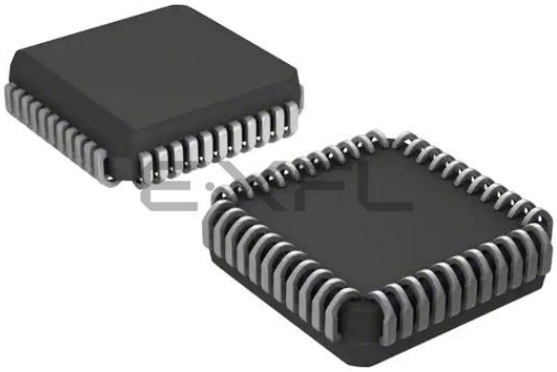


Welcome to E-XFL.COM

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	Obsolete
Core Processor	eZ8
Core Size	8-Bit
Speed	20MHz
Connectivity	I²C, IrDA, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	31
Program Memory Size	24KB (24K 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-LCC (J-Lead)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f2401vn020sc00tr

- Power-On Reset (POR)
- 3.0-3.6V operating voltage with 5V-tolerant inputs
- 0° to +70°C standard temperature and -40° to +105°C extended temperature operating ranges

Part Selection Guide

Table 1 identifies the basic features and package styles available for each device within the Z8F640x family product line.

Table 1. Z8F640x Family Part Selection Guide

Part Number	Flash (KB)	RAM (KB)	I/O	16-bit Timers with PWM	ADC Inputs	UARTs with IrDA	I^2C	SPI	40/44-pin packages	64/68-pin packages	80-pin package
Z8F1601	16	2	31	3	8	2	1	1	X		
Z8F1602	16	2	46	4	12	2	1	1		X	
Z8F2401	24	2	31	3	8	2	1	1	X		
Z8F2402	24	2	46	4	12	2	1	1		X	
Z8F3201	32	2	31	3	8	2	1	1	X		
Z8F3202	32	2	46	4	12	2	1	1		X	
Z8F4801	48	4	31	3	8	2	1	1	X		
Z8F4802	48	4	46	4	12	2	1	1		X	
Z8F4803	48	4	60	4	12	2	1	1			X
Z8F6401	64	4	31	3	8	2	1	1	X		
Z8F6402	64	4	46	4	12	2	1	1		X	
Z8F6403	64	4	60	4	12	2	1	1			X

