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	ZNEO
Core Size	16-Bit
Speed	20MHz
Connectivity	EBI/EMI, I²C, IrDA, LINbus, 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)	2.7V ~ 3.6V
Data Converters	A/D 12x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	80-BQFP
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z16f6411fi20ag

10-Bit Analog-to-Digital Converter with Programmable Gain Amplifier

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

Analog Comparator

It features an on-chip analog comparator with external input pins.

Operational Amplifier

It features a two-input, one-output operational amplifier.

General-Purpose Input/Output

The Motor Control MCUs features 76 GPIO pins. Each pin is individually programmable.

Universal Asynchronous Receiver/Transmitter

It contains two fully-featured UARTs with LIN protocol support. The UART communication is full-duplex and capable of handling asynchronous data transfers. The UARTs support 8-bit and 9-bit data modes, selectable parity and an efficient bus transceiver driver enable signal for controlling a multi-transceiver bus, such as RS-485.

Infrared Encoder/Decoders

The ZNEO Z16F Series products contain two fully-functional, high-performance UART to Infrared Encoder/Decoders (Endecs). Each infrared endec is integrated with an on-chip UART to allow easy communication between the ZNEO Z16F Series device and IrDA physical layer specification Version 1.3-compliant infrared transceivers. Infrared communication provides secure, reliable, low-cost and point-to-point communication between PCs, PDAs, cell phones, printers and other infrared enabled devices.

Inter-Integrated Circuit Master/Slave Controller

The I^2C controller makes Z16F2811 compatible with the I^2C protocol. It consists of two bidirectional bus lines, a serial data (SDA) line and a serial clock (SCL) line. The I^2C operates as a Master and/or Slave and supports multi-master bus arbitration.

Enhanced Serial Peripheral Interface

The ESPI allows the data exchange between ZNEO Z16F Series and other peripheral devices such as electrically erasable programmable read-only memory (EEPROMs),

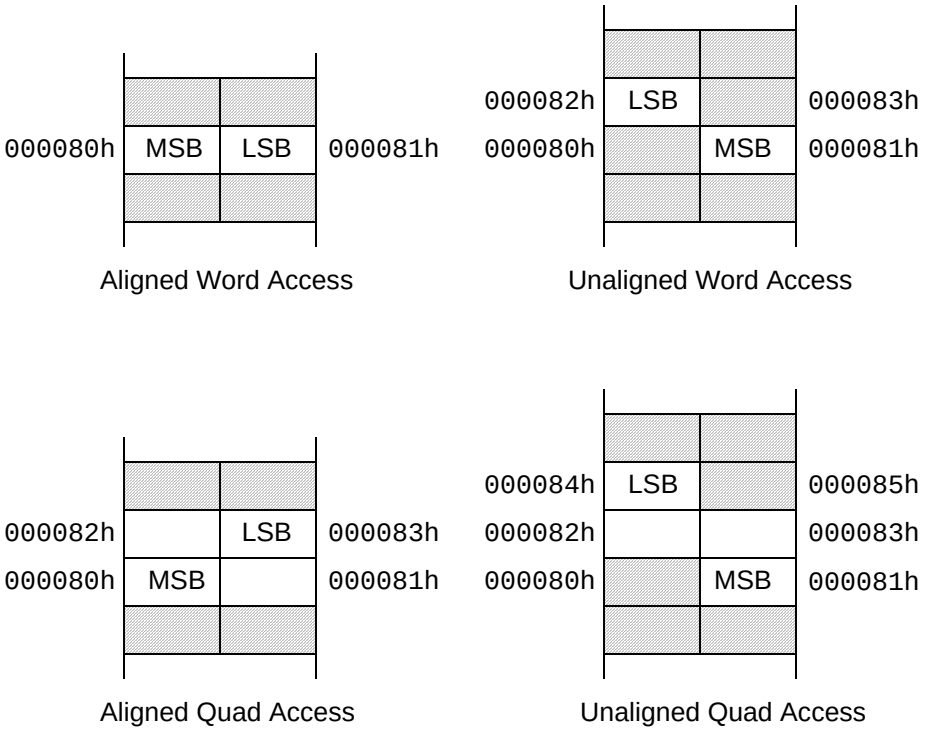

Figure 8. Alignment of Word and Quad Operations on 16-bit Memories

Table 6. Register File Address Map (Continued)

