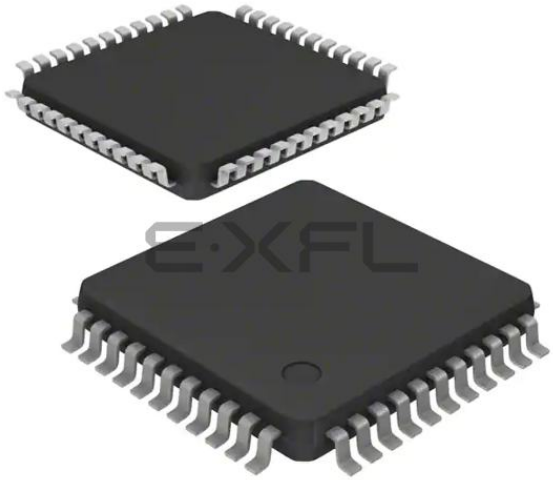

Welcome to E-XFL.COM

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	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	44-LQFP
Supplier Device Package	44-LQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f2421an020ec

I²C Status

I2CSTAT (F51H - Read Only)

D7	D6	D5	D4	D3	D2	D1	D0

- D0 — NACK Interrupt
  0 = No action required to service NAK
  1 = START/STOP not set after NAK
- D1 — Data Shift State
  0 = Data is not being transferred
  1 = Data is being transferred
- D2 — Transmit Address State
  0 = Address is not being transferred
  1 = Address is being transferred
- D3 — Read
  0 = Write operation
  1 = Read operation
- D4 — 10-Bit Address
  0 = 7-bit address being transmitted
  1 = 10-bit address being transmitted
- D5 — Acknowledge
  0 = Acknowledge not transmitted/received
  1 = For last byte, Acknowledge was transmitted/received
- D6 — Receive Data Register Full
  0 = I2C has not received data
  1 = Data register contains received data
- D7 — Transmit Data Register Empty
  0 = Data register is full
  1 = Data register is empty

I²C Control

I2CCTL (F52H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

- D0 — I2C Signal Filter Enable
  0 = Digital filtering disabled
  1 = Low-pass digital filters enabled on SDA and SCL input signals
- D1 — Flush Data
  0 = No effect
  1 = Clears I2C Data register
- D2 — Send NAK
  0 = Do not send NAK
  1 = Send NAK after next byte received from slave
- D3 — Enable TDRE Interrupts
  0 = Do not generate an interrupt when the I2C Data register is empty
  1 = Generate an interrupt when the I2C Transmit Data register is empty
- D4 — Baud Rate Generator Interrupt
  0 = Interrupts behave as set by I2C control
  1 = BRG generates an interrupt when it counts down to zero
- D5 — Send Stop Condition
  0 = Do not issue Stop condition after data transmission is complete
  1 = Issue Stop condition after data transmission is complete
- D6 — Send Start Condition
  0 = Do not send Start Condition
  1 = Send Start Condition
- D7 — I2C Enable
  0 = I2C is disabled
  1 = I2C is enabled

I2C Baud Rate Generator High Byte

I2CBRH (F53H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

- D7–D0 — I2C Baud Rate divisor [15:8]

I2C Baud Rate Generator Low Byte

I2CBRL (F54H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

- D7–D0 — I2C Baud Rate divisor [7:0]

SPI Mode
SPIMODE (F63H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

- D7: Reserved
- D6–D5: 111 = 7 bits
- D4: Diagnostic Mode Control
  0 = Reading from SPIBRH, SPIBRL returns reload values
  1 = Reading from SPIBRH, SPIBRL returns current BRG count value
- D3–D0: Reserved

SPI Diagnostic State
SPIDST (F64H - Read Only)

D7	D6	D5	D4	D3	D2	D1	D0

- D7: Shift Clock Enable
  0 = Internal shift clock enable signal is deasserted
  1 = Internal shift clock enable signal is asserted
- D6: Transmit Clock Enable
  0 = Internal transmit clock enable signal is deasserted
  1 = Internal transmit clock enable signal is asserted
- D5–D0: SPI State

SPI Baud Rate Generator High Byte
SPIBRH (F66H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

- D7–D0: SPI Baud Rate divisor [15:8]

SPI Baud Rate Generator Low Byte
SPIBRL (F67H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

