
**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 | I²C, IrDA, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, DMA, POR, PWM, WDT |
| Number of I/O | 60 |
| Program Memory Size | 64KB (64K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 4K x 8 |
| Voltage - Supply (Vcc/Vdd) | 3V ~ 3.6V |
| Data Converters | A/D 12x10b |
| Oscillator Type | Internal |
| Operating Temperature | 0°C ~ 70°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 80-BQFP |
| Supplier Device Package | - |
| Purchase URL | https://www.e-xfl.com/product-detail/zilog/z8f6423ft020sg |

# List of Figures

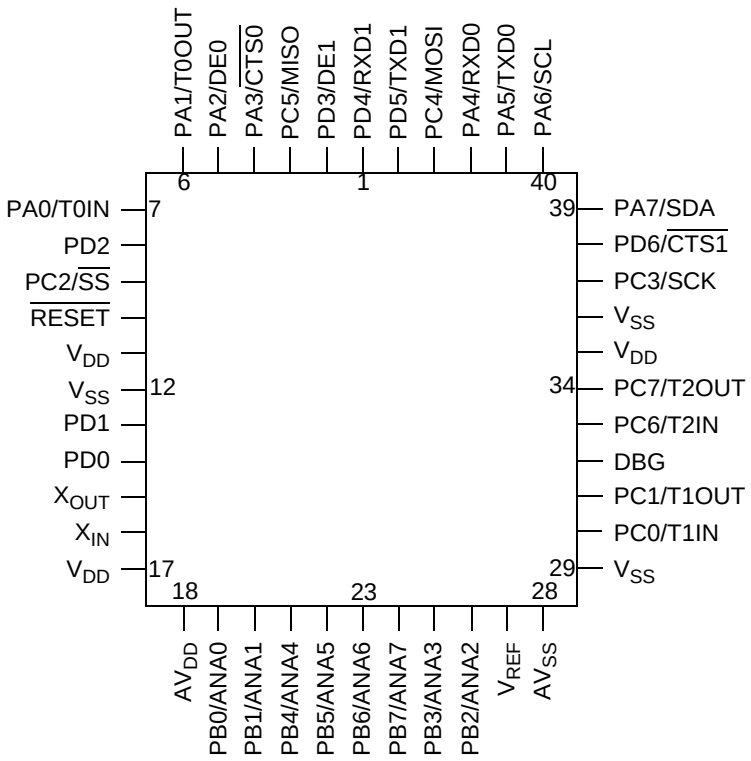


**Figure 3. Z8 Encore! XP F64xx Series in 44-Pin Plastic Leaded Chip Carrier (PLCC)**

**Note:** Timer 3 is not available in the 44-pin PLCC package.

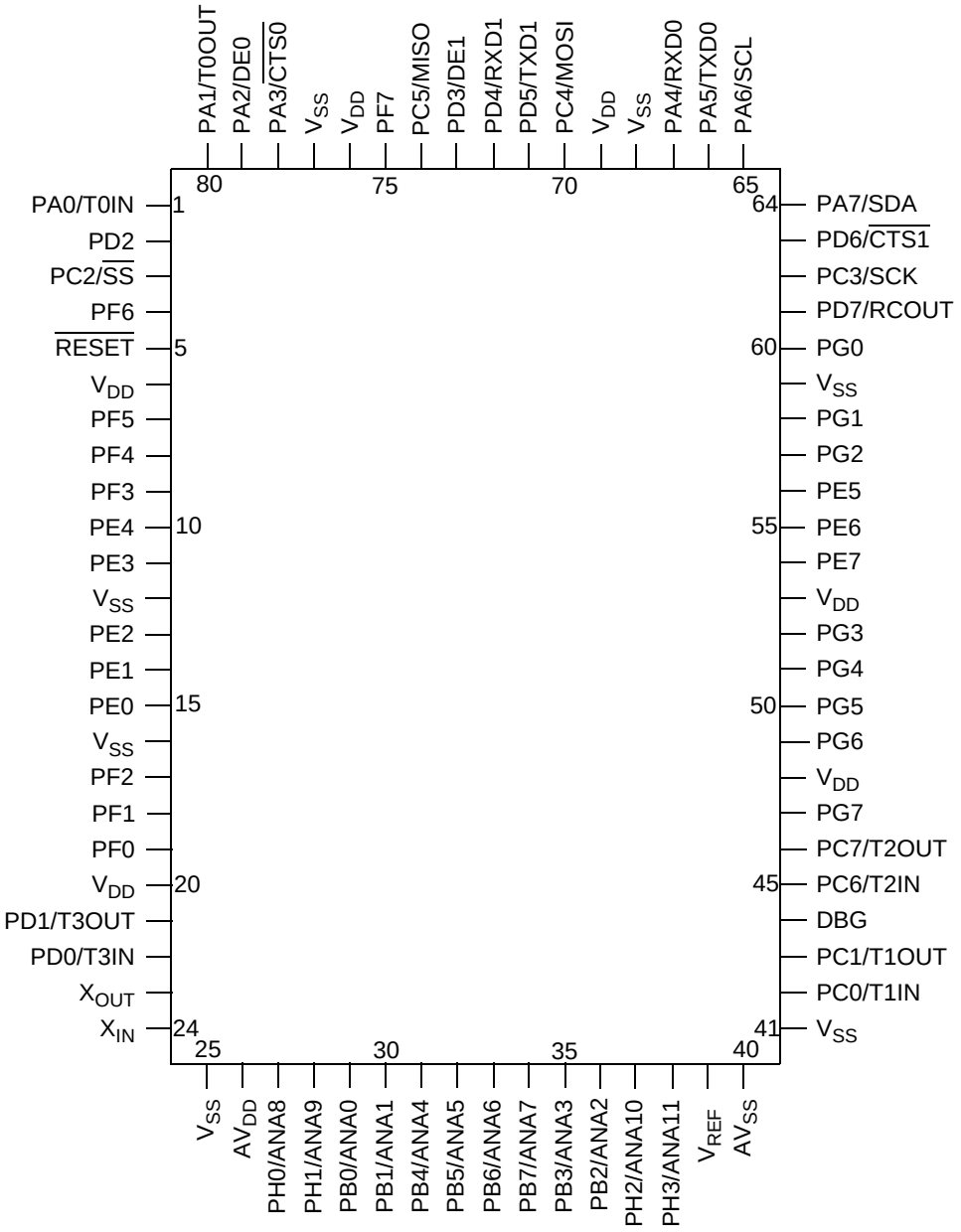


**Figure 7. Z8 Encore! XP F64xx Series in 80-Pin Quad Flat Package (QFP)**

**Table 35. IRQ2 Enable Low Bit Register (IRQ2ENL)**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | T3ENL | U1RENL | U1TENL | DMAENL | C3ENL | C2ENL | C1ENL | C0ENL |
| RESET | 0 | | | | | | | |
| R/W | R/W | | | | | | | |
| Address | FC8H | | | | | | | |

| Bit | Description |
|---|---|
| [7] T3ENL | **Timer 3 Interrupt Request Enable Low Bit** |
| [6] U1RENL | **UART 1 Receive Interrupt Request Enable Low Bit** |
| [5] U1TENL | **UART 1 Transmit Interrupt Request Enable Low Bit** |
| [4] DMAENL | **DMA Interrupt Request Enable Low Bit** |
| [3] C3ENL | **Port C3 Interrupt Request Enable Low Bit** |
| [2] C2ENL | **Port C2 Interrupt Request Enable Low Bit** |
| [1] C1ENL | **Port C1 Interrupt Request Enable Low Bit** |
| [0] C0ENL | **Port C0 Interrupt Request Enable Low Bit** |

enabled, the timer output pin changes state (from Low to High or from High to Low) at timer reload.