Address (Hex)	Register Description	Mnemonic	Reset (Hex)	Page No
FF_E07B	Chip Select 4 Control Low	EXTCS4L		46
FF_E07C	Chip Select 5 Control High	EXTCS5H		43
FF_E07D	Chip Select 5 Control Low	EXTCS5L		46
FF_E07E-FF_E07F	Reserved	—	—	—
On Chip Debugger = FF_E080				
FF_E080	Debug Receive Data	DBGRXD	XX	315
FF_E081	Debug Transmit Data	DBGTXD	XX	315
FF_E082-FF_E083	Debug Baud Rate	DBGBR	XXXX	316
FF_E084	Debug Line Control	DBGLCR	XX	316
FF_E085	Debug Status	DBGSTAT	XX	318
FF_E086	Debug Control	DBGCTL	XX	319
Hardware Breakpoints = FF_E090				
FF_E090-FF_E093	Hardware Breakpoint 0	HWBP0	00000000	324
FF_E094-FF_E097	Hardware Breakpoint 1	HWBP1	00000000	324
FF_E098-FF_E09B	Hardware Breakpoint 2	HWBP2	00000000	324
FF_E09C-FF_E09F	Hardware Breakpoint 3	HWBP3	00000000	324
Oscillator Control Base Address = FF_E0A0				
FF_E0A0	Oscillator Control	OSCCTL	A0	334
FF_E0A1	Oscillator Divide	OSCDIV	00	335
GPIO Base Address = FF_E100				
GPIO Port A Base Address = FF_E100				
FF_E100	Port A Input Data	PAIN	XX	71
FF_E101	Port A Output Data	PAOUT	00	72
FF_E102	Port A Data Direction	PADD	00	73
FF_E103	Port A High Drive Enable	PAHDE	00	74
FF_E104	Port A Alternate Function High	PAAFH	00	75
FF_E105	Port A Alternate Function Low	PAAFL	00	75
FF_E106	Port A Output Control	PAOC	00	76
FF_E107	Port A Pull-Up Enable	PAPUE	00	76
FF_E108	Port A Stop Mode Recovery Enable	PASMRE	00	77
FF_E109-FF_E10B	Port A Reserved	—	—	—

XX = Undefined.

Table 6. Register File Address Map (Continued)

Address (Hex)	Register Description	Mnemonic	Reset (Hex)	Page No
FF_E43A	DMA2 Source Address High	DMA2SARH	00	288
FF_E43B	DMA2 Source Address Low	DMA2SARL	00	288
FF_E43C	Reserved			
FF_E43D	DMA2 List Address Upper	DMA2LARU	00	289
FF_E43E	DMA2 List Address High	DMA2LARH	00	289
FF_E43F	DMA2 List Address Low	DMA2LARL	00	289
DMA Channel 3 Base Address = FF_E440				
FF_E440	DMA3 Control 0	DMA3CTL0	00	285
FF_E441	DMA3 Control 1	DMA3CTL1	00	285
FF_E442	DMA3 Transfer Length High	DMA3TXLNH	00	286
FF_E443	DMA3 Transfer Length Low	DMA3TXLNL	00	287
FF_E444	Reserved	—	—	—
FF_E445	DMA3 Destination Address Upper	DMA3DARU	00	287
FF_E446	DMA3 Destination Address High	DMA3DARH	00	287
FF_E447	DMA3 Destination Address Low	DMA3DARL	00	287
FF_E448	Reserved	—	—	—
FF_E449	DMA3 Source Address Upper	DMA3SARU	00	288
FF_E44A	DMA3 Source Address High	DMA3SARH	00	288
FF_E44B	DMA3 Source Address Low	DMA3SARL	00	288
FF_E44C	Reserved	—	—	—
FF_E44D	DMA3 List Address Upper	DMA3LARU	00	289
FF_E44E	DMA3 List Address High	DMA3LARH	00	289
FF_E44F	DMA3 List Address Low	DMA3LARL	00	289
Analog Block Base Address = FF_E500				
ADC Base Address = FF_E500				
FF_E500	ADC0 Control Register	ADC0CTL	00	246
FF_E501	Reserved	—	—	—
FF_E502	ADC0 Data High Byte Register	ADC0D_H	XX	247
FF_E503	ADC0 Data Low Bits Register	ADC0D_L	XX	248

XX = Undefined.