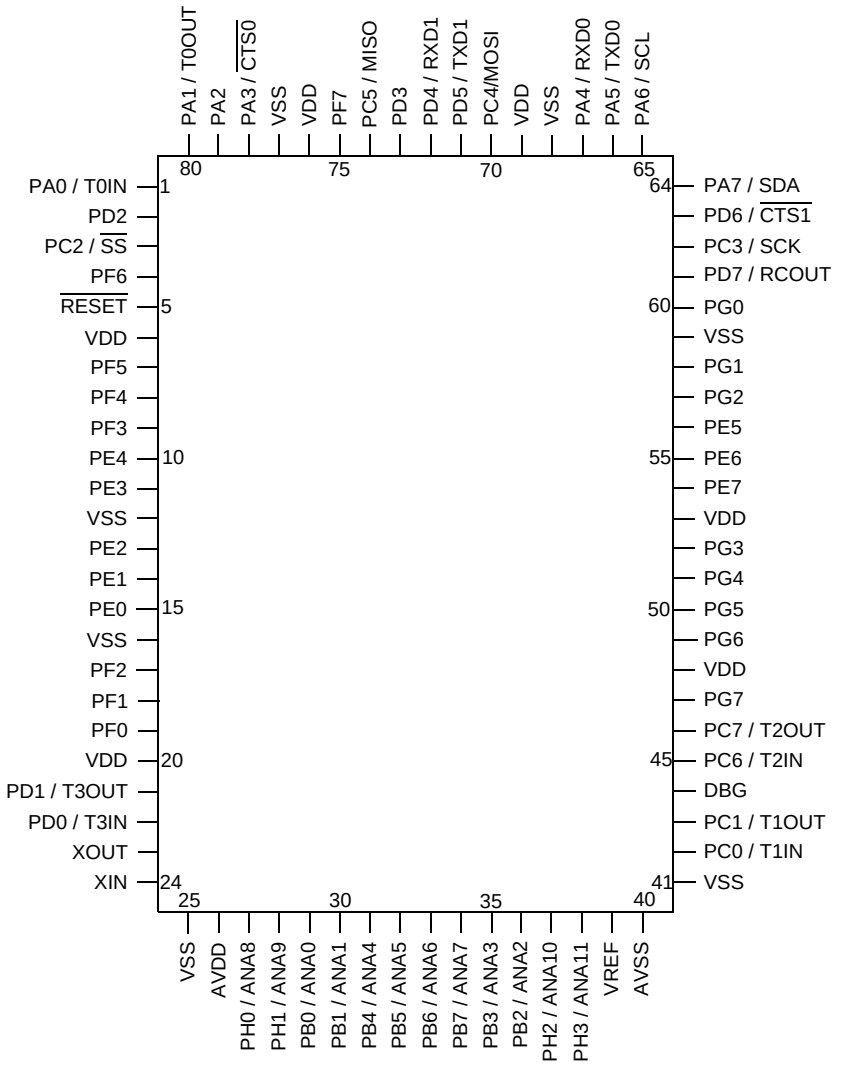


Figure 61. Z8Fxx03 in 80-Pin Quad Flat Package (QFP)

Table 2. Signal Descriptions (Continued)

Signal Mnemonic	I/O	Description
Reset		
$\overline{\text{RESET}}$	I	RESET. Generates a Reset when asserted (driven Low).
Power Supply		
VDD	I	Power Supply.
AVDD	I	Analog Power Supply.
VSS	I	Ground.
AVSS	I	Analog Ground.

Pin Characteristics

Table 3 provides detailed information on the characteristics for each pin available on the Z8F640x family products. Data in Table 3 is sorted alphabetically by the pin symbol mnemonic.

Table 3. Pin Characteristics of the Z8F640x family

Symbol Mnemonic	Direction	Reset Direction	Active Low or Active High	Tri-State Output	Internal Pull-up or Pull-down	Schmitt Trigger Input	Open Drain Output
AVSS	N/A	N/A	N/A	N/A	No	No	N/A
AVDD	N/A	N/A	N/A	N/A	No	No	N/A
DBG	I/O	I	N/A	Yes	No	Yes	Yes
VSS	N/A	N/A	N/A	N/A	No	No	N/A
PA[7:0]	I/O	I	N/A	Yes	No	Yes	Yes, Programmable
PB[7:0]	I/O	I	N/A	Yes	No	Yes	Yes, Programmable
PC[7:0]	I/O	I	N/A	Yes	No	Yes	Yes, Programmable
PD[7:0]	I/O	I	N/A	Yes	No	Yes	Yes, Programmable
PE7:0]	I/O	I	N/A	Yes	No	Yes	Yes, Programmable

x represents integer 0, 1,... to indicate multiple pins with symbol mnemonics that differ only by the integer

Power-On Reset

The Z8F640x family products contain an internal Power-On Reset (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 voltage threshold (V_{POR}), the POR Counter is enabled and counts 514 cycles of the Watch-Dog Timer oscillator. After the POR counter times out, the XTAL Counter is enabled to count a total of 16 system clock pulses. The Z8F640x family device is held in the Reset state until both the POR Counter and XTAL counter have timed out. After the device exits the Power-On Reset state, the eZ8 CPU fetches the Reset vector. Following Power-On Reset, the POR status bit in the Watch-Dog Timer Control (WDTCTL) register is set to 1.

Figure 62 illustrates Power-On Reset operation. Refer to the **Electrical Characteristics** chapter for the POR threshold voltage (V_{POR}).