- D7–D0: SPI Baud Rate divisor [7:0]

ADC Control
ADCCTL (F70H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

- D7: Conversion Enable
  0 = Conversion is complete
  1 = Begin conversion
- D6: Reserved
- D5: External VREF select
  0 = Internal voltage reference selected
  1 = External voltage reference selected
- D4: Continuous Mode Select
  0 = Single-shot conversion
  1 = Continuous conversion
- D3–D0: Analog Input Select
  0000 = ANA0 0001 = ANA1
  0010 = ANA2 0011 = ANA3
  0100 = ANA4 0101 = ANA5
  0110 = ANA6 0111 = ANA7
  1000 = ANA8 1001 = ANA9
  1010 = ANA10 1011 = ANA11
  11xx = Reserved

ADC Data High Byte
ADCD_H (F72H - Read Only)

D7	D6	D5	D4	D3	D2	D1	D0

- D7–D0: ADC Data [9:2]

ADC Data Low Bits
ADCD_L (F73H - Read Only)

D7	D6	D5	D4	D3	D2	D1	D0

- D7–D6: ADC Data [1:0]
- D5–D0: Reserved

DMA1 Control
DMA1CTL (FB8H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Request Trigger Source Select
000 = Timer 0
001 = Timer 1
010 = Timer 2
011 = Timer 3
100 = UART0 Transmit Data register is empty
101 = UART1 Transmit Data register is empty
110 = I2C Transmit Data register is empty
111 = Reserved

Word Select
0 = DMA transfers 1 byte per request
1 = DMA transfers 2 bytes per request

DMA1 Interrupt Enable
0 = DMA1 does not generate interrupts
1 = DMA1 generates an interrupt when End Address data is transferred

DMA1 Data Transfer Direction
0 = Register File to peripheral registers
1 = Peripheral registers to Register File

DMA1 Loop Enable
0 = DMA disables after End Address
1 = DMA reloads Start Address after End Address and continues to run

DMA1 Enable
0 = DMA1 is disabled
1 = DMA1 is enabled

DMA1 I/O Address
DMA1IO (FB9H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

DMA1 Peripheral Register Address
Low byte of on-chip peripheral control registers on Register File page FH

DMA1 Address High Nibble
DMA1H (FBAH - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

DMA1 Start Address [11:8]

DMA1 End Address [11:8]

DMA1 Start/Current Address Low Byte
DMA1START (FBBH - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

DMA1 Start Address [7:0]

DMA1 End Address Low Byte
DMA1END (FBCH - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

DMA1 End Address [7:0]

DMA_ADC Address
DMAA_ADDR (FBDH - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Reserved

DMA_ADC Address

Table 9. Reset Sources and Resulting Reset Type

Operating Mode	Reset Source	Reset Type
NORMAL or HALT modes	Power-On Reset/Voltage Brownout	system reset
	Watchdog Timer time-out when configured for Reset	system reset
	$\overline{\text{RESET}}$ pin assertion	system reset
	On-Chip Debugger initiated Reset (OCDCTL[0] set to 1)	system reset except the On-Chip Debugger is unaffected by the reset
STOP mode	Power-On Reset/Voltage Brownout	system reset
	$\overline{\text{RESET}}$ pin assertion	system reset
	DBG pin driven Low	system reset

Power-On Reset

Each device in the 64K Series contains an internal Power-On Reset 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 66 cycles of the Watchdog Timer oscillator. After the POR counter times out, the XTAL Counter is enabled to count a total of 16 system clock pulses. The devices are held in the Reset state until both the POR Counter and XTAL counter have timed out. After the 64K Series devices exit the Power-On Reset state, the eZ8 CPU fetches the Reset vector. Following Power-On Reset, the POR status bit in the Watchdog Timer Control (WDTCTL) register is set to 1.

Figure 8 displays Power-On Reset operation. For the POR threshold voltage (V_{POR}), see Electrical Characteristics on page 215.

Stop Mode Recovery Using a GPIO Port Pin Transition HALT

Each of the GPIO Port pins may be configured as a Stop Mode Recovery input source. On any GPIO pin enabled as a Stop Mode Recovery source, a change in the input pin value (from High to Low or from Low to High) initiates Stop Mode Recovery. The GPIO Stop Mode Recovery signals are filtered to reject pulses less than 10 ns (typical) in duration. In the Watchdog Timer Control register, the STOP bit is set to 1.

