Zilog - Z8F1602VS020EC00TR 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	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	46
Program Memory Size	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 3.6V
Data Converters	A/D 12x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	68-LCC (J-Lead)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f1602vs020ec00tr

This publication is subject to replacement by a later edition. To determine whether a later edition exists, or to request copies of publications, contact:

ZiLOG Worldwide Headquarters
532 Race Street
San Jose, CA 95126
Telephone: 408.558.8500
Fax: 408.558.8300
www.ZiLOG.com

Document Disclaimer

Table of Contents

Introduction 1
Features 1
Part Selection Guide 2
Block Diagram 3
CPU and Peripheral Overview 3
eZ8 CPU Features 3
General Purpose I/O 4
Flash Controller 4
10-Bit Analog-to-Digital Converter 4
UARTs 4
I^2C 4
Serial Peripheral Interface 5
Timers 5
Interrupt Controller 5
Reset Controller 5
On-Chip Debugger 5
DMA Controller 5
Signal and Pin Descriptions 6
Overview 6
Available Packages 6
Pin Configurations 7
Signal Descriptions 13
Pin Characteristics 15
Address Space 17
Overview 17
Register File 17
Program Memory 18
Data Memory 19
Register File Address Map 0
Reset and Stop Mode Recovery 25
Overview 25
Reset Types 25
System and Short Resets 26
Reset Sources 26
Power-On Reset 27
Voltage Brown-Out Reset 27
Watch-Dog Timer Reset 28

External Pin Reset 29
Stop Mode Recovery 29
Stop Mode Recovery Using Watch-Dog Timer Time-Out 29
Stop Mode Recovery Using a GPIO Port Pin Transition 30
Low-Power Modes 31
Overview 31
Stop Mode 31
Halt Mode 31
General-Purpose I/O 33
Overview 33
GPIO Port Availability By Device 33
Architecture 34
GPIO Alternate Functions 34
GPIO Interrupts 36
GPIO Control Register Definitions 36
Port A-H Address Registers 37
Port A-H Control Registers 38
Port A-H Input Data Registers 42
Port A-H Output Data Register 43
Interrupt Controller 44
Overview 44
Interrupt Vector Listing 44
Architecture 46
Operation 46
Master Interrupt Enable 46
Interrupt Vectors and Priority 47
Interrupt Assertion Types 47
Interrupt Control Register Definitions 48
Interrupt Request 0 Register 48
Interrupt Request 1 Register 49
Interrupt Request 2 Register 50
IRQ0 Enable High and Low Bit Registers 51
IRQ1 Enable High and Low Bit Registers 52
IRQ2 Enable High and Low Bit Registers 53
Interrupt Edge Select Register 54
Interrupt Port Select Register 55
Interrupt Control Register 56
Timers 57
Overview 57
Architecture 57
Operation 58

Table 32. IRQ2 Enable and Priority Encoding 53
Table 33. IRQ1 Enable High Bit Register (IRQ1ENH) 53
Table 34. IRQ2 Enable Low Bit Register (IRQ2ENL) 54
Table 35. IRQ2 Enable High Bit Register (IRQ2ENH) 54
Table 36. Interrupt Edge Select Register (IRQES) 55
Table 37. Interrupt Port Select Register (IRQPS) 55
Table 38. Interrupt Control Register (IRQCTL) 56
Table 39. Timer 0-3 High Byte Register (TxH) 67
Table 40. Timer 0-3 Low Byte Register (TxL) 67
Table 41. Timer 0-3 Reload High Byte Register (TxRH) 68
Table 42. Timer 0-3 Reload Low Byte Register (TxRL) 68
Table 43. Timer 0-3 PWM High Byte Register (TxPWMH) 69
Table 44. Timer 0-3 PWM Low Byte Register (TxPWML) 69
Table 45. Timer 0-3 Control Register (TxCTL) 70
Table 46. Watch-Dog Timer Approximate Time-Out Delays 73
Table 47. Watch-Dog Timer Control Register (WDTCTL) 75
Table 48. Watch-Dog Timer Reload Upper Byte Register (WDTU) 76
Table 49. Watch-Dog Timer Reload High Byte Register (WDTH) . 76
Table 50. Watch-Dog Timer Reload Low Byte Register (WDTL) . . 77
Table 51. UARTx Transmit Data Register (UxTXD) 86
Table 52. UARTx Receive Data Register (UxRXD) 87
Table 53. UARTx Status 0 Register (UxSTAT0) 87
Table 54. UARTx Control 0 Register (UxCTL0) 89
Table 55. UARTx Status 1 Register (UxSTAT1) 89
Table 56. UARTx Control 1 Register (UxCTL1) 90
Table 57. UARTx Baud Rate High Byte Register (UxBRH) 91
Table 58. UARTx Baud Rate Low Byte Register (UxBRL) 92
Table 59. UART Baud Rates 93
Table 60. SPI Clock Phase (PHASE) and Clock Polarity (CLKPOL) Operation 102
Table 61. SPI Data Register (SPIDATA) 106
Table 62. SPI Control Register (SPICTL) 107
Table 63. SPI Status Register (SPISTAT) 108
Table 64. SPI Mode Register (SPIMODE) 109
Table 65. SPI Baud Rate High Byte Register (SPIBRH) 110
Table 66. SPI Baud Rate Low Byte Register (SPIBRL) 110