Halt Mode

Execution of the ZNEO CPU's HALT instruction places the device into Halt Mode. The following list represents the operating characteristics of the ZNEO CPU in Halt Mode:

- System clock is enabled and continues to operate
- ZNEO CPU is stopped
- PC stops incrementing
- WDT's internal RC oscillator continues to operate
- If enabled, the WDT continues to operate
- All other on-chip peripherals continue to operate

The ZNEO CPU is brought out of Halt Mode by any of the following operations:

- Interrupt or System Exception
- WDT time-out (System Exception or Reset)
- Power-On Reset
- VBO reset
- External $\overline{\text{RESET}}$ pin assertion
- Instantaneous Halt Mode recovery

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

Peripheral-Level Power Control

On-chip peripherals in ZNEO Z16F Series parts automatically enter a low power mode after Reset and whenever the peripheral is disabled. To minimize power consumption, unused peripherals must be disabled. See the individual peripheral chapters for specific register settings to enable or disable the peripheral.

Power Control Option Bits

User programmable option bits are available in some versions of the ZNEO Z16F Series devices that enable very low power Stop Mode operation. These options include disabling the VBO protection circuits and disabling the WDT oscillator. For detailed description of the user options that affect power management, see the Option Bits chapter on page 292.

Port A-K Data Direction Registers

The Port A-K Data Direction registers, shown in Table 27, configure the specified port pins as either inputs or outputs.

Table 27. Port A-K Data Direction Registers (PxDD)

Bits	7	6	5	4	3	2	1	0
Field	DD7	DD6	DD5	DD4	DD3	DD2	DD1	DD0
RESET	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Addr	FF_E102, FF_E112, FF_E122, FF_E132, FF_E142, FF_E152, FF_E162, FF_E172, FF_E182, FF_E192							

Bit	Description
[7:0] DD[7:0]	**Data Direction** These bits control the direction of the associated port pin. Port alternate function operation overrides the data direction register setting. 0 = Output Data in the Port A-K Output Data Register is driven onto the port pin. 1 = Input The port pin is sampled and the value written into the Port A-K Input Data Register. The output driver is high impedance.

Table 32. Port A-K Output Control Registers (P*x*OC)

Bits	7	6	5	4	3	2	1	0
Field	POC7	POC6	POC5	POC4	POC3	POC2	POC1	POC0
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	FF_E106, FF_E116, FF_E126, FF_E136, FF_E146, FF_E156, FF_E166, FF_E176, FF_E186, FF_E196							

Bit	Description
[7:0] POC[7:0]	**Port Output Control** These bits function independently of the alternate function bits and disable the drains if set to 1. 0 = The drains are enabled for any output mode. 1 = The drain of the associated pin is disabled (open-drain mode).

Port A-K Pull-Up Enable Registers

Setting the bits in the Port A-K Pull-Up Enable registers to 1 enables a weak internal resistive pull-up on the specified port pins. These registers affect the pins directly and as a result, alternate functions are also affected. See Table 33.

Table 33. Port A-K Pull-Up Enable Registers (P*x*PUE)

Bits	7	6	5	4	3	2	1	0
Field	PUE7	PUE6	PUE5	PUE4	PUE3	PUE2	PUE1	PUE0
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	FF_E107, FF_E117, FF_E127, FF_E137, FF_E147, FF_E157, FF_E167, FF_E177, FF_E187, FF_E197							

Bit	Description
[7:0] PUE[7:0]	**Port Pull-Up Enable** These bits function independently of the alternate function bit and enable the weak pull-up if set to 1. 0 = The weak pull-up on the port pin is disabled. 1 = The weak pull-up on the port pin is enabled.

