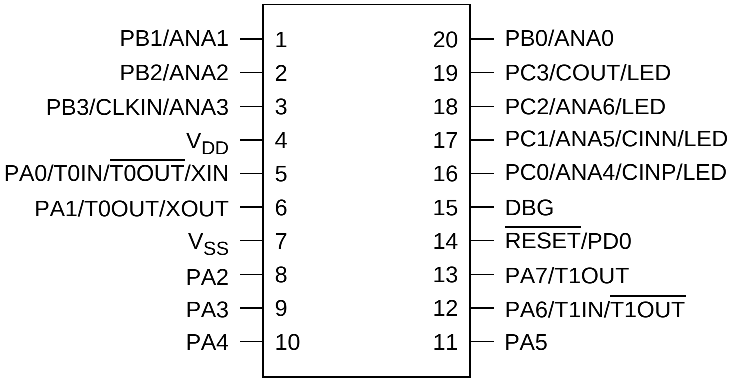


Figure 2. Z8F083A Series in 20-Pin SOIC, SSOP, PDIP Package

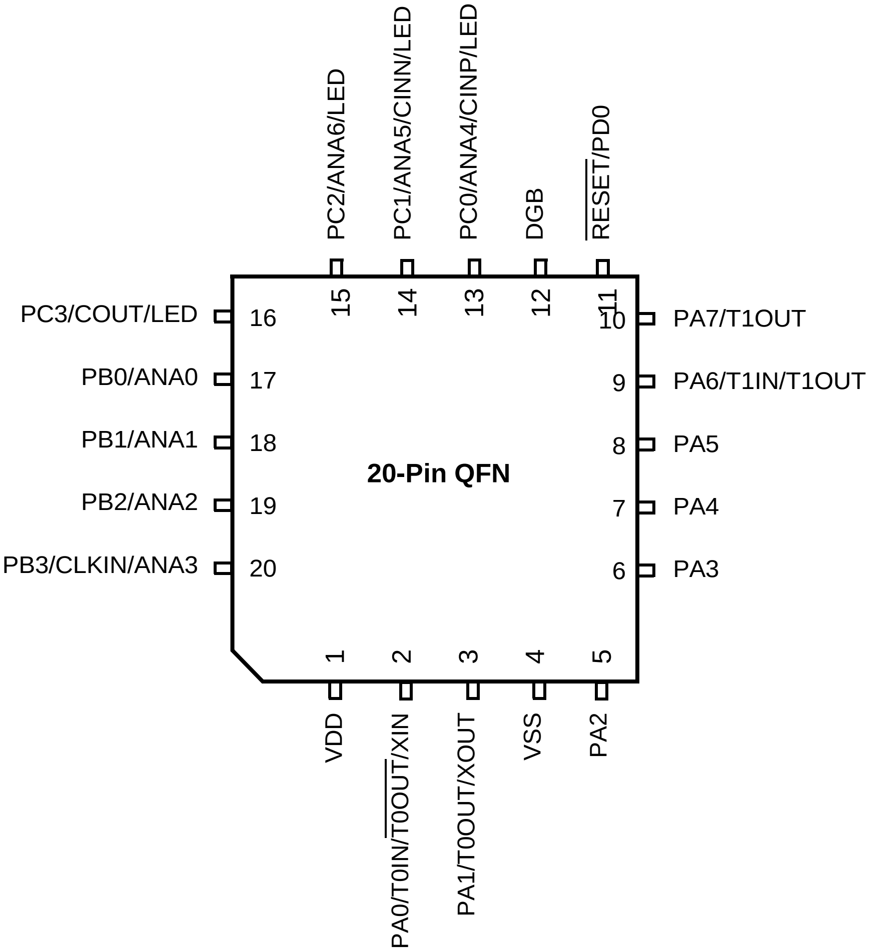


Figure 4. Z8F083A Series in 20-Pin QFN Package

Signal Descriptions

Table 4 describes the Z8 Encore! F083A Series signals. To determine the signals available for a specific package style, see the Pin Configurations section on page 7.