Signal Descriptions

Table 2 describes the Z8F640x family signals. Refer to the section **Pin Configurations on page 7** to determine the signals available for the specific package styles.

Table 2. Signal Descriptions

Signal Mnemonic	I/O	Description
General-Purpose I/O Ports A-H		
PA[7:0]	I/O	Port A[7:0]. These pins are used for general-purpose I/O.
PB[7:0]	I/O	Port B[7:0]. These pins are used for general-purpose I/O.
PC[7:0]	I/O	Port C[7:0]. These pins are used for general-purpose I/O.
PD[7:0]	I/O	Port D[7:0]. These pins are used for general-purpose I/O.
PE[7:0]	I/O	Port E[7:0]. These pins are used for general-purpose I/O.
PF[7:0]	I/O	Port F[7:0]. These pins are used for general-purpose I/O.
PG[7:0]	I/O	Port G[7:0]. These pins are used for general-purpose I/O.
PH[3:0]	I/O	Port H[3:0]. These pins are used for general-purpose I/O.
I^2C Controller		
SCL	O	Serial Clock. This is the output clock for the I^2C. This pin is multiplexed with a general-purpose I/O pin. When the general-purpose I/O pin is configured for alternate function to enable the SCL function, this pin is open-drain.
SDA	I/O	Serial Data. This open-drain pin is used to transfer data between the I^2C and a slave. This pin is multiplexed with a general-purpose I/O pin. When the general-purpose I/O pin is configured for alternate function to enable the SDA function, this pin is open-drain.
SPI Controller		
$\overline{SS}$	I/O	Slave Select. This signal can be an output or an input. If the Z8 Encore! is the SPI master, this pin may be configured as the Slave Select output. If the Z8 Encore! is the SPI slave, this pin is the input slave select. It is multiplexed with a general-purpose I/O pin.
SCK	I/O	SPI Serial Clock. The SPI master supplies this pin. If the Z8 Encore! is the SPI master, this pin is an output. If the Z8 Encore! is the SPI slave, this pin is an input. It is multiplexed with a general-purpose I/O pin.
MOSI	I/O	Master Out Slave In. This signal is the data output from the SPI master device and the data input to the SPI slave device. It is multiplexed with a general-purpose I/O pin.
MISO	I/O	Master In Slave Out. This pin is the data input to the SPI master device and the data output from the SPI slave device. It is multiplexed with a general-purpose I/O pin.

Port A-H Data Direction Sub-Registers

The Port A-H Data Direction sub-register is accessed through the Port A-H Control register by writing 01H to the Port A-H Address register (Table 15).

Table 15. Port A-H Data Direction Sub-Registers

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	If 01H in Port A-H Address Register, accessible via Port A-H Control Register							

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-H Output Data register is driven onto the port pin.
1 = Input. The port pin is sampled and the value written into the Port A-H Input Data Register. The output driver is tri-stated.