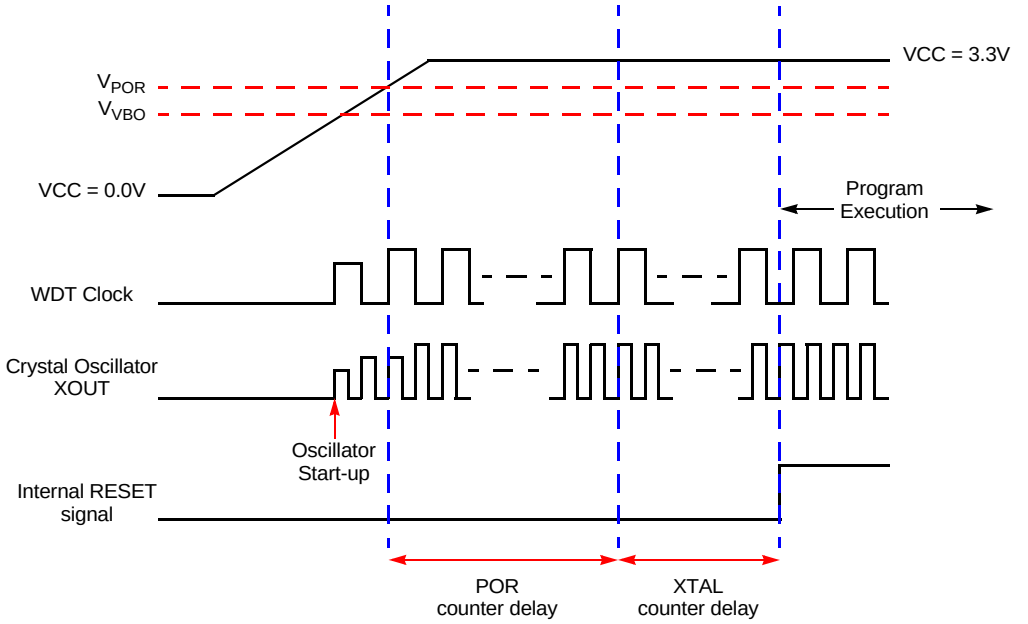


Figure 62. Power-On Reset Operation (not to scale)

Voltage Brown-Out Reset

The devices in the Z8F640x family provide low Voltage Brown-Out (VBO) protection. The VBO circuit senses when the supply voltage drops to an unsafe level (below the VBO

General-Purpose I/O

Overview

The Z8F640x family products support a maximum of seven 8-bit ports (Ports A-G) and one 4-bit port (Port H) for general-purpose input/output (I/O) operations. Each port contains control and data registers. The GPIO control registers are used to determine data direction, open-drain, output drive current and alternate pin functions. Each port pin is individually programmable.

GPIO Port Availability By Device

Not all Z8F640x family products support all 8 ports (A-H). Table 10 lists the port pins available with each device and package type.

Table 10. Port Availability by Device and Package Type

Device	Packages	Port A	Port B	Port C	Port D	Port E	Port F	Port G	Port H
Z8F1601	40-pin	[7:0]	[7:0]	[6:0]	[6:3, 1:0]	-	-	-	-
Z8F1601	44-pin	[7:0]	[7:0]	[7:0]	[6:0]				
Z8F1602	64- and 68-pin	[7:0]	[7:0]	[7:0]	[7:0]	[7:0]	[7]	[3]	[3:0]
Z8F2401	40-pin	[7:0]	[7:0]	[6:0]	[6:3, 1:0]	-	-	-	-
Z8F2401	44-pin	[7:0]	[7:0]	[7:0]	[6:0]	-	-	-	-
Z8F2402	64- and 68-pin	[7:0]	[7:0]	[7:0]	[7:0]	[7:0]	[7]	[3]	[3:0]
Z8F3201	40-pin	[7:0]	[7:0]	[6:0]	[6:3, 1:0]	-	-	-	-
Z8F3201	44-pin	[7:0]	[7:0]	[7:0]	[6:0]	-	-	-	-
Z8F3202	64- and 68-pin	[7:0]	[7:0]	[7:0]	[7:0]	[7:0]	[7]	[3]	[3:0]
Z8F4801	40-pin	[7:0]	[7:0]	[6:0]	[6:3, 1:0]	-	-	-	-
Z8F4801	44-pin	[7:0]	[7:0]	[7:0]	[6:0]	-	-	-	-
Z8F4802	64- and 68-pin	[7:0]	[7:0]	[7:0]	[7:0]	[7:0]	[7]	[3]	[3:0]
Z8F4803	80-pin	[7:0]	[7:0]	[7:0]	[7:0]	[7:0]	[7:0]	[7:0]	[3:0]
Z8F6401	40-pin	[7:0]	[7:0]	[6:0]	[6:3, 1:0]	-	-	-	-

Table 11. Port Alternate Function Mapping (Continued)