Bit	Description (Continued)
[5] DIV0	**Divide by Zero** If this bit is 1, a divide operation was executed where the denominator was zero. Writing 1 to this bit clear it to 0.
[4] DIVOVF	**Divide Over Flow** If this bit is 1, a divide overflow occurred. A divide overflow happens when the result is greater than FFFFFFFFh. Writing 1 to this bit clears it to 0.
[3] ILL	**Illegal Instruction** If this bit is 1, an illegal instruction occurred. Writing 1 to this bit clears it to 0.
[2:0]	**Reserved** These bits are reserved and must be programmed to 000.

Table 42. System Exception Register Low (SYSEXCPL)

Bits	7	6	5	4	3	2	1	0
Field	Reserved					WDTOSC	PRIOSC	WDT
RESET	0	0	0	0	0	0	0	0
R/W	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C
Addr	FF_E021h							

Bit	Description
[7:3]	**Reserved** These bits are reserved and must be programmed to 00000.
[2] WDTOSC	**WDT Oscillator Fail** If this bit is 1, a WDT oscillator fail exception occurred. Writing 1 to this bit clears it to 0.
[1] PRIOSC	**Primary Oscillator Fail** If this bit is 1, a primary oscillator fail exception occurred. Writing 1 to this bit clears it to 0.
[0] WDT	**Watchdog Timer Interrupt** If this bit is 1, a WDT exception occurred. Writing 1 to this bit clears it to 0.

Last IRQ Register

When an interrupt occurs, the 5th bit value of the interrupt vector is stored in the Last IRQ Register, shown in Table 43. This register allows the software to determine which interrupt source was last serviced. It is used by RTOS which have a single interrupt entry point. To implement this the software must set all interrupt vectors to the entry point address. The entry point service routine then reads this register to determine which source caused the interrupt or exception and respond accordingly.

Table 62. Timer 0–2 Control 0 Register (TxCTL0)

Bits	7	6	5	4	3	2	1	0
Field	TMODE[3]	TICONFIG		CASCADE	PWMD			INCAP
RESET	0	00		0	000			0
R/W	R/W	R/W		R/W	R/W			R
Addr	FF–E306h, FF–E316h, FF–E326h							

Bit	Description
[7] TMODE[3]	**Timer Mode High Bit** This bit along with TMODE[2:0] field in T0CTL1 register determines the operating mode of the timer. This bit is the most significant bit of the timer mode selection value. For more details, see the Timer 0–2 Control 1 Register (TxCTL1) section on page 111.
[6:5] TICONFIG	**Timer Interrupt Configuration** This field configures timer interrupt definitions. These bits affect all modes. The effect per mode is explained below: ONE SHOT, Continuous, COUNTER, PWM, Compare, DUAL PWM, TRIGGERED One-Shot, COMPARATOR COUNTER: 0x Timer interrupt occurs on reload. 10 Timer interrupts are disabled. 11 Timer Interrupt occurs on reload. **Gated**: 0x Timer interrupt occurs on reload. 10 Timer interrupt occurs on inactive gate edge. 11 Timer interrupt occurs on reload. **Capture, Capture/Compare, Capture Restart**: 0x Timer interrupt occurs on reload and capture. 10 Timer interrupt occurs on capture only. 11 Timer interrupt occurs on reload only.
[4] CASCADE	**Timer Cascade** This field allows the timers to be cascaded for larger counts. Only Counter Mode must be used with this feature. 0 = The timer is not cascaded. 1 = Timer is cascaded. If timer 0 CASCADE bit is set, ANALOG COMPARATOR output is used as input. If timer 1 CASCADE bit is set, the Timer 0 output is used as the input. If timer 2 CASCADE bit is set, the timer 1 output is used as input.