Port A-H Alternate Function Sub-Registers

The Port A-H Alternate Function sub-register (Table 16) is accessed through the Port A-H Control register by writing 02H to the Port A-H Address register. The Port A-H Alternate Function sub-registers select the alternate functions for the selected pins. Refer to the **GPIO Alternate Functions** section to determine the alternate function associated with each port pin.

Caution: Do not enable alternate function for GPIO port pins which do not have an associated alternate function. Failure to follow this guideline may result in unpredictable operation.

Table 16. Port A-H Alternate Function Sub-Registers

BITS	7	6	5	4	3	2	1	0
FIELD	AF7	AF6	AF5	AF4	AF3	AF2	AF1	AF0
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	If 02H in Port A-H Address Register, accessible via Port A-H Control Register							

If the Timer reaches FFFFH, the timer rolls over to 0000H and continue counting.

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

1. Write to the Timer Control register to:
   – Disable the timer
   – Configure the timer for Compare mode.
   – Set the prescale value.
   – Set the initial logic level (High or Low) for the Timer Output alternate function, if desired.
2. Write to the Timer High and Low Byte registers to set the starting count value.
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.
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 given by the following equation:

$$\textbf{Compare Mode Time (s)} = \frac{(\textbf{Compare Value} - \textbf{Start Value}) \times \textbf{Prescale}}{\textbf{System Clock Frequency (Hz)}}$$

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 if a 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. When 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 is still asserted). Also, 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.

The steps for configuring a timer for Gated mode and initiating the count are as follows:

1. Write to the Timer Control register to:
   – Disable the timer

Serial Peripheral Interface

Overview

The Serial Peripheral Interface™ (SPI) is a synchronous interface allowing several SPI-type devices to be interconnected. SPI-compatible devices include EEPROMs, Analog-to-Digital Converters, and ISDN devices. Features of the SPI include:

- Full-duplex, synchronous, character-oriented communication
- Four-wire interface
- Data transfers rates up to a maximum of one-fourth the system clock frequency
- Error detection
- Write and mode collision detection
- Dedicated Baud Rate Generator

Architecture

The SPI may be configured as either a Master (in single or multi-master systems) or a Slave as illustrated in Figures 74 through 76.

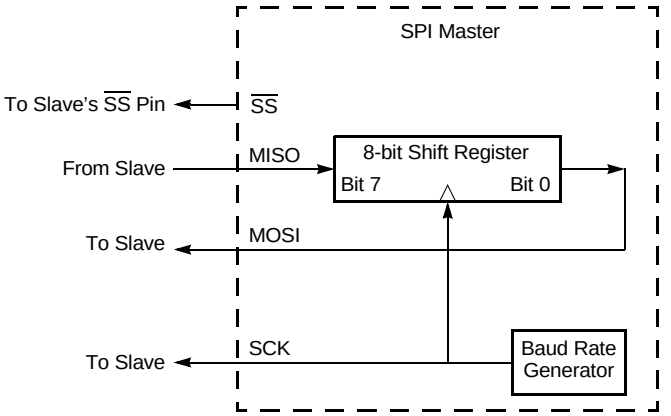

Figure 74. SPI Configured as a Master in a Single Master, Single Slave System

I²C Controller

Overview

The I²C Controller makes the Z8F640x family device bus-compatible with the I²C™ protocol. The I²C Controller consists of two bidirectional bus lines—a serial data signal (SDA) and a serial clock signal (SCL). Features of the I²C Controller include:

- Transmit and Receive Operation in Master mode
- Maximum data rate of 400kbit/sec
- 7- and 10-bit Addressing Modes for Slaves
- Unrestricted Number of Data Bytes Transmitted per Transfer

The I²C Controller in the Z8F640x family device does not operate in Slave mode.

Operation

The I²C Controller operates in Master mode to transmit and receive data. Only a single master is supported. Arbitration between two masters must be accomplished in software. I²C supports the following operations:

- Master transmits to a 7-bit slave
- Master transmits to a 10-bit slave
- Master receives from a 7-bit slave
- Master receives from a 10-bit slave

SDA and SCL Signals

I²C sends all addresses, data and acknowledge signals over the SDA line, most-significant bit first. SCL is the common clock for the I²C Controller. When the SDA and SCL pin alternate functions are selected for their respective GPIO ports, the pins are automatically configured for open-drain operation.

