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	8032
Core Size	8-Bit
Speed	40MHz
Connectivity	I²C, IrDA, SPI, UART/USART, USB
Peripherals	LVD, POR, PWM, WDT
Number of I/O	46
Program Memory Size	288KB (288K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	32K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 3.6V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	80-LQFP
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/upsd3454evb40u6

3 Hardware description

The UPSD34xx has a modular architecture built from a stacked die process. There are two dice, one is designated “MCU module” in this document, and the other is designated “PSD module” (see *Figure 4 on page 29*). In all cases, the MCU module die operates at 3.3 V with 5 V tolerant I/O. The PSD module is either a 3.3 V die or a 5 V die, depending on the UPSD34xx device as described below.

The MCU module consists of a fast 8032 core, that operates with 4 clocks per instruction cycle, and has many peripheral and system supervisor functions. The PSD module provides the 8032 with multiple memories (two Flash and one SRAM) for program and data, programmable logic for address decoding and for general-purpose logic, and additional I/O. The MCU module communicates with the PSD module through internal address and data busses (AD0 – AD15) and control signals ($\overline{RD}$, $\overline{WR}$, $\overline{PSEN}$, ALE, $\overline{RESET}$).

There are slightly different I/O characteristics for each module. I/Os for the MCU module are designated as Ports 1, 3, and 4. I/Os for the PSD module are designated as Ports A, B, C, and D.

For all 5 V UPSD34xx devices, a 3.3 V MCU module is stacked with a 5 V PSD module. In this case, a 5 V UPSD34xx device must be supplied with 3.3 V_{CC} for the MCU module and 5.0 V_{DD} for the PSD module. Ports 3 and 4 of the MCU module are 3.3 V ports with tolerance to 5 V devices (they can be directly driven by external 5 V devices and they can directly drive external 5 V devices while producing a V_{OH} of 2.4V min and V_{CC} max). Ports A, B, C, and D of the PSD module are true 5 V ports.

For all 3.3 V UPSD34xxV devices, a 3.3 V MCU module is stacked with a 3.3 V PSD module. In this case, a 3.3 V UPSD34xx device needs to be supplied with a single 3.3 V voltage source at both V_{CC} and V_{DD}. I/O pins on Ports 3 and 4 are 5 V tolerant and can be connected to external 5 V peripherals devices if desired. Ports A, B, C, and D of the PSD module are 3.3 V ports, which are not tolerant to external 5 V devices.

Refer to *Table 3* for port type and voltage source requirements.

80-pin UPSD34xx devices provide access to 8032 address, data, and control signals on external pins to connect external peripheral and memory devices. 52-pin UPSD34xx devices do not provide access to the 8032 system bus.

All non-volatile memory and configuration portions of the UPSD34xx device are programmed through the JTAG interface and no special programming voltage is needed. This same JTAG port is also used for debugging of the 8032 core at runtime providing breakpoint, single-step, display, and trace features. A non-volatile security bit may be programmed to block all access via JTAG interface for security. The security bit is defeated only by erasing the entire device, leaving the device blank and ready to use again.

Table 3. Port type and voltage source combinations

Device type	V_{CC} for MCU module	V_{DD} for PSD module	Ports 1, 3, and 4 on MCU module	Ports A, B, C, and D on PSD module
5 V: UPSD34xx	3.3 V	5.0 V	3.3 V (Ports 3 and 4 are 5 V tolerant)	5 V
3.3 V: UPSD34xxV	3.3 V	3.3 V	3.3 V (Ports 3 and 4 are 5 V tolerant)	3.3 V. NOT 5 V tolerant

Table 10. Program branching instruction set (continued)

Mnemonic(1) and use		Description	Length/cycles
CJNE	@Ri, #data, rel	Compare immediate to indirect, jump if not equal	3 byte/2 cycle
DJNZ	Rn, rel	Decrement register and jump if not zero	2 byte/2 cycle
DJNZ	direct, rel	Decrement direct byte and jump if not zero	3 byte/2 cycle

1. All mnemonics copyrighted ©Intel Corporation 1980.

Table 11. Miscellaneous instruction set

Mnemonic(1) and Use		Description	Length/Cycles
Miscellaneous			
NOP		No operation	1 byte/1 cycle