Caution: *In STOP mode, the GPIO Port Input Data registers (PxIN) are disabled. The Port Input Data registers record the Port transition only if the signal stays on the Port pin through the end of the Stop Mode Recovery delay. Thus, short pulses on the Port pin can initiate Stop Mode Recovery without being written to the Port Input Data register or without initiating an interrupt (if enabled for that pin).*

GPIO Control Register Definitions

Four registers for each Port provide access to GPIO control, input data, and output data. Table 13 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.

Table 13. GPIO Port Registers and Sub-Registers

Port Register Mnemonic	Port Register Name
PxADDR	Port A–H Address Register (Selects sub-registers)
PxCTL	Port A–H Control Register (Provides access to sub-registers)
PxIN	Port A–H Input Data Register
PxOUT	Port A–H Output Data Register
Port Sub-Register Mnemonic	**Port Register Name**
PxDD	Data Direction
PxAF	Alternate Function
PxOC	Output Control (Open-Drain)
PxDD	High Drive Enable
PxSMRE	Stop Mode Recovery Source Enable

Port A–H Address Registers

The Port A–H Address registers select the GPIO Port functionality accessible through the Port A–H Control registers. The Port A–H Address and Control registers combine to provide access to all GPIO Port control (Table 14).

Table 14. Port A–H GPIO Address Registers (PxADDR)

BITS	7	6	5	4	3	2	1	0
FIELD	PADDR[7:0]							
RESET	00H							
R/W	R/W							
ADDR	FD0H, FD4H, FD8H, FDCH, FE0H, FE4H, FE8H, FECH							

- Executing a Trap instruction.
- Illegal Instruction trap.

Interrupt Vectors and Priority

The interrupt controller supports three levels of interrupt priority. Level 3 is the highest priority, Level 2 is the second highest priority, and Level 1 is the lowest priority. If all of the interrupts were enabled with identical interrupt priority (all as Level 2 interrupts, for example), then interrupt priority would be assigned from highest to lowest as specified in Table 23 on page 68. Level 3 interrupts always have higher priority than Level 2 interrupts which, in turn, always have higher priority than Level 1 interrupts. Within each interrupt priority level (Level 1, Level 2, or Level 3), priority is assigned as specified in Table 23 on page 68. Reset, Watchdog Timer interrupt (if enabled), and Illegal Instruction Trap always have highest priority.

Interrupt Assertion

Interrupt sources assert their interrupt requests for only a single system clock period (single pulse). When the interrupt request is acknowledged by the eZ8 CPU, the corresponding bit in the Interrupt Request register is cleared until the next interrupt occurs. Writing a 0 to the corresponding bit in the Interrupt Request register likewise clears the interrupt request.

Caution: *The following style of coding to clear bits in the Interrupt Request registers is NOT recommended. All incoming interrupts that are received between execution of the first LDX command and the last LDX command are lost.*

Poor coding style that can result in lost interrupt requests:

```
LDX r0, IRQ0
AND r0, MASK
LDX IRQ0, r0
```

To avoid missing interrupts, the following style of coding to clear bits in the Interrupt Request 0 register is recommended:

Good coding style that avoids lost interrupt requests:

```
ANDX IRQ0, MASK
```

Software Interrupt Assertion

Program code can generate interrupts directly. Writing a 1 to the desired bit in the Interrupt Request register triggers an interrupt (assuming that interrupt is enabled). When the interrupt request is acknowledged by the eZ8 CPU, the bit in the Interrupt Request register is automatically cleared to 0.

– 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 1 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 1 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.

Follow the steps below for configuring a timer for CAPTURE/COMPARE mode and initiating the count:

1. Write to the Timer Control 1 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.
5. Configure the associated GPIO port pin for the Timer Input alternate function.

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

Table 49. Watchdog Timer Reload Upper Byte Register (WDTU)

BITS	7	6	5	4	3	2	1	0
FIELD	WDTU							
RESET	1							
R/W	R/W*							
ADDR	FF1H							

Note: R/W* - Read returns the current WDT count value. Write sets the desired Reload Value.

WDTU—WDT Reload Upper Byte
Most significant byte, Bits[23:16], of the 24-bit WDT reload value.