The master (I²C) is responsible for driving the SCL clock signal, although the clock signal can become skewed by a slow slave device. During the high period of the clock, the slave pulls the SCL signal Low to suspend the transaction. When the slave has released the line, the I²C Controller continues the transaction. All data is transferred in bytes and there is no

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.

START—Send Start Condition
This bit sends the Start condition. Once asserted, it is cleared by the I2C Controller after it sends the START condition or by deasserting the IEN bit. After this bit is set, the Start condition is sent if there is data in the I2C Data or I2C Shift register. If there is no data in one of these registers, the I2C Controller waits until data is loaded. If this bit is set while the I2C Controller is shifting out data, it generates a START condition after the byte shifts and the acknowledge phase completed. If the STOP bit is also set, it also waits until the STOP condition is sent before the START condition. If this bit is 1, it cannot be cleared to 0 by writing to the register.This bit clears when the I2C is disabled.

STOP—Send Stop Condition
This bit causes the I2C Controller to issue a Stop condition after the byte in the I2C Shift register has completed transmission or after a byte has been received in a receive operation. Once set, this bit is reset by the I2C Controller after a Stop condition has been sent or by deasserting the IEN bit. If this bit is 1, it cannot be cleared to 0 by writing to the register.This bit clears when the I2C is disabled.

BIRQ—Baud Rate Generator Interrupt Request
This bit causes an interrupt to occur every time the baud rate generator counts down to zero. This bit allows the I2C Controller to be used as an additional counter when it is not being used elsewhere. This bit must only be set when the I2C Controller is disabled.

TXI—Enable TDRE interrupts
This bit enables interrupts when the I2C Data register is empty on the I2C Controller.

NAK—Send NAK
This bit sends a Not Acknowledge condition after the next byte of data has been read from the I2C slave. Once asserted, it is deasserted after a Not Acknowledge is sent or the IEN bit is deasserted.

FLUSH—Flush Data
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 an NAK is received after the data has been sent to the I2C Data register. Reading this bit always returns 0.

FILTEN—I2C Signal Filter Enable
Setting this bit to 1 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.

DMA*x* Start/Current Address Low Byte Register

The DMA*x* Start/Current Address Low register, in conjunction with the DMA*x* Address High Nibble register, forms a 12-bit Start/Current Address. Writes to this register set the Start Address for DMA operations. Each time the DMA completes a data transfer, the 12-bit Start/Current Address increments by either 1 (single-byte transfer) or 2 (two-byte word transfer). Reads from this register return the low byte of the Current Address to be used for the next DMA data transfer.

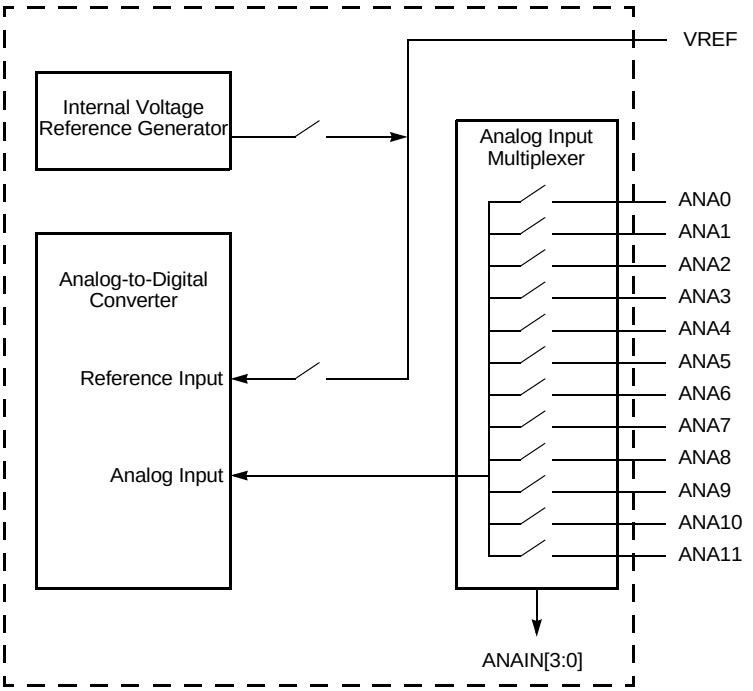

Figure 83. Analog-to-Digital Converter Block Diagram

Operation

Automatic Power-Down

If the ADC is idle (no conversions in progress) for 160 consecutive system clock cycles, portions of the ADC are automatically powered-down. From this power-down state, the ADC requires 40 system clock cycles to power-up. The ADC powers up when a conversion is requested using the ADC Control register.