Observe the following procedure for configuring a timer for COUNTER Mode and initiating the count:

1. Write to the Timer Control 1 Register to:
   - Disable the timer.
   - Configure the timer for COUNTER Mode.
   - Select either the rising edge or falling edge of the timer input signal for the count. This also sets the initial logic level (High or Low) for the timer output alternate function. However, the timer output function does not have to be enabled.
2. Write to the Timer High and Low Byte registers to set the starting count value. This only affects the first pass in COUNTER Mode. After the first timer reload in COUNTER Mode, counting always begins at the reset value of 0001H. Generally, in COUNTER Mode the Timer High and Low Byte registers must be written with the value 0001H.
3. Write to the Timer Reload High and Low Byte registers to set the reload value.
4. If appropriate, 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. If using the timer output function, configure the associated GPIO port pin for the timer output alternate function.
7. Write to the Timer Control 1 Register to enable the timer.

In COUNTER Mode, the number of timer input transitions since the timer start is calculated using the following equation:

COUNTER Mode Timer Input Transitions = Current Count Value – Start Value

### PWM Mode

In PWM Mode, the timer outputs a Pulse-Width Modulator (PWM) output signal through a GPIO port pin. The timer input is the system clock. The timer first counts up to the 16-bit PWM match value stored in the Timer PWM High and Low Byte registers. When the timer count value matches the PWM value, the timer output toggles. The timer continues counting until it reaches the reload value stored in the Timer Reload High and Low Byte registers. 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.

## Timer Reload High and Low Byte Registers

The Timer 0–3 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 0–3 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 | | | | | | | |

**Table 42. 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 | | | | | | | |

| Bit | Description |
|---|---|
| [7:0] TRH, TRL | **Timer Reload Register High and Low**<br>These two bytes form the 16-bit reload value, {TRH[7:0], TRL[7:0]}. This value sets the maximum count value which initiates a timer reload to 0001H. In COMPARE Mode, these two bytes form the 16-bit compare value. |

| Bit | Description |
|---|---|
| [7]<br>TEN | **Timer Enable**<br>0 = Timer is disabled.<br>1 = Timer enabled to count. |
| [6]<br>TPOL | **Timer Input/Output Polarity**<br>Operation of this bit is a function of the current operating mode of the timer.<br>**ONE-SHOT Mode**<br>When the timer is disabled, the timer output signal is set to the value of this bit. When the timer is enabled, the timer output signal is complemented upon timer reload.<br>**CONTINUOUS Mode**<br>When the timer is disabled, the timer output signal is set to the value of this bit. When the timer is enabled, the timer output signal is complemented upon timer reload.<br>**COUNTER Mode**<br>When the timer is disabled, the timer output signal is set to the value of this bit. When the timer is enabled, the timer output signal is complemented upon timer reload.<br>0 = Count occurs on the rising edge of the timer input signal.<br>1 = Count occurs on the falling edge of the timer input signal.<br>**PWM Mode**<br>0 = timer output is forced Low (0) when the timer is disabled. When enabled, the timer output is forced High (1) upon PWM count match and forced Low (0) upon reload.<br>1 = timer output is forced High (1) when the timer is disabled. When enabled, the timer output is forced Low (0) upon PWM count match and forced High (1) upon reload.<br>**CAPTURE Mode**<br>0 = Count is captured on the rising edge of the timer input signal.<br>1 = Count is captured on the falling edge of the timer input signal.<br>**COMPARE Mode**<br>When the timer is disabled, the timer output signal is set to the value of this bit. When the timer is enabled, the timer output signal is complemented upon timer reload.<br>**GATED Mode**<br>0 = Timer counts when the timer input signal is High (1) and interrupts are generated on the falling edge of the timer input.<br>1 = Timer counts when the timer input signal is Low (0) and interrupts are generated on the rising edge of the timer input.<br>**CAPTURE/COMPARE Mode**<br>0 = Counting is started on the first rising edge of the timer input signal. The current count is captured on subsequent rising edges of the timer input signal.<br>1 = Counting is started on the first falling edge of the timer input signal. The current count is captured on subsequent falling edges of the timer input signal.<br><br>**Caution:** When the timer output alternate function TxOUT on a GPIO port pin is enabled, TxOUT will change to whatever state the TPOL bit is in. The timer does not need to be enabled for that to happen. Also, the Port Data Direction Subregister is not needed to be set to output on TxOUT. Changing the TPOL bit with the timer enabled and running does not immediately change the TxOUT. |

### 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 TXD*x* pin. The write-only UART Transmit Data Register shares a Register File address with the read-only UART Receive Data Register.