Table 50. Watchdog Timer Reload High Byte Register (WDTH)

BITS	7	6	5	4	3	2	1	0
FIELD	WDTH							
RESET	1							
R/W	R/W*							
ADDR	FF2H							

Note: R/W* - Read returns the current WDT count value. Write sets the desired Reload Value.

WDTH—WDT Reload High Byte
Middle byte, Bits[15:8], of the 24-bit WDT reload value.

Table 51. Watchdog Timer Reload Low Byte Register (WDTL)

BITS	7	6	5	4	3	2	1	0
FIELD	WDTL							
RESET	1							
R/W	R/W*							
ADDR	FF3H							

Note: R/W* - Read returns the current WDT count value. Write sets the desired Reload Value.

WDTL—WDT Reload Low
Least significant byte, Bits[7:0], of the 24-bit WDT reload 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 desired 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 on the infrared operation, see Infrared Encoder/Decoder on page 125.

UART Transmit Data Register

Data bytes written to the UART Transmit Data register (Table 52) 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.

Table 52. UART Transmit Data Register (U*x*TXD)

BITS	7	6	5	4	3	2	1	0
FIELD	TXD							
RESET	X							
R/W	W							
ADDR	F40H and F48H							

Table 55. UART Status 1 Register (UxSTAT1)

BITS	7	6	5	4	3	2	1	0
FIELD	Reserved						NEWFRM	MPRX
RESET	0							
R/W	R				R/W		R	
ADDR	F44H and F4CH							

Reserved—Must be 0.

NEWFRM—Status bit denoting the start of a new frame. Reading the UART Receive Data register resets this bit to 0.
0 = The current byte is not the first data byte of a new frame.
1 = The current byte is the first data byte of a new frame.

MPRX—Multiprocessor Receive
Returns the value of the last multiprocessor bit received. Reading from the UART Receive Data register resets this bit to 0.

UART Control 0 and Control 1 Registers

The UART Control 0 and Control 1 registers (see Table 56 and Table 57 on page 118) configure the properties of the UART's transmit and receive operations. The UART Control registers must not been written while the UART is enabled.

Table 56. UART Control 0 Register (UxCTL0)

BITS	7	6	5	4	3	2	1	0
FIELD	TEN	REN	CTSE	PEN	PSEL	SBRK	STOP	LBEN
RESET	0							
R/W	R/W							
ADDR	F42H and F4AH							

TEN—Transmit Enable
This bit enables or disables the transmitter. The enable is also controlled by the $\overline{\text{CTS}}$ signal and the CTSE bit. If the $\overline{\text{CTS}}$ signal is low and the CTSE bit is 1, the transmitter is enabled.
0 = Transmitter disabled.
1 = Transmitter enabled.