Single-Shot Conversion

When configured for single-shot conversion, the ADC performs a single analog-to-digital conversion on the selected analog input channel. After completion of the conversion, the ADC shuts down. The steps for setting up the ADC and initiating a single-shot conversion are as follows:

Program Memory Address 0000H

Table 90. Option Bits At Program Memory Address 0000H

BITS	7	6	5	4	3	2	1	0
FIELD	WDT_RES	WDT_AO	Reserved			RP	FHSWP	FWP
RESET	U	U	U	U	U	U	U	U
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
ADDR	Program Memory 0000H							

Note: U = Unchanged by Reset. R/W = Read/Write.

WDT_RES—Watch-Dog Timer Reset
0 = Watch-Dog Timer time-out generates an interrupt request. Interrupts must be globally enabled for the eZ8 CPU to acknowledge the interrupt request.
1 = Watch-Dog Timer time-out causes a Short Reset. This setting is the default for unprogrammed (erased) Flash.

WDT_AO—Watch-Dog Timer Always On
0 = Watch-Dog Timer is automatically enabled upon application of system power. Watch-Dog Timer can not be disabled.
1 = Watch-Dog Timer is enabled upon execution of the WDT instruction. Once enabled, the Watch-Dog Timer can only be disabled by a Reset or Stop Mode Recovery. This setting is the default for unprogrammed (erased) Flash.

Reserved
These Option Bits are reserved for future use and must always be set to 1. This setting is the default for unprogrammed (erased) Flash.

RP—Read Protect
0 = User program code is inaccessible. Limited control features are available through the On-Chip Debugger.
1 = User program code is accessible. All On-Chip Debugger commands are enabled. This setting is the default for unprogrammed (erased) Flash.

Operation

OCD Interface

The On-Chip Debugger uses the DBG pin for communication with an external host. This one-pin interface is a bi-directional open-drain interface that transmits and receives data. Data transmission is half-duplex, in that transmit and receive cannot occur simultaneously. The serial data on the DBG pin is sent using the standard asynchronous data format defined in RS-232. This pin can interface the Z8F640x family device to the serial port of a host PC using minimal external hardware.Two different methods for connecting the DBG pin to an RS-232 interface are depicted in Figures 87 and 88.

Caution: For operation of the On-Chip Debugger, ***all*** power pins (VDD and AVDD) must be supplied with power, and ***all*** ground pins (VSS and AVSS) must be properly grounded.
The DBG pin is open-drain and must always be connected to V_{DD} through an external pull-up resistor to ensure proper operation.

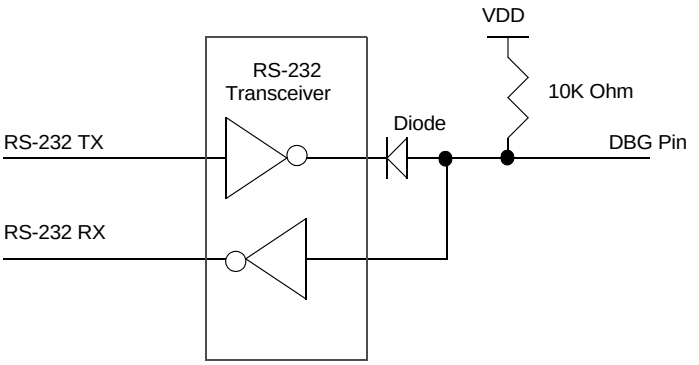

Figure 87. Interfacing the On-Chip Debugger's DBG Pin with an RS-232 Interface (1)

ister. When the Watchpoint event occurs, the Z8F640x family device enters Debug mode and the DBGMODE bit in the OCDCTL register becomes 1.