## Slave Operation

The SPI block is configured for SLAVE Mode operation by setting the SPIEN bit to 1 and the MMEN bit to 0 in the SPICTL Register and setting the SSIO bit to 0 in the SPIMODE Register. The IRQE, PHASE, CLKPOL, WOR bits in the SPICTL Register and the NUMBITS field in the SPIMODE Register must be set to be consistent with the other SPI devices. The STR bit in the SPICTL Register may be used if appropriate to force a *start-up* interrupt. The BIRQ bit in the SPICTL Register and the SSV bit in the SPIMODE Register are not used in SLAVE Mode. The SPI baud rate generator is not used in SLAVE Mode so the SPIBRH and SPIBRL registers need not be initialized.

If the slave has data to send to the master, the data must be written to the SPIDAT Register before the transaction starts (first edge of SCK when $\overline{SS}$ is asserted). If the SPIDAT Register is not written prior to the slave transaction, the MISO pin outputs whatever value is currently in the SPIDAT Register.

Due to the delay resulting from synchronization of the SPI input signals to the internal system clock, the maximum SPICLK baud rate that can be supported in SLAVE Mode is the system clock frequency ($X_{IN}$) divided by 8. This rate is controlled by the SPI master.

## Error Detection

The SPI contains error detection logic to support SPI communication protocols and recognize when communication errors have occurred. The SPI Status Register indicates when a data transmission error has been detected.

### Overrun (Write Collision)

An overrun error (write collision) indicates that a write to the SPI Data Register was attempted while a data transfer was in progress (in either MASTER or SLAVE modes). An overrun sets the OVR bit in the SPI Status Register to 1. Writing a 1 to OVR clears this error flag. The data register is not altered when a write occurs while data transfer is in progress.

### Mode Fault (Multimaster Collision)

A mode fault indicates when more than one Master is trying to communicate at the same time (a multimaster collision). The mode fault is detected when the enabled Master's $\overline{SS}$ pin is asserted. A mode fault sets the COL bit in the SPI Status Register to 1. Writing a 1 to COL clears this error flag.

### Slave Mode Abort

In the SLAVE Mode of operation, if the $\overline{SS}$ pin deasserts before all bits in a character have been transferred, the transaction is aborted. When this condition occurs, the ABT bit is set in the SPISTAT Register as well as the IRQ bit (indicating the transaction is complete).

| Bit | Description (Continued) |
|---|---|
| [4]<br>PHASE | **Phase Select**<br>Sets the phase relationship of the data to the clock. For more information about operation of the PHASE bit, see the SPI Clock Phase and Polarity Control section on page 116. |
| [3]<br>CLKPOL | **Clock Polarity**<br>0 = SCK idles Low (0).<br>1 = SCK idle High (1). |
| [2]<br>WOR | **Wire-OR (OPEN-DRAIN) Mode Enabled**<br>0 = SPI signal pins not configured for open-drain.<br>1 = All four SPI signal pins (SCK, SS, MISO, MOSI) configured for open-drain function. This setting is typically used for multimaster and/or multislave configurations. |
| [1]<br>MMEN | **SPI Master Mode Enable**<br>0 = SPI configured in SLAVE Mode.<br>1 = SPI configured in MASTER Mode. |
| [0]<br>SPIEN | **SPI Enable**<br>0 = SPI disabled.<br>1 = SPI enabled. |