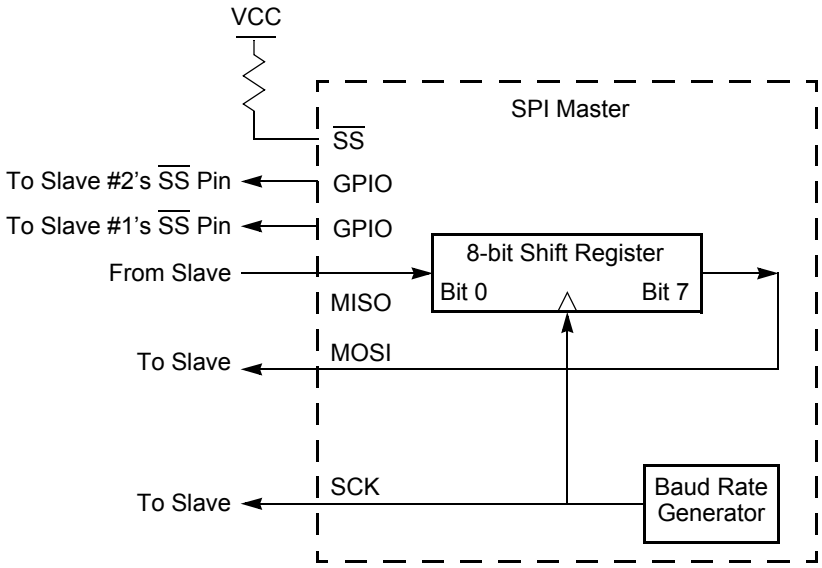

Figure 23. SPI Configured as a Master in a Single Master, Multiple Slave System

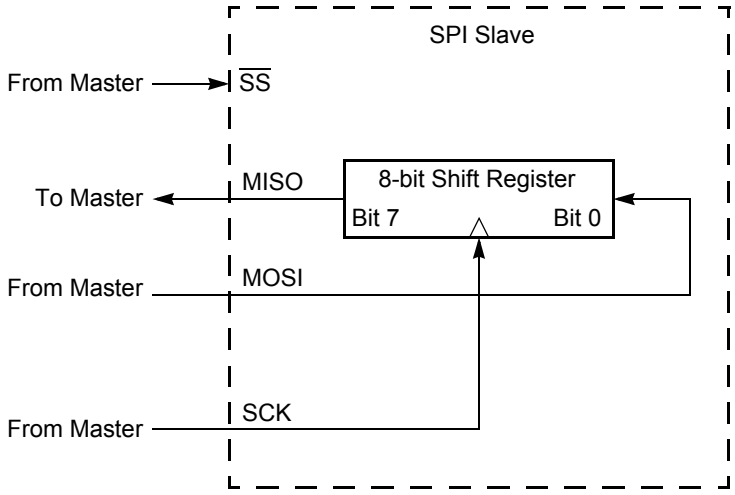

Figure 24. SPI Configured as a Slave

Operation

The SPI is a full-duplex, synchronous, character-oriented channel that supports a four-wire interface (serial clock, transmit, receive and Slave select). The SPI block consists of a transmit/receive shift register, a Baud Rate (clock) Generator and a control unit.

The Master and Slave are each capable of exchanging a character of data during a sequence of NUMBITS clock cycles (see NUMBITS field in the SPI Mode Register on page 140). In both Master and Slave SPI devices, data is shifted on one edge of the SCK and is sampled on the opposite edge where data is stable. Edge polarity is determined by the SPI phase and polarity control.

Slave Select

The active Low Slave Select ($\overline{SS}$) input signal selects a Slave SPI device. $\overline{SS}$ must be Low prior to all data communication to and from the Slave device. $\overline{SS}$ must stay Low for the full duration of each character transferred. The $\overline{SS}$ signal may stay Low during the transfer of multiple characters or may deassert between each character.

When the SPI is configured as the only Master in an SPI system, the $\overline{SS}$ pin can be set as either an input or an output. For communication between the Z8F642x family Z8R642x family device's SPI Master and external Slave devices, the $\overline{SS}$ signal, as an output, can assert the $\overline{SS}$ input pin on one of the Slave devices. Other GPIO output pins can also be employed to select external SPI Slave devices.

When the SPI is configured as one Master in a multi-master SPI system, the $\overline{SS}$ pin must be set as an input. The $\overline{SS}$ input signal on the Master must be High. If the $\overline{SS}$ signal goes Low (indicating another Master is driving the SPI bus), a Collision error Flag is set in the SPI Status register.

SPI Clock Phase and Polarity Control

The SPI supports four combinations of serial clock phase and polarity using two bits in the SPI Control register. The clock polarity bit, CLKPOL, selects an active high or active Low clock and has no effect on the transfer format. Table 62 lists the SPI Clock Phase and Polarity Operation parameters. The clock phase bit, PHASE, selects one of two fundamentally different transfer formats. For proper data transmission, the clock phase and polarity must be identical for the SPI Master and the SPI Slave. The Master always places data on the MOSI line a half-cycle before the receive clock edge (SCK signal), in order for the Slave to latch the data.

Table 62. SPI Clock Phase (PHASE) and Clock Polarity (CLKPOL) Operation

PHASE	CLKPOL	SCK Transmit Edge	SCK Receive Edge	SCK Idle State
0	0	Falling	Rising	Low
0	1	Rising	Falling	High
1	0	Rising	Falling	Low
1	1	Falling	Rising	High