Runtime Counter

The On-Chip Debugger contains a 16-bit Runtime Counter. It counts system clock cycles between Breakpoints. The counter starts counting when the On-Chip Debugger leaves Debug mode and stops counting when it enters Debug mode again or when it reaches the maximum count of FFFFH.

On-Chip Debugger Commands

The host communicates to the On-Chip Debugger by sending OCD commands using the DBG interface. During normal operation of the Z8F640x family device, only a subset of the OCD commands are available. In Debug mode, all OCD commands become available unless the user code and control registers are protected by programming the Read Protect Option Bit (RP). The Read Protect Option Bit prevents the code in memory from being read out of the Z8F640x family device. When this option is enabled, several of the OCD commands are disabled. Table 93 contains a summary of the On-Chip Debugger commands. Each OCD command is described in further detail in the bulleted list following Table 93. Table 93 indicates those commands that operate when the Z8F640x family device is not in Debug mode (normal operation) and those commands that are disabled by programming the Read Protect Option Bit.

Table 93. On-Chip Debugger Commands

Debug Command	Command Byte	Enabled when NOT in Debug mode?	Disabled by Read Protect Option Bit
Read OCD Revision	00H	Yes	-
Reserved	01H	-	-
Read OCD Status Register	02H	Yes	-
Read Runtime Counter	03H	-	-
Write OCD Control Register	04H	Yes	Cannot clear DBGMODE bit
Read OCD Control Register	05H	Yes	-
Write Program Counter	06H	-	Disabled
Read Program Counter	07H	-	Disabled
Write Register	08H	-	Only writes of the Flash Memory Control registers are allowed. Additionally, only the Mass Erase command is allowed to be written to the Flash Control register.
Read Register	09H	-	Disabled

BRKEN—Breakpoint Enable
This bit controls the behavior of the BRK instruction (opcode 00H). By default, Breakpoints are disabled and the BRK instruction behaves like a NOP. If this bit is set to 1, when a BRK instruction is decoded, the DBGMODE bit of the OCDCTL register is automatically set to one.
0 = Breakpoints are disabled.
1 = Breakpoints are enabled.

DBGACK—Debug Acknowledge
This bit enables the debug acknowledge feature. If this bit is set to 1, then the OCD sends an Debug Acknowledge character (FFH) to the host when a Breakpoint or Watchpoint occurs.
0 = Debug Acknowledge is disabled.
1 = Debug Acknowledge is enabled.

Reserved
These bits are reserved and must be 0.

RST—Reset
Setting this bit to 1 resets the Z8F640x family device. The device goes through a normal Power-On Reset sequence with the exception that the On-Chip Debugger is not reset. This bit is automatically cleared to 0 when the reset finishes.
0 = No effect.
1 = Reset Z8F640x family device.

OCD Status Register

The OCD Status register reports status information about the current state of the debugger and the Z8F640x family device.

Table 95. OCD Status Register (OCDSTAT)

BITS	7	6	5	4	3	2	1	0
FIELD	DBG	HALT	RPEN	Reserved				
RESET	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

DBG—Debug Status
0 = The Z8F640x family device is operating in normal mode.
1 = The Z8F640x family device is in Debug mode.

HALT—Halt Mode
0 = The Z8F640x family device is not in Halt mode.
1 = The Z8F640x family device is in Halt mode.

On-Chip Debugger Timing

Figure 95 and Table 109 provide timing information for DBG pins. The timing specifications presume a rise and fall time on DBG of less than 4µs.