Port	Pin	Mnemonic	Alternate Function Description
Port D	PD0	T3IN	Timer 3 In (not available in 40- and 44-pin packages)
	PD1	T3OUT	Timer 3 Out (not available in 40- and 44-pin packages)
	PD2	N/A	No alternate function
	PD3	N/A	No alternate function
	PD4	RXD1 / IRRX1	UART 1 / IrDA 1 Receive Data
	PD5	TXD1 / IRTX1	UART 1 / IrDA 1 Transmit Data
	PD6	$\overline{\text{CTS1}}$	UART 1 Clear to Send
	PD7	RCOUT	Watch-Dog Timer RC Oscillator Output
Port E	PE[7:0]	N/A	No alternate functions
Port F	PF[7:0]	N/A	No alternate functions
Port G	PG[7:0]	N/A	No alternate functions
Port H	PH0	ANA8	ADC Analog Input 8
	PH1	ANA9	ADC Analog Input 9
	PH2	ANA10	ADC Analog Input 10
	PH3	ANA11	ADC Analog Input 11

GPIO Interrupts

Many of the GPIO port pins can be used as interrupt sources. Some port pins may be configured to generate an interrupt request on either the rising edge or falling edge of the pin input signal. Other port pin interrupts generate an interrupt when any edge occurs (both rising and falling). Refer to the **Interrupt Controller** chapter for more information on interrupts using the GPIO pins.

GPIO Control Register Definitions

Four registers for each Port provide access to GPIO control, input data, and output data. Table 12 lists these Port registers. Use the Port A-H Address and Control registers together to provide access to sub-registers for Port configuration and control.

– Disable the timer
– Configure the timer for PWM mode.
– Set the prescale value.
– Set the initial logic level (High or Low) and PWM High/Low transition for the Timer Output alternate function.

2. Write to the Timer High and Low Byte registers to set the starting count value (typically 0001H). This only affects the first pass in PWM mode. After the first timer reset in PWM mode, counting always begins at the reset value of 0001H.
3. Write to the PWM High and Low Byte registers to set the PWM value.
4. Write to the Timer Reload High and Low Byte registers to set the Reload value (PWM period). The Reload value must be greater than the PWM value.
5. If desired, enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers.
6. Configure the associated GPIO port pin for the Timer Output alternate function.
7. Write to the Timer Control register to enable the timer and initiate counting.

The PWM period is given by the following equation:

$$\text{PWM Period (s)} = \frac{\text{Reload Value} \times \text{Prescale}}{\text{System Clock Frequency (Hz)}}$$

If an initial starting value other than 0001H is loaded into the Timer High and Low Byte registers, the One-Shot mode equation must be used to determine the first PWM time-out period.

If TPOL is set to 0, the ratio of the PWM output High time to the total period is given by:

$$\text{PWM Output High Time Ratio (\%)} = \frac{\text{Reload Value} - \text{PWM Value}}{\text{Reload Value}} \times 100$$

If TPOL is set to 1, the ratio of the PWM output High time to the total period is given by:

$$\text{PWM Output High Time Ratio (\%)} = \frac{\text{PWM Value}}{\text{Reload Value}} \times 100$$

Capture Mode

In Capture mode, the current timer count value is recorded when the desired external Timer Input transition occurs. The Capture count value is written to the Timer PWM High and Low Byte Registers. The timer input is the system clock. The TPOL bit in the Timer Control register determines if the Capture occurs on a rising edge or a falling edge of the

– 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. This 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. If desired, enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers.
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 desired 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 desired 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 the Capture event occurs, an interrupt is generated, the count value in the Timer High and Low Byte registers is reset to 0001H, and counting resumes.

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 steps for configuring a timer for Capture/Compare mode and initiating the count are as follows:

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. If desired, enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers.

6. Write to the UART Control 0 register to:
   – Set the receive enable bit (REN) to enable the UART for data reception
   – Enable parity, if desired, and select either even or odd parity.

The UART and DMA are now configured for data reception and automatic data transfer to the Register File. When a valid data byte is received by the UART the following occurs:

7. The UART notifies the DMA Controller that a data byte is available in the UART Receive Data register.
8. The DMA Controller requests control of the system bus from the eZ8 CPU.
9. The eZ8 CPU acknowledges the bus request.
10. The DMA Controller transfers the data from the UART Receive Data register to another location in RAM and then return bus control back to the eZ8 CPU.

The UART and DMA can continue to transfer incoming data bytes without eZ8 CPU intervention. When a UART error is detected, the UART Receiver interrupt is generated. The associated interrupt service routine (ISR) should perform the following:

11. Check the UART Status 0 register to determine the source of the UART error or break condition and then respond appropriately.