## 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 = 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]<br>IRQ | **Interrupt Request**<br>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.<br>0 = No SPI interrupt request pending.<br>1 = SPI interrupt request is pending. |
| [6]<br>OVR | **Overrun**<br>0 = An overrun error has not occurred.<br>1 = An overrun error has been detected. |

| Bit | Description (Continued) |
|---|---|
| [1]<br>FLUSH | **Flush Data**<br>Setting this bit to 1 clears the I²C Data Register and sets the TDRE bit to 1. This bit allows flushing of the I²C Data Register when a Not Acknowledge interrupt is received after the data has been sent to the I²C Data Register. Reading this bit always returns 0. |
| [0]<br>FILTEN | **I²C Signal Filter Enable**<br>This bit enables low-pass digital filters on the SDA and SCL input signals. These filters reject any input pulse with periods less than a full system clock cycle. The filters introduce a 3-system clock cycle latency on the inputs.<br>1 = low-pass filters are enabled.<br>0 = low-pass filters are disabled. |

## I²C Baud Rate High and Low Byte Registers

The I²C Baud Rate High and Low Byte registers, shown in Tables 74 and 75, combine to form a 16-bit reload value, BRG[15:0], for the I²C Baud Rate Generator.

When the I²C 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 I²C by clearing the IEN bit in the I²C Control Register to 0.
2. Load the appropriate 16-bit count value into the I²C Baud Rate High and Low Byte registers.
3. Enable the Baud Rate Generator timer function and associated interrupt by setting the BIRQ bit in the I²C Control Register to 1.

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

Interrupt Interval (s) = System Clock Period (s) × BRG[15:0]

## ADC Data Low Bits Register

The ADC Data Low Bits Register, Table 89, contains the lower two bits of the conversion value. The data in the ADC Data Low Bits Register is latched each time the ADC Data High Byte Register is read. Reading this register always returns the lower two bits of the conversion last read into the ADC High Byte Register. Access to the ADC Data Low Bits Register is read-only. The full 10-bit ADC result is provided by {ADCD_H[7:0], ADCD_L[7:6]}.

**Table 89. ADC Data Low Bits Register (ADCD_L)**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | ADCD_L | | Reserved | | | | | |
| RESET | X | | | | | | | |
| R/W | R | | | | | | | |
| Address | F73H | | | | | | | |

| Bit | Description |
|---|---|
| [7:6] ADCD_L | **ADC Data Low Bits**<br>These are the least significant two bits of the 10-bit ADC output. These bits are undefined after a Reset. |
| [5:0] | **Reserved**<br>These bits are reserved and are always undefined. |

If the OCD receives a serial break (nine or more continuous bits Low) the Autobaud Detector/Generator resets. The Autobaud Detector/Generator can then be reconfigured by sending 80H.

## OCD Serial Errors

The On-Chip Debugger can detect any of the following error conditions on the DBG pin:

- Serial break (a minimum of nine continuous bits Low)
- Framing error (received stop bit is Low)
- Transmit collision (OCD and host simultaneous transmission detected by the OCD)

When the OCD detects one of these errors, it aborts any command currently in progress, transmits a serial break 4096 system clock cycles long back to the host, and resets the Autobaud Detector/Generator. A framing error or transmit collision may be caused by the host sending a serial break to the OCD. Because of the open-drain nature of the interface, returning a serial break back to the host only extends the length of the serial break if the host releases the serial break early.

The host transmits a serial break on the DBG pin when first connecting to the Z8 Encore! XP F64xx Series devices or when recovering from an error. A serial break from the host resets the Autobaud Generator/Detector but does not reset the OCD Control Register. A serial break leaves the device in DEBUG Mode if that is the current mode. The OCD is held in Reset until the end of the serial break when the DBG pin returns High. Because of the open-drain nature of the DBG pin, the host can send a serial break to the OCD even if the OCD is transmitting a character.