Table 4. Signal Descriptions

Signal Mnemonic	I/O	Description
General-Purpose Input/Output Ports A–D		
PA[7:0]	I/O	Port A. These pins are used for GPIO.
PB[5:0]	I/O	Port B. These pins are used for GPIO.
PC[7:0]	I/O	Port C. These pins are used for GPIO.
PD[0]	I/O	Port D. This pin is used for general purpose output only.
Timers		
T0OUT/T1OUT	O	Timer output 0–1. These signals are the output from the timers.
$\overline{\text{T0OUT}}$/$\overline{\text{T1OUT}}$	O	Timer complement output 0–1. These signals are output from the timers in PWM DUAL OUTPUT mode.
T0IN/T1IN	I	Timer Input 0–1. These signals are used as the capture, gating and counter inputs. The T0IN signal is multiplexed $\overline{\text{T0OUT}}$ signals.
Comparator		
CINP/CINN	I	Comparator inputs. These signals are the positive and negative inputs to the comparator.
COUT	O	Comparator output. This is the output of the comparator.
Analog		
ANA[7:0]	I	Analog port. These signals are used as inputs to the Analog-to-Digital Converter (ADC).
VREF	I/O	ADC reference voltage input. Note: When configuring ADC using external Vref, PB5 is used as VREF in 28-pin package.
Oscillators		
XIN	I	External crystal input. This is the input pin to the crystal oscillator. A crystal is connected between it and the **XOUT** pin to form the oscillator. In addition, this pin is used with external RC networks or external clock drivers to provide the system clock.
XOUT	O	External crystal output. This pin is the output of the crystal oscillator. A crystal is connected between it and the **XIN** pin to form the oscillator.
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.

Table 9. Reset and Stop Mode Recovery Characteristics and Latency

	Reset Characteristics and Latency		
Reset Type	**Control Registers**	**eZ8 CPU**	**Reset Latency (Delay)**
System Reset	Reset (as applicable)	Reset	About 66 internal precision oscillator cycles.
System Reset with Crystal Oscillator Enabled	Reset (as applicable)	Reset	About 5000 internal precision oscillator cycles.
Stop Mode Recovery	Unaffected, except WDT_CTL and OSC_CTL registers	Reset	About 66 internal precision oscillator cycles.
Stop Mode Recovery with crystal oscillator enabled	Unaffected, except WDT_CTL and OSC_CTL registers	Reset	About 5000 internal precision oscillator cycles.

During a system Reset or Stop Mode Recovery, the Z8 Encore! F083A Series device is held in reset for about 66 cycles of the internal precision oscillator. If the crystal oscillator is enabled in the Flash option bits, the reset period is increased to about 5000 IPO cycles. When a reset occurs because of a low voltage condition or POR, the reset delay is measured from the time the supply voltage first exceeds the POR level (discussed later in this chapter). If the external pin reset remains asserted at the end of the reset period, the device remains in reset until the pin is deasserted.

At the beginning of reset, all GPIO pins are configured as inputs with pull-up resistor disabled, except PD0 which is shared with the reset pin. On Reset, the Port D0 pin is configured as a bidirectional open-drain reset. This pin is internally driven low during port reset, after which the user code reconfigures this pin as a general purpose output.

During reset, the eZ8 CPU and on-chip peripherals are idle; however, the on-chip crystal oscillator and Watchdog Timer Oscillator continues to run.

On 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 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.

Because the control registers are reinitialized by a system reset, the system clock after reset is always the IPO. User software must reconfigure the oscillator control block, to enable and select the correct system clock source.

Port A–D High Drive Enable Subregisters

The Port A–D High Drive Enable Subregister, shown in Table 24, is accessed through the Port A–D Control Register by writing `04H` to the Port A–D Address Register. Setting the bits in the Port A–D High Drive Enable subregisters to 1 configures the specified port pins for high-output current drive operation. The Port A–D High Drive Enable Subregister affects the pins directly and, as a result, alternate functions are also affected.

Table 24. Port A–D High Drive Enable Subregisters (PxHDE)

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	PHDE7	PHDE6	PHDE5	PHDE4	PHDE3	PHDE2	PHDE1	PHDE0
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	If 04H in Port A–D Address Register, accessible through the Port A–D Control Register							

Bit	**Description**
[7:0] PHDEx	**Port High Drive Enable** 0 = The port pin is configured for standard output current drive. 1 = The port pin is configured for high output current drive.

Note: x indicates the specific GPIO port pin number (7–0).

Port A–D Stop Mode Recovery Source Enable Subregisters

The Port A–D Stop Mode Recovery Source Enable Subregister, shown in Table 25, is accessed through the Port A–D Control Register by writing `05H` to the Port A–D Address Register. Setting the bits in the Port A–D Stop Mode Recovery Source Enable subregisters to1, configures the specified port pins as a Stop Mode Recovery source. During STOP Mode, any logic transition on a port pin enabled as a Stop Mode Recovery source initiates Stop Mode Recovery.