Multiprocessor (9-bit) mode

The UART has a Multiprocessor mode that uses an extra (9th) bit for selective communication when a number of processors share a common UART bus. In Multiprocessor (9-bit) mode (also referred to as 9-Bit mode), the multiprocessor bit (MP) is transmitted immediately following the 8-bits of data and immediately preceding the STOP bit(s) as illustrated in Figure 70. The character format is:

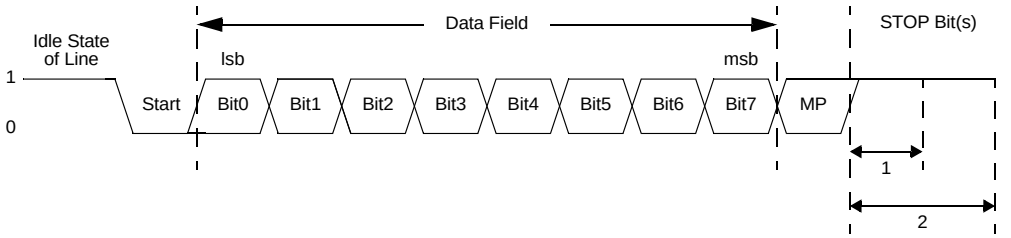


Figure 70. UART Asynchronous Multiprocessor (9-bit) Mode Data Format

In Multiprocessor (9-bit) mode, parity is not an option as the Parity bit location (9th bit) becomes the Multiprocessor control bit. The UART Control 1 and Status 1 registers provide multiprocessor (9-bit) mode control and status information.

1. Software writes the I2C Data register with a 7-bit slave address followed by a 1 (read).
2. Software asserts the START bit of the I2C Control register.
3. Software asserts the NAK bit of the I2C Control register so that after the first byte of data has been read by the I2C Controller, a Not Acknowledge is sent to the I2C slave.
4. The I2C Controller sends the START condition.
5. The I2C Controller sends the address and read bit by the SDA signal.
6. The I2C slave sends an Acknowledge by pulling the SDA signal Low during the next high period of SCL.
7. The I2C Controller reads the first byte of data from the I2C slave.
8. The I2C Controller asserts the Receive interrupt.
9. Software responds by reading the I2C Data register.
10. The I2C Controller sends a NAK to the I2C slave.
11. A NAK interrupt is generated by the I2C Controller.
12. Software responds by setting the STOP bit of the I2C Control register.
13. A STOP condition is sent to the I2C slave.

Reading a Transaction with a 10-Bit Address

Figure 82 illustrates the receive format for a 10-bit addressed slave. The shaded regions indicate data transferred from the I2C Controller to slaves and unshaded regions indicate data transferred from the slaves to the I2C Controller.

S	Slave Address 1st 7 bits	W=0	A	Slave address 2nd Byte	A	S	Slave Address 1st 7 bits	R=1	A	Data	A	Data	$\overline{A}$	P

Figure 82. Receive Data Format for a 10-Bit Addressed Slave

The first seven bits transmitted in the first byte are 11110XX. The two bits XX are the two most-significant bits of the 10-bit address. The lowest bit of the first byte transferred is the write signal.

The data transfer format for a receive operation on a 10-bit addressed slave is as follows:

1. Software writes an address 11110B followed by the two address bits and a 0 (write).
2. Software asserts the START bit of the I2C Control register.
3. The I2C Controller sends the Start condition.

I²C Control Register Definitions

I²C Data Register

The I²C Data register holds the data that is to be loaded into the I²C Shift register during a write to a slave. This register also holds data that is loaded from the I²C Shift register during a read from a slave. The I²C Shift is not accessible in the Register File address space, but is used only to buffer incoming and outgoing data.

Table 66. I²C Data Register (I2CDATA)

BITS	7	6	5	4	3	2	1	0
FIELD	DATA							
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	F50H							

I²C Status Register

The Read-only I²C Status register indicates the status of the I²C Controller.

Table 67. I²C Status Register (I2CSTAT)

BITS	7	6	5	4	3	2	1	0
FIELD	TDRE	RDRF	ACK	10B	RD	TAS	DSS	NCKI
RESET	1	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R
ADDR	F51H							

TDRE—Transmit Data Register Empty
When the I²C Controller is enabled, this bit is 1 when the I²C Data register is empty. When active, this bit causes the I²C Controller to generate an interrupt, except when the I²C Controller is shifting in data during the reception of a byte or when shifting an address and the RD bit is set. This bit and the interrupt are cleared by writing to the I²CD register.