PWM Output Polarity and Off-State

The default off-state and polarity of the PWM outputs are controlled by the option bits PWMHI and PWMLO. The PWMHI option controls the off-state and polarity for PWM high-side outputs PWMH0, PWMH1 and PWMH2. The PWMLO option controls the off-state and polarity for low-side outputs PWML0, PWML1 and PWML2.

The off-state is the value programmed in the option bit. For example, programming PWMHI to 1 makes the off-state of PWMH0, PWMH1 and PWMH2 a High logic value and the active state a Low logic value. Conversely, programming PWMHI to 0 causes the off-state to be a Low logic value. PWMLO is programmed in a similar manner.

PWM Enable

The MCEN option bit enables output pairs PWM0, PWM1 and PWM2. If the Motor Control option is not enabled, the PWM outputs remain in a high-impedance state after reset and is used as alternate functions like general purpose input. If the Motor Control option is enabled, following a Power-On Reset (POR) the PWM pins enter a high impedance state. As the internal reset proceeds, the PWM outputs are forced to the off-state as determined by the PWMHI and PWMLO off-state option bits.

PWM Reload Event

To prevent erroneous PWM pulse-widths and periods, registers that control the timing of the output are buffered. Buffering causes all of the PWM compare values to update. In other words, the registers controlling the duty cycle and clock source prescaler only take effect on a PWM reload event. A PWM reload event is configured to occur at the end of each PWM period or only every 2, 4, or 8 PWM periods by setting the RELFREQ bits in the PWM Control 1 Register (PWMCTL1). Software indicates that all new values are ready by setting the READY bit in the PWM Control 0 Register (PWMCTL0) to 1. When the READY bit is set to 1, the buffered values take effect at the next reload event.

PWM Prescaler

The prescaler decreases the PWM clock signal by factors of 1, 2, 4, or 8 with respect to the system clock. The PRES[1:0] bit field in the PWM Control 1 Register (PWMCTL1) controls prescaler operation. This 2-bit PRES field is buffered so that the prescale value only changes on a PWM Reload event.

PWM Period and Count Resolution

The PWM counter operates in two modes to allow edge-aligned and center-aligned outputs. Figures 22 and 23 illustrate edge and center-aligned PWM outputs. The mode in which the PWM operates determine the period of the PWM outputs (PERIOD). The programmed duty-cycle (PWMDC) and the programmed deadband time (PWMDB) deter-

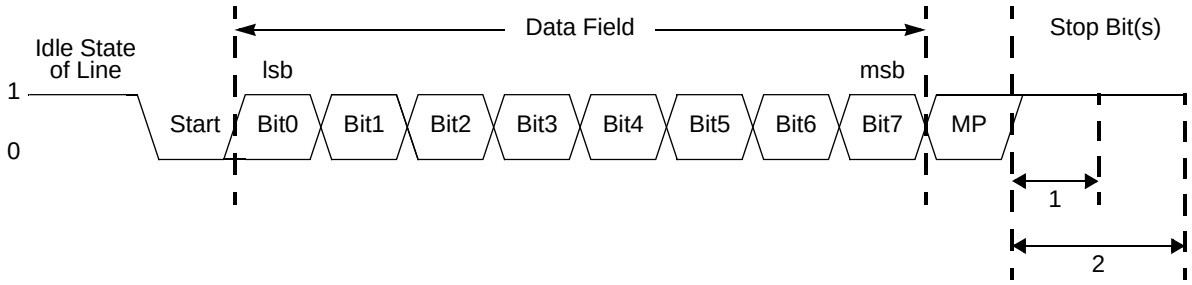

Figure 28. LIN-UART Asynchronous Multiprocessor Mode Data Format

In MULTIPROCESSOR (9-Bit) Mode, the Parity (9th) bit location becomes the MULTIPROCESSOR control bit. The LIN-UART Control 1 and Status 1 registers provide MULTIPROCESSOR (9-Bit) Mode control and status information. If an automatic address matching scheme is enabled, the LIN-UART Address Compare Register holds the network address of the device.