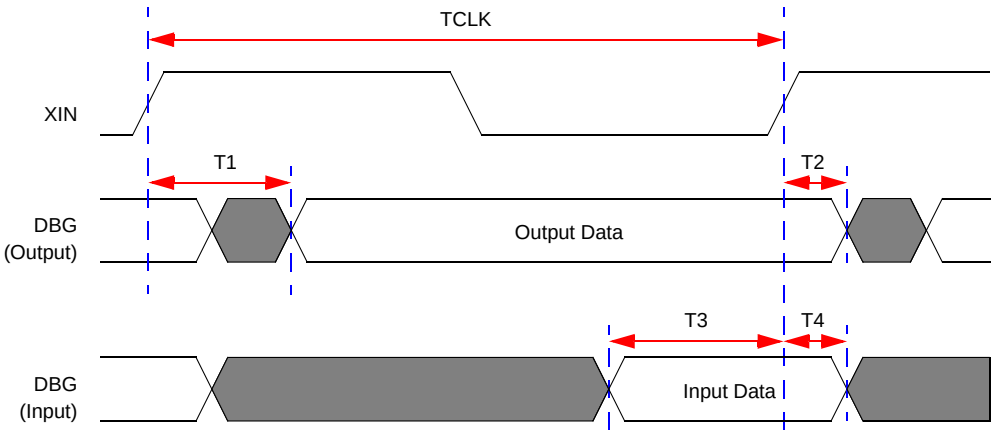

Figure 95. On-Chip Debugger Timing

Table 109. On-Chip Debugger Timing

		Delay (ns)	
Parameter	**Abbreviation**	**Minimum**	**Maximum**
DBG			
T_1	XIN Rise to DBG Valid Delay	–	15
T_2	XIN Rise to DBG Output Hold Time	2	–
T_3	DBG to XIN Rise Input Setup Time	10	–
T_4	DBG to XIN Rise Input Hold Time	5	–
	DBG frequency		System Clock / 4

Table 118. Arithmetic Instructions (Continued)

Mnemonic	Operands	Instruction
SBC	dst, src	Subtract with Carry
SBCX	dst, src	Subtract with Carry using Extended Addressing
SUB	dst, src	Subtract
SUBX	dst, src	Subtract using Extended Addressing

Table 119. Bit Manipulation Instructions

Mnemonic	Operands	Instruction
BCLR	bit, dst	Bit Clear
BIT	p, bit, dst	Bit Set or Clear
BSET	bit, dst	Bit Set
BSWAP	dst	Bit Swap
CCF	—	Complement Carry Flag
RCF	—	Reset Carry Flag
SCF	—	Set Carry Flag
TCM	dst, src	Test Complement Under Mask
TCMX	dst, src	Test Complement Under Mask using Extended Addressing
TM	dst, src	Test Under Mask
TMX	dst, src	Test Under Mask using Extended Addressing

Table 120. Block Transfer Instructions

Mnemonic	Operands	Instruction
LDCI	dst, src	Load Constant to/from Program Memory and Auto-Increment Addresses
LDEI	dst, src	Load External Data to/from Data Memory and Auto-Increment Addresses

Flags Register

The Flags Register contains the status information regarding the most recent arithmetic, logical, bit manipulation or rotate and shift operation. The Flags Register contains six bits of status information that are set or cleared by CPU operations. Four of the bits (C, V, Z and S) can be tested for use with conditional jump instructions. Two flags (H and D) cannot be tested and are used for Binary-Coded Decimal (BCD) arithmetic.

The two remaining bits, User Flags (F1 and F2), are available as general-purpose status bits. User Flags are unaffected by arithmetic operations and must be set or cleared by instructions. The User Flags cannot be used with conditional Jumps. They are undefined at initial power-up and are unaffected by Reset. Figure 99 illustrates the flags and their bit positions in the Flags Register.

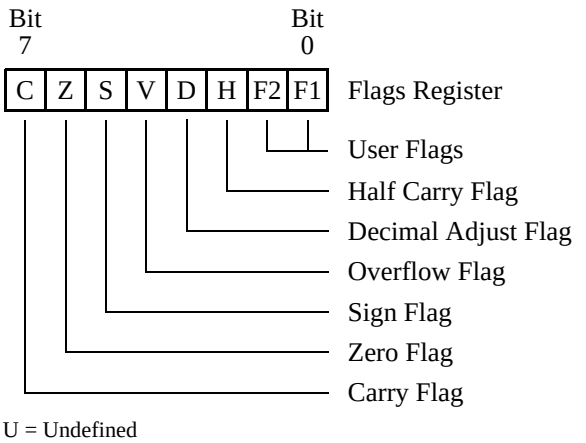

Figure 99. Flags Register

Interrupts, the Software Trap (TRAP) instruction, and Illegal Instruction Traps all write the value of the Flags Register to the stack. Executing an Interrupt Return (IRET) instruction restores the value saved on the stack into the Flags Register.