1. All mnemonics copyrighted ©Intel Corporation 1980.

Table 12. Notes on instruction set and addressing modes

Rn	Register R0 - R7 of the currently selected register bank.
direct	8-bit address for internal 8032 DATA SRAM (locations 00h - 7Fh) or SFR registers (locations 80h - FFh).
@Ri	8-bit internal 8032 SRAM (locations 00h - FFh) addressed indirectly through contents of R0 or R1.
#data	8-bit constant included within the instruction.
#data16	16-bit constant included within the instruction.
addr16	16-bit destination address used by LCALL and LJMP.
addr11	11-bit destination address used by ACALL and AJMP.
rel	Signed (two-s compliment) 8-bit offset byte.
bit	Direct addressed bit in internal 8032 DATA SRAM (locations 20h to 2Fh) or in SFR registers (88h, 90h, 98h, A8h, B0, B8h, C0h, C8h, D0h, D8h, E0h, F0h).

Table 21. IEA register bit definition (continued)

Bit	Symbol	R/W	Function
3	–	–	Reserved, do not set to logic '1.'
2	–	–	Reserved, do not set to logic '1.'
1(1)	EI^{2}C	R,W	Enable I^{2}C Interrupt
0	EUSB	R,W	Enable USB Interrupt

1. 1 = Enable Interrupt, 0 = Disable Interrupt.

Table 22. IP: interrupt priority register (SFR B8h, reset value 00h)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
–	–	PT2	PS0	PT1	PX1	PT0	PX0

Table 23. IP register bit definition

Bit	Symbol	R/W	Function
7	–	–	Reserved
6	–	–	Reserved
5(1)	PT2	R,W	Timer 2 Interrupt priority level
4(1)	PS0	R,W	UART0 Interrupt priority level
3(1)	PT1	R,W	Timer 1 Interrupt priority level
2(1)	PX1	R,W	External Interrupt INT1 priority level
1(1)	PT0	R,W	Timer 0 Interrupt priority level
0(1)	PX0	R,W	External Interrupt INT0 priority level

1. 1 = Assigns high priority level, 0 = Assigns low priority level.

Table 24. IPA: Interrupt Priority Addition register (SFR B7h, reset value 00h)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PADC	PSPI	PPCA	PS1	–	–	PI^{2}C	PUSB

Table 25. IPA register bit definition

Bit	Symbol	R/W	Function
7(1)	PADC	R,W	ADC Interrupt priority level
6(1)	PSPI	R,W	SPI Interrupt priority level
5(1)	PPCA	R,W	PCA Interrupt level
4(1)	PS1	R,W	UART1 Interrupt priority level
3	–	–	Reserved
2	–	–	Reserved
1(1)	PI^{2}C	R,W	I^{2}C Interrupt priority level
0	PUSB	R,W	USB Interrupt priority level

1. 1 = Assigns high priority level, 0 = Assigns low priority level.

Table 28. CCON0 register bit definition (continued)

Bit	Symbol	R/W	Definition
3	CPUAR	R,W	Automatic MCU Clock Recovery 0 = There is no change of CPUPS[2:0] when an interrupt occurs. 1 = Contents of CPUPS[2:0] automatically become 000b whenever any interrupt occurs.
2:0	CPUPS	R,W	MCUCLK Pre-Scaler 000b: $f_{MCU} = f_{OSC}$ (Default after reset) 001b: $f_{MCU} = f_{OSC}/2$ 010b: $f_{MCU} = f_{OSC}/4$ 011b: $f_{MCU} = f_{OSC}/8$ 100b: $f_{MCU} = f_{OSC}/16$ 101b: $f_{MCU} = f_{OSC}/32$ 110b: $f_{MCU} = f_{OSC}/1024$ 111b: $f_{MCU} = f_{OSC}/2048$

Table 29. CCON1 PLL control register (SFR FAh, reset value 00h)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PLLM[3:0]				PLLD[3:0]			

Table 30. CCON1 register bit definition

Bit	Symbol	R/W	Definition
7:4	PLLM[3:0]	R,W	Lower 4 bits of the 5-bit PLLM[4:0] Multiplier (Default after reset: PLLM = 00h) PLLM[4] is in the CCON0 register.
3:0	PLLD[3:0]	R,W	4-bit PLL Divider (Default after reset: PLLD = 0h)

A[10:8] and the remaining pins can be configured for other functions such as generating chip selects to the external devices.