Multiprocessor (9-Bit) Mode Receive Interrupts

When Multiprocessor Mode is enabled, the LIN-UART processes only frames addressed to it. You can determine whether a frame of data is addressed to the LIN-UART is made in hardware, software or a combination of the two, depending on the multiprocessor configuration bits. In general, the address compare feature reduces the load on the CPU because it is not required to access the LIN-UART when it receives data directed to other devices on the multi-node network. The following 3 MULTIPROCESSOR modes are available in the hardware:

1. Interrupt on all address bytes.
2. Interrupt on matched address bytes and correctly framed data bytes.
3. Interrupt only on correctly framed data bytes.

These modes are selected with MPMD[1:0] in the LIN-UART Control 1 Register. For all MULTIPROCESSOR modes, bit MPEN of the LIN-UART Control 1 Register must be set to 1.

The first scheme is enabled by writing `01b` to MPMD[1:0]. In this mode, all incoming address bytes cause an interrupt, while data bytes never cause an interrupt. The ISR checks the address byte which triggered the interrupt. If it matches the LIN-UART address, the software clears MPMD[0]. At this point, each new incoming byte interrupts the CPU. The software determines the end of the frame and checks for it by reading the MPRX bit of the LIN-UART Status 1 Register for each incoming byte. If MPRX=1, a new frame has begun. If the address of this new frame is different from the LIN-UART's address, then

DMA Description

The DMA is used to off load the processor from doing repetitive tasks. DMA transfers data from one memory address to another memory address. Because all peripherals are mapped in memory, the DMA transfers data to or from peripherals.

The DMA transfers data from the source address to the destination address. This requires a read and/or write cycle that is generated by the DMA Controller. Each DMA transfer requires a minimum of two system clock cycles to execute.

The DMA operates in Direct or Linked List modes. Direct Mode and Linked List Mode are almost the same. In Direct Mode, the software loads the DMA Channel registers directly. In Linked List Mode, the DMA loads its registers from memory.

DMA Register Description

Each DMA channel consists of a 16-bit control register, a 16-bit transfer length register, a 24-bit destination address register, a 24-bit source address register and a 24-bit list address register; see Figure 57.

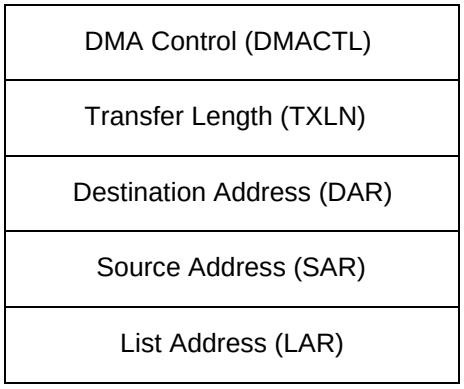

Figure 57. DMA Channel Registers

Buffers

A buffer is an allocation of contiguous memory bytes. Buffers are allocated by software to be used by the DMA. The DMA transfers data to or from buffers. A typical application would be to send data to serial channels such as I^2C, UART and SPI. The data to be sent is placed in a buffer by software.

Direct DMA Setup and Operation

Observe the following steps to set up the DMA in DIRECT Mode:

1. Write the DAMxREQSEL to select the request source.
2. Write the DMAxDAR Register with the destination address.
3. Write the DMAxSAR Register with the source address.
4. Write the DMAxTXLN with the transfer length.
5. Write DMAxLARU with watermark if required, otherwise write to 0.
6. Write DMAxCTL. Note that the control register and the address are written directly with word and quad operations.
   - DMA*x*EN: set to 1.
   - LOOP: reset to 0; not used in this mode.
   - TXSIZE: set to the transfer size, byte, word or quad.
   - DSTCTL: set to fixed, increment, or decrement.
   - SRCCTL: set to fixed, increment, or decrement.
   - IEOB: set to 1 to generate an interrupt at the end of buffer or watermark.
   - TXFR: reset to 0; not used in this mode.
   - EOF: set this bit to 1 if it is an EOF buffer.
   - HALT: reset to 0; not used in this mode.
   - CMDSTAT: set these bits with the command for the peripheral.
7. The DMA is now set up and begins operating when it receives a request.