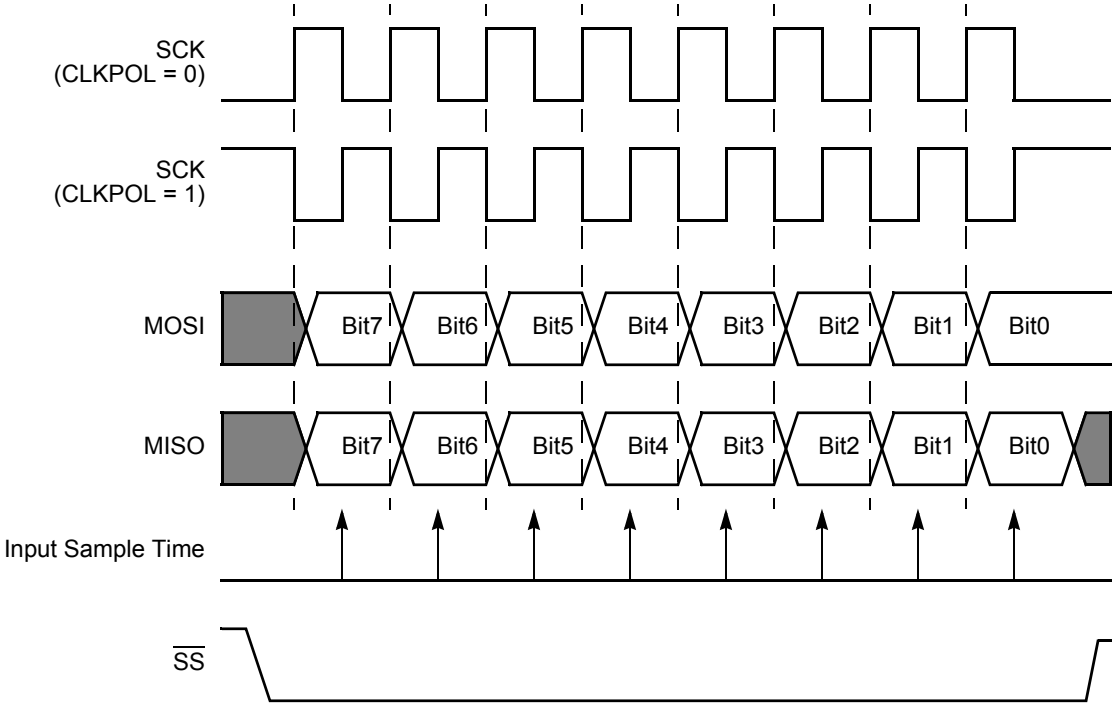

Figure 26. SPI Timing When PHASE **is 1**

Multi-Master Operation

In a multi-master SPI system, all SCK pins are tied together, all MOSI pins are tied together and all MISO pins are tied together. All SPI pins must then be configured in OPEN-DRAIN mode to prevent bus contention. At any one time, only one SPI device is configured as the Master and all other SPI devices on the bus are configured as Slaves. The Master enables a single Slave by asserting the $\overline{SS}$ pin on that Slave only. Then, the single Master drives data out its SCK and MOSI pins to the SCK and MOSI pins on the Slaves (including those which are not enabled). The enabled Slave drives data out its MISO pin to the MISO Master pin.

For a Master device operating in a multi-master system, if the $\overline{SS}$ pin is configured as an input and is driven Low by another Master, the COL bit is set to 1 in the SPI Status Register. The COL bit indicates the occurrence of a multi-master collision (mode fault error condition).

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

13. The I2C Controller shifts the data out of using the SDA signal. After the first bit is sent, the Transmit interrupt is asserted.
14. If more bytes remain to be sent, return to step 9.
15. Software responds by setting the STOP bit of the I2C Control register (or START bit to initiate a new transaction). In the STOP case, software clears the TXI bit of the I2C Control register at the same time.
16. The I2C Controller completes transmission of the data on the SDA signal.
17. The slave may either Acknowledge or Not Acknowledge the last byte. Because either the STOP or START bit is already set, the NCKI interrupt does not occur.
18. The I2C Controller sends the STOP (or RESTART) condition to the I2C bus. The STOP or START bit is cleared.

Address Only Transaction with a 10-bit Address

In the situation where software wants to determine if a slave with a 10-bit address is responding without sending or receiving data, a transaction can be done which only consists of an address phase. Figure 30 displays this 'address only' transaction to determine if a slave with 10-bit address will acknowledge. As an example, this transaction can be used after a 'write' has been done to a EEPROM to determine when the EEPROM completes its internal write operation and is once again responding to I2C transactions. If the slave does not Acknowledge the transaction can be repeated until the slave is able to Acknowledge.