## Breakpoints

Execution breakpoints are generated using the BRK instruction (op code 00H). When the eZ8 CPU decodes a BRK instruction, it signals the On-Chip Debugger. If breakpoints are enabled, the OCD idles the eZ8 CPU and enters DEBUG Mode. If breakpoints are not enabled, the OCD ignores the BRK signal and the BRK instruction operates as an NOP.

If breakpoints are enabled, the OCD can be configured to automatically enter DEBUG Mode, or to loop on the break instruction. If the OCD is configured to loop on the BRK instruction, then the CPU is still enabled to service DMA and interrupt requests.

The loop on BRK instruction can be used to service interrupts in the background. For interrupts to be serviced in the background, there cannot be any breakpoints in the interrupt service routine. Otherwise, the CPU stops on the breakpoint in the interrupt routine. For interrupts to be serviced in the background, interrupts must also be enabled. Debugging software should not automatically enable interrupts when using this feature, since

Figure 45 displays the typical current consumption in HALT Mode while operating at 25ºC plotted opposite the system clock frequency. All GPIO pins are configured as outputs and driven High.

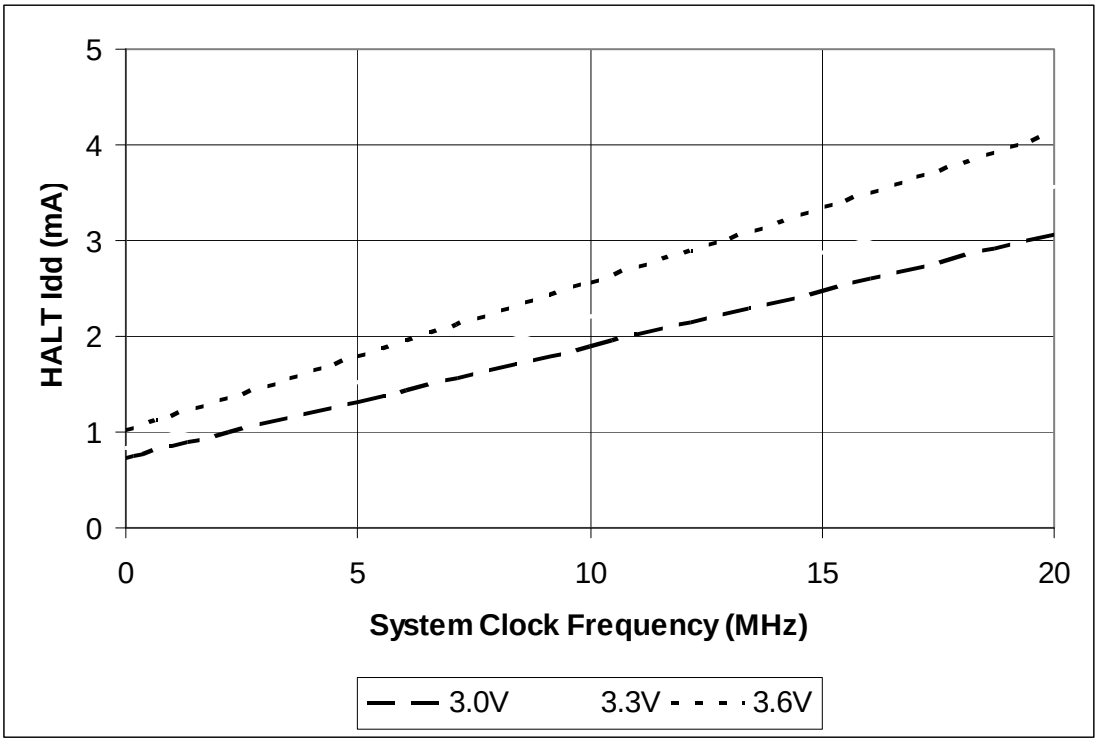


**Figure 45. Typical HALT Mode $I_{DD}$ vs. System Clock Frequency**

Figure 47 displays the maximum current consumption in STOP Mode with the VBO and Watchdog Timer enabled plotted opposite the power supply voltage. All GPIO pins are configured as outputs and driven High.