After the DMA is set up and a request is received, the DMA performs the following operation:

1. Generates a request to the CPU.
2. Transfers data for each request until the transfer length reaches zero or the DMA receives a Request EOF signal.
3. When the DMA receives the Request EOF signal, or the transfer length reaches 0, it resets the DMAxEN bit, then performs the following operation, based upon the EOF and IEOB bits:
   a. If EOF is set, then the DMA reads the status from the peripheral and places it in the CMDSTAT field of the DMAxCTL Register.
   b. If the IEOB bit is set, or if the buffer ended with a Request EOF, the DMA Channel generates a request to the CPU.

Option Bits

Option bits allow user configuration of certain aspects of the ZNEO® Z16F Series operation. The feature configuration data is stored in the Program memory and read during Reset. The features available for control using the option bits are:

- WDT time-out response selection–interrupt or Reset
- WDT enabled at Reset
- The ability to prevent unwanted read access to user code in Program memory
- The ability to prevent accidental programming and erasure of the user code in Program memory
- Voltage Brown-Out (VBO) configuration—always enabled or disabled during Stop Mode to reduce Stop Mode power consumption
- Oscillator mode selection for high, medium and low power crystal oscillators, or external RC oscillator
- PWM pin setup for motor control application

Operation

Each time the option bits are programmed or erased, the device must be Reset for the change to take place. During any reset operation (System Reset, Short Reset, or Stop Mode Recovery), the option bits are automatically read from the Program memory and written to Option Configuration registers. The Option Configuration registers control operation of the device. Option Bit Control Register are loaded before the device exits Reset and the ZNEO CPU begins code execution. The Option Configuration registers are not part of the Register file and are not accessible for read or write access.

Option Bit Address Space

The first four bytes of Program Memory at addresses 0000h through 0003h, shown in Tables 161 and 162, respectively, are reserved for the user option bits. These bytes are used to configure user specific options. You can change the option bits to meet application requirements.

Program Memory Address 0000h

Option bits in this space are altered to change the chip configuration at reset.

On-Chip Debugger

The ZNEO® Z16F Series products have an integrated On-Chip Debugger (OCD) that provides the following features:

- Reading and writing memory
- Reading and writing CPU registers
- Execution of CPU instructions
- In-circuit programming and erasing of the Flash
- Unlimited number of software breakpoints
- Four hardware breakpoints
- Instruction execution trace
- Single-pin serial communication interface

Architecture

The OCD consists of two main blocks: the transmitter/receiver unit and the debug control logic. Figure 62 displays the architecture of the OCD.

On-Chip Debugger Commands

The hardware OCD supports several commands for controlling the device. In the following list of commands, data sent from the host to the OCD is identified by:

```
DBG <-- Data
```

Data sent from the OCD back to the host is identified by:

```
DBG --> Data
```

Multiple bytes transmitted are represented with double arrows, either <<or >>.

Read Revision. The Read Revision command returns the revision identifier.

```
DBG <-- 0000_0000
DBG --> RevID[15:8]
DBG --> RevID[7:0]
DBG --> CRC[0:7]
```

Read Status Register. The Read Status Register command returns the contents of the OCDSTAT Register.

```
DBG <-- 0000_0001
DBG --> status[7:0]
DBG --> CRC[0:7]
```

Read Control Register. The Read Control register command returns the contents of the OCDCTL Register.

```
DBG <-- 0000_0010
DBG --> OCDCTL[7:0]
DBG --> CRC[0:7]
```

Write Control Register. The Write Control register command writes data to the OCDCTL Register.

```
DBG <-- 0000_0011
DBG <-- OCDCTL[7:0]
DBG --> CRC[0:7]
```

Read Registers. The Read registers command returns the contents of CPU registers R15 through R0.

```
DBG <-- 0000_0100
DBG ->> regdata[31:24]
DBG ->> regdata[23:16]
DBG ->> regdata[15:8]
DBG ->> regdata[7:0]
DBG --> CRC[0:7]
```