RDRF—Receive Data Register Full
This bit is set active high when the I²C Controller is enabled and the I²C Controller has

DMA Status Register

The DMA Status register indicates the DMA channel that generated the interrupt and the ADC Analog Input that is currently undergoing conversion. Reads from this register reset the Interrupt Request Indicator bits (IRQA, IRQ1, and IRQ0) to 0. Therefore, software interrupt service routines that read this register must process all three interrupt sources from the DMA.

Table 79. DMA_ADC Status Register (DMAA_STAT)

BITS	7	6	5	4	3	2	1	0
FIELD	CADC[3:0]				Reserved	IRQA	IRQ1	IRQ0
RESET	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R
ADDR	FBFH							

CADC[3:0]—Current ADC Analog Input
This field identifies the Analog Input that the ADC is currently converting.

Reserved
This bit is reserved and must be 0.

IRQA—DMA_ADC Interrupt Request Indicator
This bit is automatically reset to 0 each time a read from this register occurs.
0 = DMA_ADC is not the source of the interrupt from the DMA Controller.
1 = DMA_ADC completed transfer of data from the last ADC Analog Input and generated an interrupt.

IRQ1—DMA1 Interrupt Request Indicator
This bit is automatically reset to 0 each time a read from this register occurs.
0 = DMA1 is not the source of the interrupt from the DMA Controller.
1 = DMA1 completed transfer of data to/from the End Address and generated an interrupt.

IRQ0—DMA0 Interrupt Request Indicator
This bit is automatically reset to 0 each time a read from this register occurs.
0 = DMA0 is not the source of the interrupt from the DMA Controller.
1 = DMA0 completed transfer of data to/from the End Address and generated an interrupt.

1. Enable the desired analog inputs by configuring the general-purpose I/O pins for alternate function. This configuration disables the digital input and output drivers.
2. Write to the ADC Control register to configure the ADC and begin the conversion. The bit fields in the ADC Control register can be written simultaneously:
   – Write to ANAIN[3:0] to select one of the 12 analog input sources.
   – Clear CONT to 0 to select a single-shot conversion.
   – Write to VREF to enable or disable the internal voltage reference generator.
   – Set CEN to 1 to start the conversion.
3. CEN remains 1 while the conversion is in progress. A single-shot conversion requires 5129 system clock cycles to complete. If a single-shot conversion is requested from an ADC powered-down state, the ADC uses 40 additional clock cycles to power-up before beginning the 5129 cycle conversion.
4. When the conversion is complete, the ADC control logic performs the following operations:
   – 10-bit data result written to {ADCD_H[7:0], ADCD_L[7:6]}.
   – CEN resets to 0 to indicate the conversion is complete.
   – An interrupt request is sent to the Interrupt Controller.
5. If the ADC remains idle for 160 consecutive system clock cycles, it is automatically powered-down.

Continuous Conversion

When configured for continuous conversion, the ADC continuously performs an analog-to-digital conversion on the selected analog input. Each new data value over-writes the previous value stored in the ADC Data registers. An interrupt is generated only at the end of the first conversion after enabling.

Caution: In Continuous mode, users must be aware that ADC updates are limited by the input signal bandwidth of the ADC and the latency of the ADC and its digital filter. Step changes at the input are not seen at the next output from the ADC. The response of the ADC (in all modes) is limited by the input signal bandwidth and the latency.

The steps for setting up the ADC and initiating continuous conversion are as follows:

1. Enable the desired analog input by configuring the general-purpose I/O pins for alternate function. This disables the digital input and output driver.
2. Write to the ADC Control register to configure the ADC for continuous conversion. The bit fields in the ADC Control register may be written simultaneously:
   – Write to ANAIN[3:0] to select one of the 12 analog input sources.

Flash Operation Timing Using the Flash Frequency Registers

Before performing either a program or erase operation on the Flash memory, the user must first configure the Flash Frequency High and Low Byte registers. The Flash Frequency registers allow programming and erasure of the Flash with system clock frequencies ranging from 32KHz (32768Hz) through 20MHz.

The Flash Frequency High and Low Byte registers combine to form a 16-bit value, FFREQ, to control timing for Flash program and erase operations. The 16-bit binary Flash Frequency value must contain the system clock frequency (in kHz). This value is calculated using the following equation:.

$$\text{FFREQ[15:0]} = \frac{\text{System Clock Frequency (Hz)}}{1000}$$

Caution: Flash programming and erasure are not supported for system clock frequencies below 32KHz (32768Hz) or above 20MHz. The Flash Frequency High and Low Byte registers must be loaded with the correct value to insure proper operation of the Z8F640x family device.