Table 25. Port A–D Stop Mode Recovery Source Enable Subregisters (PxSMRE)

Bit	7	6	5	4	3	2	1	0
Field	PSMRE7	PSMRE6	PSMRE5	PSMRE4	PSMRE3	PSMRE2	PSMRE1	PSMRE0
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	If 05H in Port A–D Address Register, accessible through the Port A–D Control Register							

Bit	Description
[7:0] PSMREx	**Port Stop Mode Recovery Source Enable** 0 = The port pin is not configured as a Stop Mode Recovery source. Transitions on this pin during STOP Mode do not initiate Stop Mode Recovery. 1 = The port pin is configured as a Stop Mode Recovery source. Any logic transition on this pin during STOP Mode initiates Stop Mode Recovery.

Note: x indicates the specific GPIO port pin number (7–0).

Architecture

Figure 9 displays the Interrupt Controller block diagram.

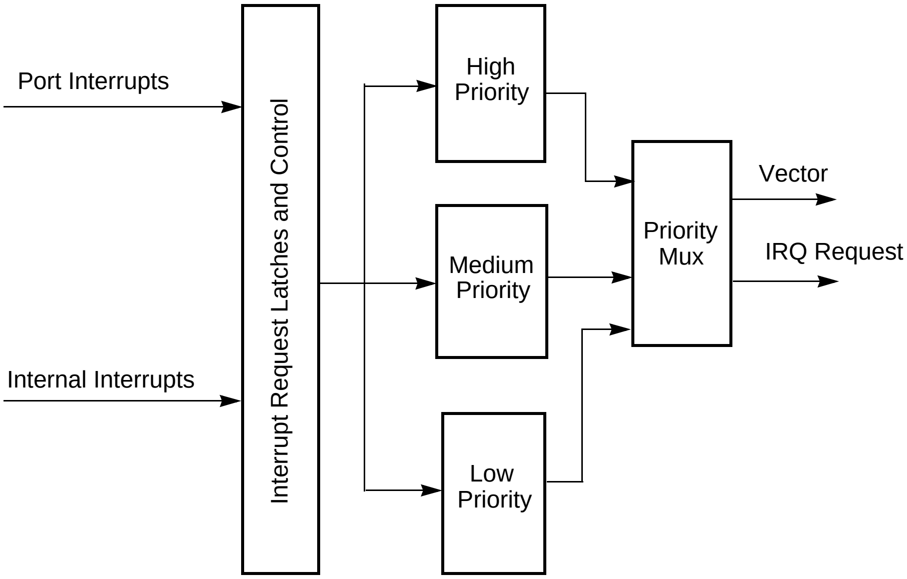


Figure 9. Interrupt Controller Block Diagram

Operation

This section describes the operational aspects of the following functions.

Master Interrupt Enable: see page 56

Interrupt Vectors and Priority: see page 57

Interrupt Assertion: see page 57

Software Interrupt Assertion: see page 58

Master Interrupt Enable

The master interrupt enable bit (`IRQE`) in the Interrupt Control Register globally enables and disables the interrupts.

Interrupts are globally enabled by any of the following actions:

- Execution of an enable interrupt (EI) instruction
- Execution of an return from interrupt (IRET) instruction

Interrupt Request 0 Register

The Interrupt Request 0 (IRQ0) Register, shown in Table 35, stores the interrupt requests for both vectored and polled interrupts. When a request is sent 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 35. Interrupt Request 0 Register (IRQ0)

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	Reserved	T1I	T0I	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
Address	FC0H							

Bit	Description
[7]	**Reserved** This bit is reserved and must be programmed to 0.
[6] T1I	**Timer 1 Interrupt Request** 0 = No interrupt request is pending for Timer 1. 1 = An interrupt request from timer 1 is awaiting service.
[5] T0I	**Timer 0 Interrupt Request** 0 = No interrupt request is pending for Timer 0. 1 = An interrupt request from timer 0 is awaiting service.
[4:1]	**Reserved** These bits are reserved and must be programmed to 0000.
[0] ADCI	**ADC Interrupt Request** 0 = No interrupt request is pending for the ADC. 1 = An interrupt request from the ADC is awaiting service.

Interrupt Edge Select Register

The Interrupt Edge Select (IRQES) Register, shown in Table 47, determines whether an interrupt is generated for the rising edge or falling edge on the selected GPIO Port A or Port D input pin.