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

Figure 30. 10-Bit Address Only Transaction Format

Follow the steps below for an address only transaction to a 10-bit addressed slave:

1. Software asserts the IEN bit in the I2C Control register.
2. Software asserts the TXI bit of the I2C Control register to enable Transmit interrupts.
3. The I2C interrupt asserts, because the I2C Data register is empty (TDRE = 1)
4. Software responds to the TDRE interrupt by writing the first slave address byte. The least-significant bit must be 0 for the write operation.
5. Software asserts the START bit of the I2C Control register.
6. The I2C Controller sends the START condition to the I2C slave.

Flash Sector Protect Register

The Flash Sector Protect register (Table 95) protects Flash memory sectors from being programmed or erased from user code. The Flash Sector Protect register shares its Register File address with the Page Select register. The Flash Sector protect register can be accessed only after writing the Flash Control register with 5EH.

User code can only write bits in this register to 1 (bits cannot be cleared to 0 by user code).

Table 95. Flash Sector Protect Register (FPROT)

BITS	7	6	5	4	3	2	1	0
FIELD	SECT7	SECT6	SECT5	SECT4	SECT3	SECT2	SECT1	SECT0
RESET	0							
R/W	R/W1							
ADDR	FF9H							

Note: R/W1 = Register is accessible for Read operations. Register can be written to 1 only (via user code).

SECT*n*—Sector Protect
0 = Sector *n* can be programmed or erased from user code.
1 = Sector *n* is protected and cannot be programmed or erased from user code.

* User code can only write bits from 0 to 1.

Flash Frequency High and Low Byte Registers

The Flash Frequency High and Low Byte registers (Table 96 and Table 97) combine to form a 16-bit value, FFREQ, to control timing for Flash program and erase operations. The 16-bit Flash Frequency registers must be written with the system clock frequency in kHz for Program and Erase operations. Calculate the Flash Frequency value using the following equation:

$$FFREQ[15:0] = \{FFREQH[7:0],FFREQL[7:0]\} = \frac{\text{System Clock Frequency}}{1000}$$

Caution: *Flash programming and erasure is not supported for system clock frequencies below 20 kHz, above 20 MHz, or outside of the valid operating frequency range for the device. The Flash Frequency High and Low Byte registers must be loaded with the correct value to insure proper program and erase times.*

DC Characteristics

Table 106 lists the DC characteristics of the 64K Series products. All voltages are referenced to V_{SS}, the primary system ground.

Table 106. DC Characteristics

Symbol	Parameter	T_A = –40 °C to 125 °C			Units	Conditions
		Minimum	Typical	Maximum		
V_{DD}	Supply Voltage	3.0	–	3.6	V	
V_{IL1}	Low Level Input Voltage	-0.3	–	0.3*V_{DD}	V	For all input pins except $\overline{\text{RESET}}$, DBG, XIN
V_{IL2}	Low Level Input Voltage	-0.3	–	0.2*V_{DD}	V	For $\overline{\text{RESET}}$, DBG, and XIN.
V_{IH1}	High Level Input Voltage	0.7*V_{DD}	–	5.5	V	Port A, C, D, E, F, and G pins.
V_{IH2}	High Level Input Voltage	0.7*V_{DD}	–	V_{DD}+0.3	V	Port B and H pins.
V_{IH3}	High Level Input Voltage	0.8*V_{DD}	–	V_{DD}+0.3	V	$\overline{\text{RESET}}$, DBG, and XIN pins
V_{OL1}	Low Level Output Voltage Standard Drive	–	–	0.4	V	I_{OL} = 2 mA; VDD = 3.0 V High Output Drive disabled.
V_{OH1}	High Level Output Voltage Standard Drive	2.4	–	–	V	I_{OH} = -2 mA; VDD = 3.0 V High Output Drive disabled.
V_{OL2}	Low Level Output Voltage High Drive	–	–	0.6	V	I_{OL} = 20 mA; VDD = 3.3 V High Output Drive enabled T_A = -40 °C to +70 °C
V_{OH2}	High Level Output Voltage High Drive	2.4	–	–	V	I_{OH} = -20 mA; VDD = 3.3 V High Output Drive enabled; T_A = -40 °C to +70 °C
V_{OL3}	Low Level Output Voltage High Drive	–	–	0.6	V	I_{OL} = 15 mA; VDD = 3.3 V High Output Drive enabled; T_A = +70 °C to +105 °C
V_{OH3}	High Level Output Voltage High Drive	2.4	–	–	V	I_{OH} = 15 mA; VDD = 3.3 V High Output Drive enabled; T_A = +70 °C to +105 °C
V_{RAM}	RAM Data Retention	0.7	–	–	V	
I_{IL}	Input Leakage Current	-5	–	+5	µA	V_{DD} = 3.6 V; V_{IN} = VDD or VSS[1]
I_{TL}	Tri-State Leakage Current	-5	–	+5	µA	V_{DD} = 3.6 V