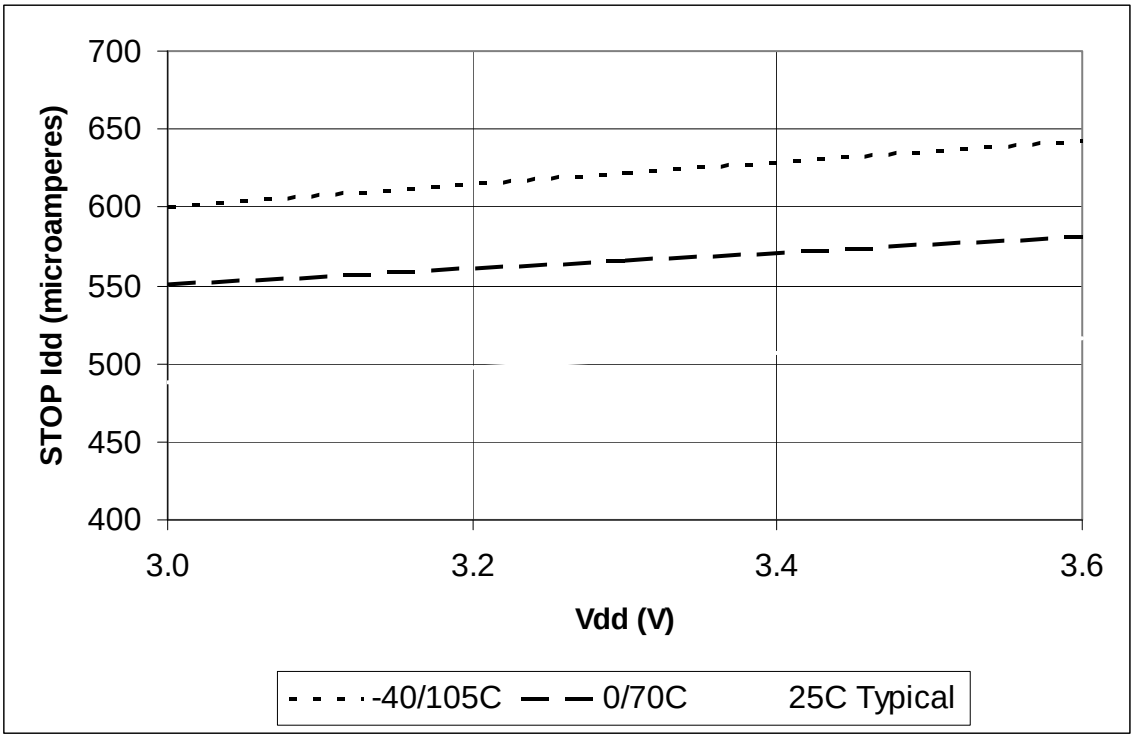


**Figure 47. Maximum STOP Mode $I_{DD}$ with VBO Enabled vs. Power Supply Voltage**

**Table 113. Analog-to-Digital Converter Electrical Characteristics and Timing (Continued)**

| | | $V_{DD}$ = 3.0V–3.6V<br>$T_A$ = –40°C to 125°C | | | | |
|---|---|---|---|---|---|---|
| **Symbol** | **Parameter** | **Minimum** | **Typical** | **Maximum** | **Units** | **Conditions** |
| $I_{REF}$ | Current draw into $V_{REF}$ pin when driving with external source. | | 25.0 | 40.0 | µA | |