Table 47. 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	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Address	FCDH							

Bit	**Description**
[7:0] IESx	**Interrupt Edge Select** 0 = An interrupt request is generated on the falling edge of the PA*x* input or PDx. 1 = An interrupt request is generated on the rising edge of the PA*x* input or PDx.
Note: x indicates register bits 7–0.	

4. Enable the timer interrupt 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 is 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 whether the 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. Upon 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). Additionally, 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.

Observe the following steps to configure a timer for GATED Mode and to initiate the count.

1. Write to the Timer Control Register to:
   - Disable the timer
   - Configure the timer for GATED Mode
   - Set the prescale value
2. Write to the Timer High and Low Byte registers to set the starting count value. Writing these registers only affects the first pass in GATED Mode. After the first timer reset in GATED Mode, counting always begins at the reset value of `0001H`.
3. Write to the Timer Reload High and Low Byte registers to set the reload value.
4. Enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers. By default, the timer interrupt is generated for both input deasser-

tion and reload events. Therefore, configure the timer interrupt to be generated only at the input deassertion event or the reload event by setting TICONFIG bit of the TxCTL1 Register.

5. Configure the associated GPIO port pin for the timer input alternate function.
6. Write to the Timer Control Register to enable the timer.
7. Assert the timer input signal to initiate the counting.

CAPTURE/COMPARE Mode

In CAPTURE/COMPARE Mode, the timer begins counting on the first external timer input transition. The acceptable transition (rising edge or falling edge) is set by the TPOL bit in the Timer Control Register. The timer input is the system clock.

Every subsequent acceptable transition (after the first) of the timer input signal, captures the current count value. The capture value is written to the timer PWM High and Low Byte registers. When a capture event occurs, an interrupt is generated, the count value in the Timer High and Low Byte registers is reset to `0001H` and the counting resumes. The INPCAP bit in the TxCTL1 Register is set to indicate that the timer interrupt is caused by an input capture event.

If no capture event occurs, the timer counts up to the 16-bit compare value stored in the Timer Reload High and Low Byte registers. Upon reaching the compare value, the timer generates an interrupt, the count value in the Timer High and Low Byte registers is reset to `0001H` and counting resumes. The INPCAP bit in the TxCTL1 Register is cleared to indicate that the timer interrupt is not caused by an input capture event.

Observe the following steps to configure a timer for CAPTURE/COMPARE Mode and to initiate the count.

1. Write to the Timer Control Register to:
   - Disable the timer
   - Configure the timer for CAPTURE/COMPARE Mode
   - Set the prescale value
   - Set the capture edge (rising or falling) for the timer input
2. Write to the Timer High and Low Byte registers to set the starting count value (typically `0001H`).
3. Write to the Timer Reload High and Low Byte registers to set the compare value.
4. Enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers.By default, the timer interrupt is generated for both input capture and reload events. The user must configure the timer interrupt to be generated only at the input capture event or the reload event by setting TICONFIG bit of the TxCTL1 Register.

Watchdog Timer Reload Upper, High and Low Byte Registers

The Watchdog Timer Reload Upper, High and Low Byte (WDTU, WDTH, WDTL) registers, shown in Tables 60 through 62, form the 24-bit reload value{WDTU[7:0], WDTH[7:0], WDTL[7:0]} that is loaded into the Watchdog Timer when a WDT instruction is executed. Writing to these registers sets the appropriate reload value; reading from these registers returns the current Watchdog Timer count value.

! **Caution:** The 24-bit WDT reload value must not be set to a value less than `000004H`.

Table 60. Watchdog Timer Reload Upper Byte Register (WDTU)

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	WDTU							
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	FF1H							
Note: *R/W = A read returns the current WDT count value; a write sets the appropriate reload value.								

Bit	**Description**
[7:0] WDTU	**WDT Reload Upper Byte** MSB, Bits[23:16], of the 24-bit WDT reload value.

Table 61. Watchdog Timer Reload High Byte Register (WDTH)

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	WDTH							
RESET	0	0	0	0	0	1	0	0
R/W	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*
Address	FF2H							
Note: *R/W = A read returns the current WDT count value; a write sets the appropriate reload value.								

Bit	**Description**
[7:0] WDTH	**WDT Reload High Byte** Middle byte, bits[15:8] of the 24-bit WDT reload value.

Bit	Description (Continued)
[1] IGADDR	**Illegal Address** When NVDS byte reads from invalid addresses occur (those exceeding the NVDS array size), this bit is set to 1.
[0]	**Reserved** This bit is reserved and must be programmed to 0.