Flash Code Protection Against External Access

The user code contained within the Z8F640x family device's Flash memory can be protected against external access via the On-Chip Debugger. Programming the RP Option Bit prevents reading of the user code through the On-Chip Debugger. Refer to the **Option Bits** chapter and the **On-Chip Debugger** chapter for more information.

Flash Code Protection Against Accidental Program and Erasure

The Z8F640x family device provides several levels of protection against accidental program and erasure of the Flash memory contents. This protection is provided by a combination of the Option bits and the locking mechanism of the Flash Controller.

- **Read Data Memory (0DH)**—The Read Data Memory command reads from Data Memory. This command is equivalent to the LDE and LDEI instructions. Data can be read 1-65536 bytes at a time (65536 bytes can be read by setting size to zero). If the Z8F640x family 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 CRC (cyclic redundancy check) of Program Memory using the 16-bit CRC-CCITT polynomial. If the Z8F640x family device is not in Debug mode, this command returns FFFFH for the CRC value. Unlike most other OCD Read commands, there is a delay from issuing of the command until the OCD returns the data. The OCD reads the Program Memory, calculates the CRC value, and returns the result. The delay is a function of the Program Memory size and is approximately equal to the system clock period multiplied by the number of bytes in the Program Memory.

```
DBG <-- 0EH
DBG --> CRC[15:8]
DBG --> CRC[7:0]
```

- **Step Instruction (10H)**—The Step Instruction command steps one assembly instruction at the current Program Counter (PC) location. If the Z8F640x family device is not in Debug mode or the Read Protect Option Bit is enabled, the OCD ignores this command.

```
DBG <-- 10H
```

- **Stuff Instruction (11H)**—The Stuff Instruction command steps one assembly instruction and allows specification of the first byte of the instruction. The remaining 0-4 bytes of the instruction are read from Program Memory. This command is useful for stepping over instructions where the first byte of the instruction has been overwritten by a Breakpoint. If the Z8F640x family device is not in Debug mode or the Read Protect Option Bit is enabled, the OCD ignores this command.

```
DBG <-- 11H
DBG <-- opcode[7:0]
```

- **Execute Instruction (12H)**—The Execute Instruction command allows sending an entire instruction to be executed to the eZ8 CPU. This command can also step over Breakpoints. The number of bytes to send for the instruction depends on the opcode. If the Z8F640x family device is not in Debug mode or the Read Protect Option Bit is enabled, this command reads and discards one byte.