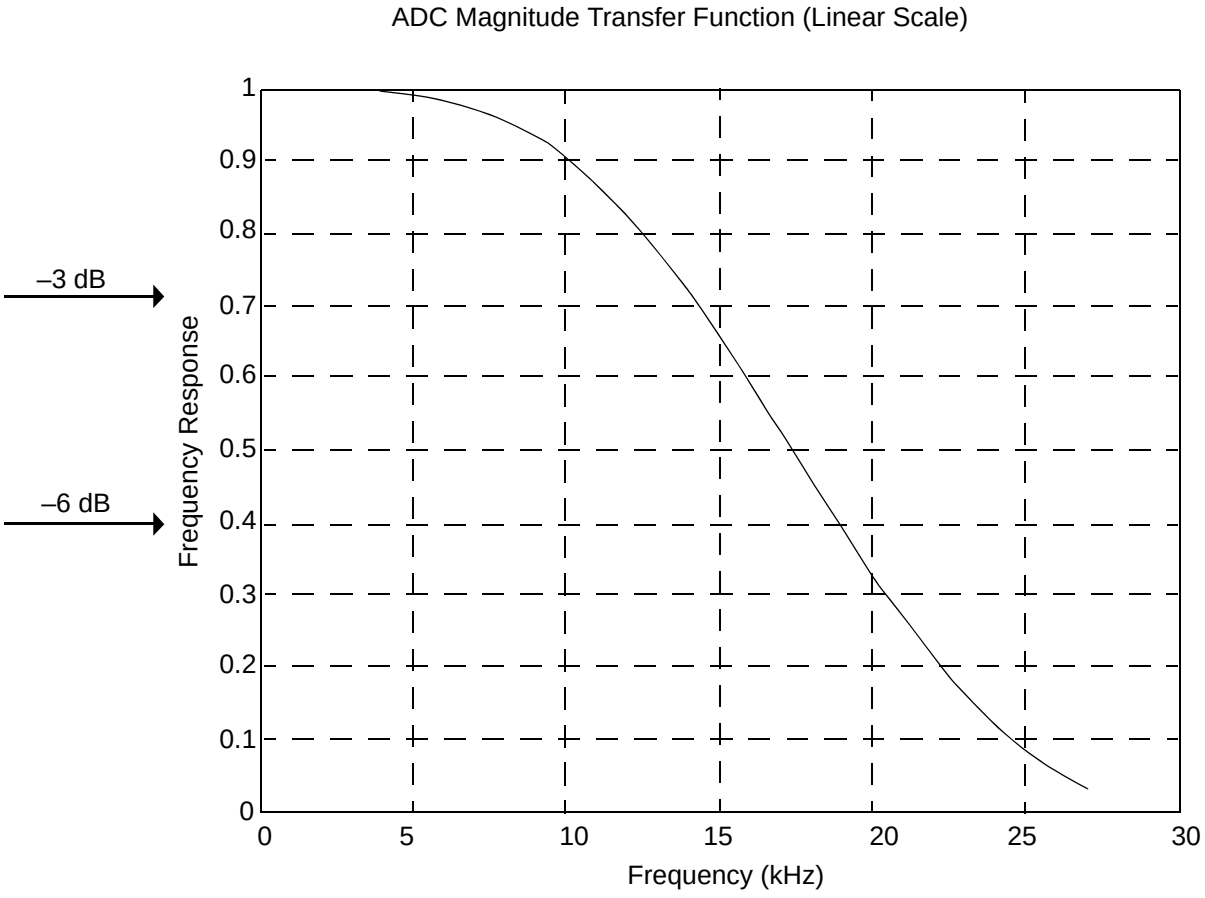


**Figure 49. Analog-to-Digital Converter Frequency Response**

**Table 137. Op Code Map Abbreviations (Continued)**

| Abbreviation | Description | Abbreviation | Description |
|---|---|---|---|
| IM | Immediate data value | r2, R2, Ir2, Irr2, IR2, rr2, RR2, IRR2, ER2 | Source address |
| Ir | Indirect working register | RA | Relative |
| IR | Indirect register | rr | Working register pair |
| Irr | Indirect working register pair | RR | Register pair |

### Hex Address: FDB

**Table 240. Port A–H Output Data Register (P*x*OUT)**

| 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: FDC

**Table 241. Port A–H GPIO Address Registers (P*x*ADDR)**

| 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: FDD

**Table 242. Port A–H Control Registers (P*x*CTL)**

| 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: FDE

**Table 243. 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.