Power Failure Protection

The NVDS routines employ error-checking mechanisms to ensure that any power failure will only endanger the most recently written byte. Bytes previously written to the array are not perturbed. For this protection to function, the VBO must be enabled (See *the* Low-Power Modes chapter on page 30) and configured for a threshold voltage of 2.4V or greater (See *the* Trim Bit Address Space *section on page 129*).

A system reset (such as a pin reset or Watchdog Timer reset) that occurs during a write operation also perturbs the byte currently being written. All other bytes in the array are unperturbed.

Optimizing NVDS Memory Usage for Execution Speed

As Table 94 shows, NVDS read times vary drastically, this discrepancy being a trade-off for minimizing the frequency of writes that require post-write page erases. The NVDS read time of address N is a function of the number of writes to addresses other than N since the most recent write to address N, as well as the number of writes since the most recent page erase. Neglecting effects caused by page erases and results caused by the initial condition in which the NVDS is blank, a rule of thumb is that every write since the most recent page erase causes read times of unwritten addresses to increase by 0.8 µs, up to a maximum of 258 µs.

Table 94. NVDS Read Time

Operation	Minimum Latency (µs)	Maximum Latency (µs)
Read	71	258
Write	126	136
Illegal Read	6	6
Illegal Write	7	7

```
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 is read 1–65536 bytes at a time (65536 bytes are 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 is written 1–65536 bytes at a time (65536 bytes are 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 is read from 1 to 65536 bytes at a time (65536 bytes are 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 CRC command, computes and returns the cyclic redundancy check (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 the other OCD read commands, there is a delay from issuing of the command until the OCD returns the data. The OCD reads program memory, calculates the CRC value and returns the result. The delay is a function of program memory size and is approximately equal to the system clock period multiplied by the number of bytes in program memory.

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 01H value is written into the
                  ; register at address 234H.
```

Assembly Language Syntax

For proper instruction execution, eZ8 CPU assembly language syntax requires that the operands be written as ‘destination, source’. After assembly, the object code usually has the operands in the order ‘source, destination’, but ordering is op code-dependent. The following instruction examples display the format of some basic assembly instructions and the resulting object code produced by the assembler. This binary format must be followed by users that prefer manual program coding or intend to implement their own assembler.

Example 1

If the contents of registers 43H and 08H are added and the result is stored in 43H, the assembly syntax and resulting object code is:

Table 102. Assembly Language Syntax Example 1

Assembly Language Code	ADD	43H,	08H	(ADD dst, src)
Object Code	04	08	43	(OPC src, dst)

Example 2

In general, when an instruction format requires an 8-bit register address, the address specify any register location in the range 0–255 or, using escaped mode addressing, a working register R0–R15. If the contents of register 43H and working register R8 are added and the result is stored in 43H, the assembly syntax and resulting object code is:

Figures 26 and 27 provide operation code mapping information about each of the eZ8 CPU instructions.

Lower Nibble (Hex)

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

Figure 26. First Op Code Map

Table 128. Power Consumption Reference Table

		Power Consumption	
Category	**Block**	**Typical**	**Maximum**
Logic	CPU/Peripherals @20 MHz	5 mA	
Flash	Flash@20 MHz		12 mA
Analog	ADC@20 MHz	4 mA	4.5 mA
	IPO@20 MHz	350 µA	400 µA
	Comparator@10 MHz	330 µA	450 µA
	POR & VBO	120 µA	150 µA
	WDT OSC	2 µA	3 µA
	OSC@20 MHz	600 µA	900 µA
	Clock Filter	120 µA	150 µA

Note: The power consumption values in this table are subject to change upon characterization.

Figure 33. Flash Current Diagram

Hex Address: FC5

Table 164. IRQ1 Enable Low Bit Register (IRQ1ENL)

Bit	7	6	5	4	3	2	1	0
Field	PA7ENL	PA6CENL	PA5ENL	PA4ENL	PA3ENL	PA2ENL	PA1ENL	PA0ENL
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	FC5H							

Hex Address: FC6

Table 165. Interrupt Request 2 Register (IRQ2)

Bit	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
Address	FC6H							

Hex Address: FC7

Table 166. IRQ2 Enable High Bit Register (IRQ2ENH)

Bit	7	6	5	4	3	2	1	0
Field	Reserved				C3ENH	C2ENH	C1ENH	C0ENH
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	FC7H							

Hex Address: FC8

Table 167. IRQ2 Enable Low Bit Register (IRQ2ENL)

Bit	7	6	5	4	3	2	1	0
Field	Reserved				C3ENL	C2ENL	C1ENL	C0ENL
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	FC8H							

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.