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

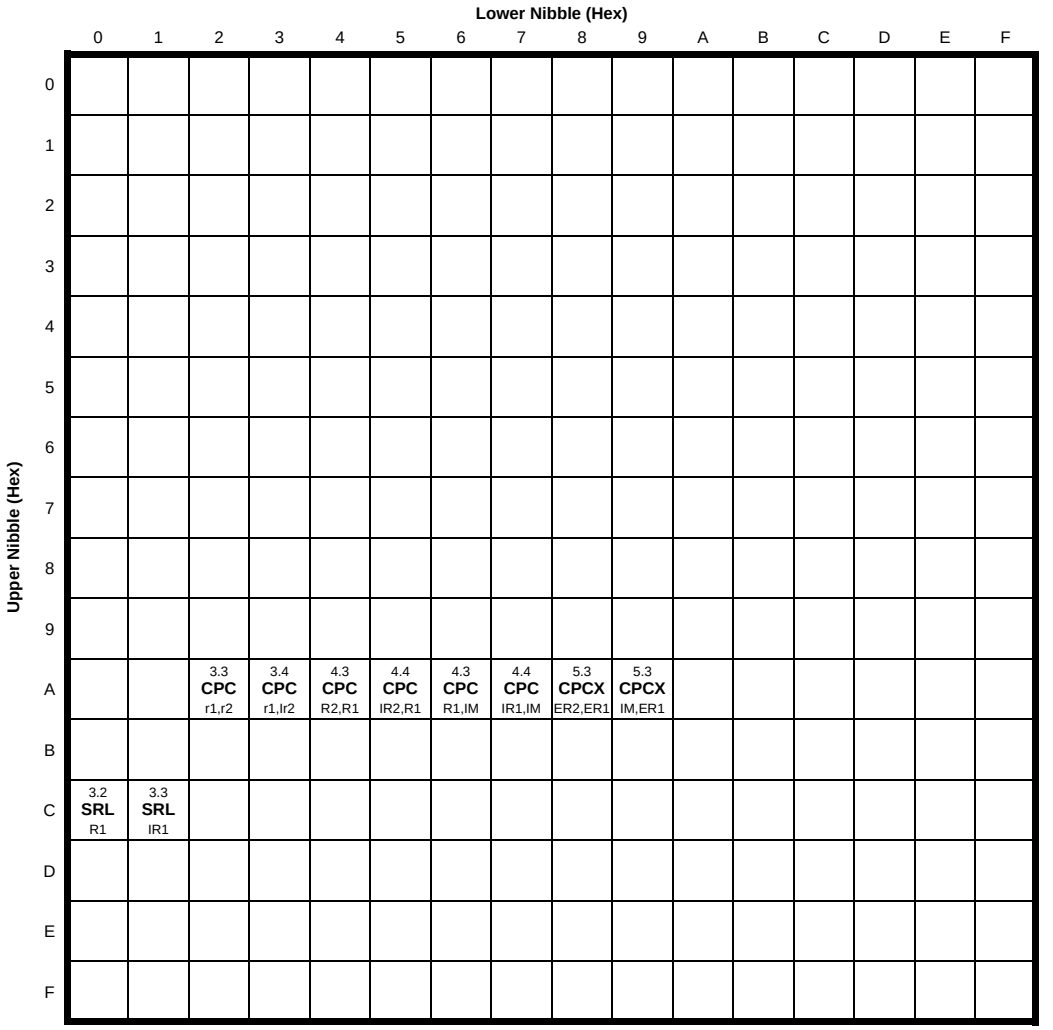
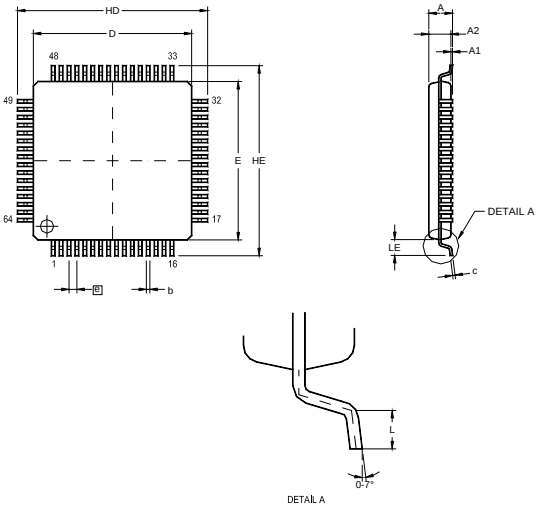


Figure 102. Second Opcode Map after 1FH

Figure 105 illustrates the 64-pin LQFP (low-profile quad flat package) available for the Z8F1602, Z8F2402, Z8F3202, Z8F4802, and Z8F6402 devices.



SYMBOL	MILLIMETER		INCH	
	MIN	MAX	MIN	MAX
A	1.40	1.60	0.055	0.063
A1	0.05	0.15	0.002	0.006
A2	1.35	1.45	0.053	0.057
b	0.17	0.27	0.007	0.011
c	0.09	0.20	0.004	0.008
HD	11.75	12.25	0.463	0.482
D	9.90	10.10	0.390	0.398
HE	11.75	12.25	0.463	0.482
E	9.90	10.10	0.390	0.398
e	0.50 BSC		0.0197 BSC	
L	0.45	0.75	0.018	0.030
LE	1.00 REF		0.039 REF	

1. CONTROLLING DIMENSIONS : mm
2. MAX. COPLANARITY : 0.10mm / 0.004"

Figure 106. 64-Lead Low-Profile Quad Flat Package (LQFP)

Customer Feedback Form

The Z8 Encore!™ Product Specification

If you experience any problems while operating this product, or if you note any inaccuracies while reading this Product Specification, please copy and complete this form, then mail or fax it to ZiLOG (see *Return Information*, below). We also welcome your suggestions!

Customer Information

Name	Country
Company	Phone
Address	Fax
City/State/Zip	E-Mail

Product Information

Part #, Serial #, Board Fab #, or Rev. #
Software Version
Document Number
Host Computer Description/Type

Return Information

ZiLOG
532 Race Street
San Jose, CA 95126
Fax: (408) 558-8536
Email: tools@zilog.com