Write Registers. The Write registers command writes data to CPU registers R15 through R0.

```
DBG <-- 0000_0101
DBG <<- regdata[31:24]
```

```
DBG <<- regdata[23:16]
DBG <<- regdata[15:8]
DBG <<- regdata[7:0]
DBG --> CRC[0:7]
```

Read PC. The Read Program Counter command returns the contents of the program counter.

```
DBG <-- 0000_0110
DBG --> 00h
DBG --> PC[23:16]
DBG --> PC[15:8]
DBG --> PC[7:0]
DBG --> CRC[0:7]
```

Write PC. The Write Program Counter command writes data to the program counter.

```
DBG <-- 0000_0111
DBG <-- 00h
DBG <-- PC[23:16]
DBG <-- PC[15:8]
DBG <-- PC[7:0]
DBG --> CRC[0:7]
```

Read Flags. The Read Flags command returns the contents of the CPU flags.

```
DBG <-- 0000_1000
DBG --> 00h
DBG --> flags[7:0]
DBG --> CRC[0:7]
```

Write Instruction. The Write Instruction command writes one word of Op Code to the CPU.

```
DBG <-- 0000_1001
DBG <-- opcode[15:8]
DBG <-- opcode[7:0]
DBG --> CRC[0:7]
```

Read Register. The Read Register command returns the contents of a single CPU register.

```
DBG <-- {0100,regno[3:0]}
DBG --> regdata[31:24]
DBG --> regdata[23:16]
DBG --> regdata[15:8]
DBG --> regdata[7:0]
DBG --> CRC[0:7]
```

Write Register. The Write Register command writes data to a single CPU register.

```
DBG <-- {0101,regno[3:0]}
DBG <-- regdata[31:24]
DBG <-- regdata[23:16]
DBG <-- regdata[15:8]
DBG <-- regdata[7:0]
DBG --> CRC[0:7]
```

Figure 74 displays typical current consumption while operating at 3.3 V at 30ºC in Halt Mode versus the system clock frequency.

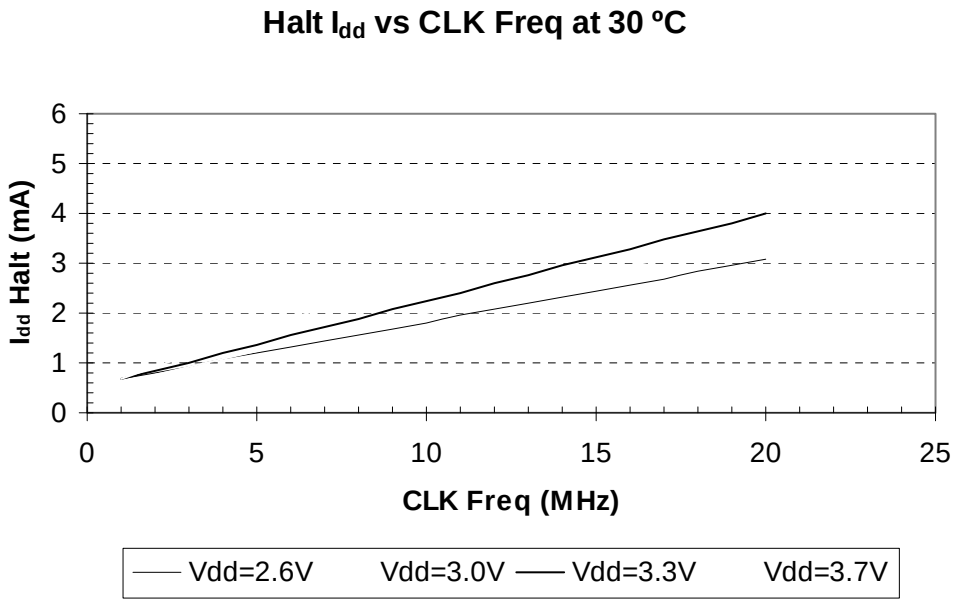

Figure 74. Typical Halt Mode I_{DD} Versus System Clock Frequency

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 or consider participating in the Zilog 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.