UART Timing

Figure 56 and Table 120 provide timing information for UART pins for the case where the Clear To Send input pin ($\overline{CTS}$) is used for flow control. In this example, it is assumed that the Driver Enable polarity has been configured to be Active Low and is represented here by $\overline{DE}$. The $\overline{CTS}$ to $\overline{DE}$ assertion delay (T1) assumes the UART Transmit Data register has been loaded with data prior to $\overline{CTS}$ assertion.

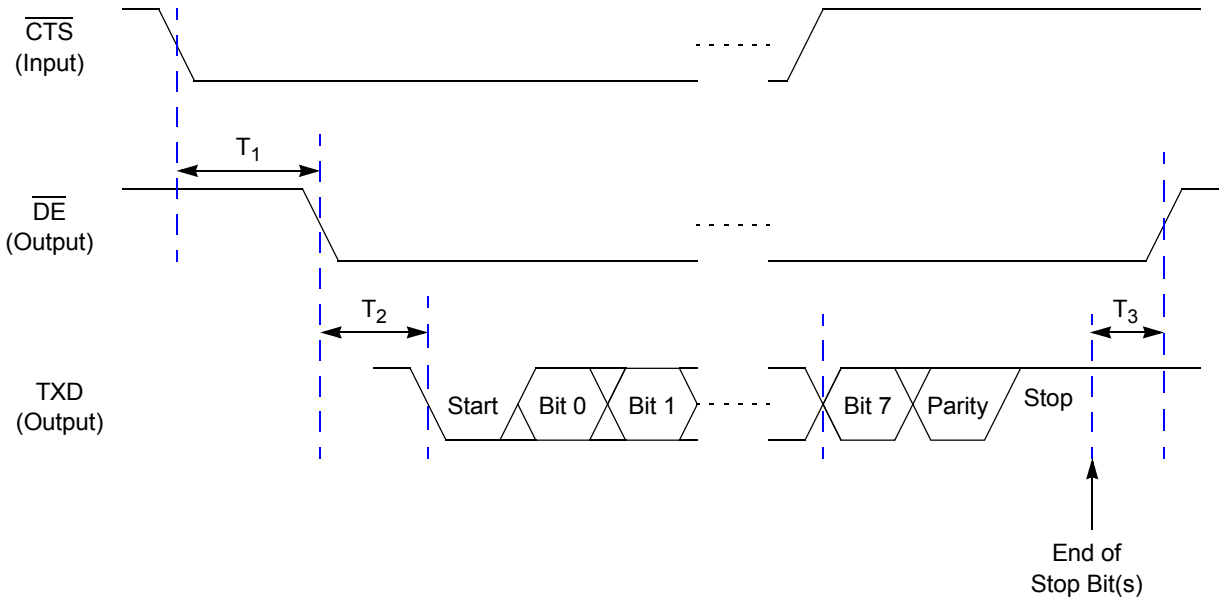

Figure 56. UART Timing with $\overline{CTS}$

Table 120. UART Timing with $\overline{CTS}$

		Delay (ns)	
Parameter	**Abbreviation**	**Minimum**	**Maximum**
T_1	$\overline{CTS}$ Fall to $\overline{DE}$ Assertion Delay	2 * XIN period	2 * XIN period + 1 Bit period
T_2	$\overline{DE}$ Assertion to TXD Falling Edge (Start) Delay	1 Bit period	1 Bit period + 1 * XIN period
T_3	End of Stop Bit(s) to $\overline{DE}$ Deassertion Delay	1 * XIN period	2 * XIN period

V

W

X

Z