Figure 19. Connecting external devices using ports A and B for address AD[15:0]

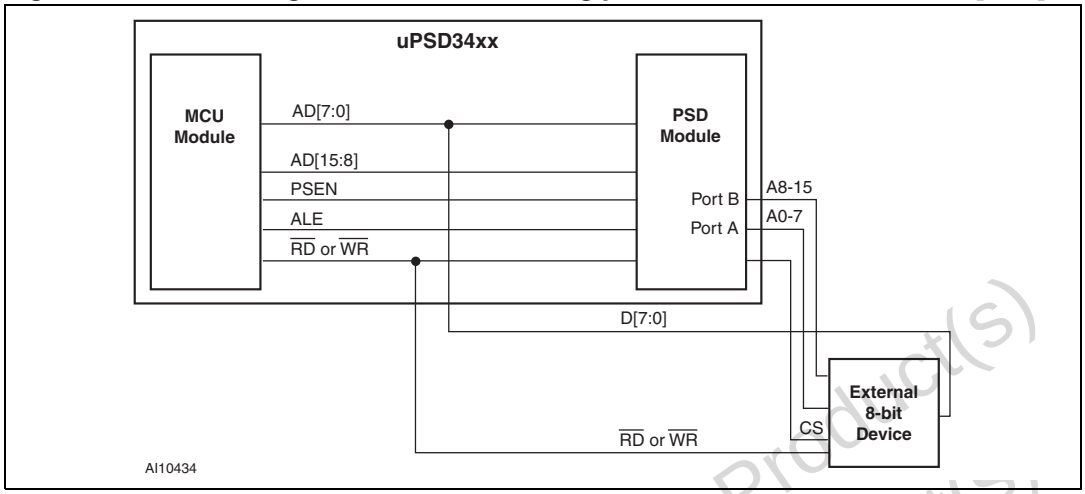
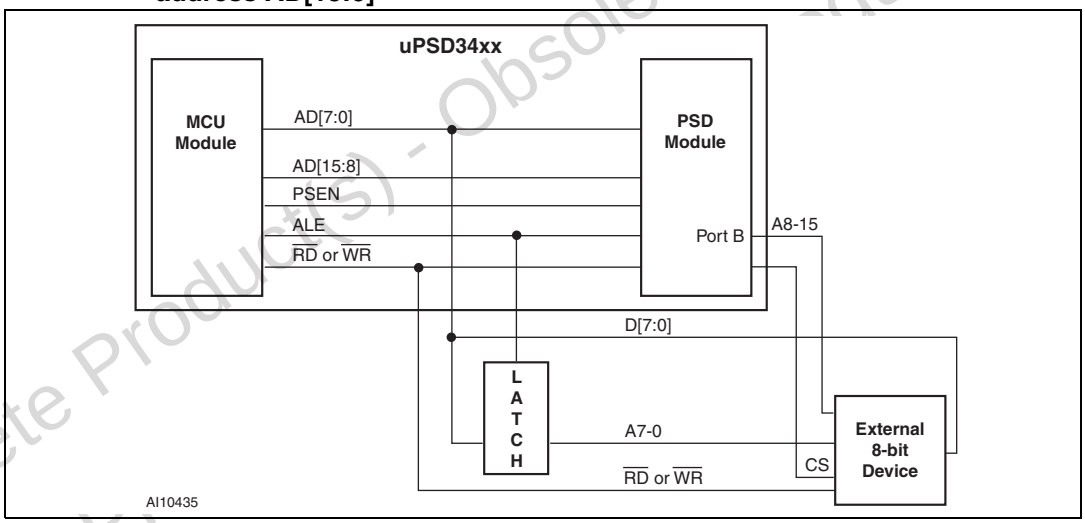


Figure 20. Connecting external devices using port A and an external latch for address AD[15:0]



18.4 Programmable bus timing

The length of the bus cycles are user programmable at run time. The number of MCU_CLK periods in a bus cycle can be specified in the SFR register named BUSCON (see *Table 49 on page 88*). By default, the BUSCON register is loaded with long bus cycle times (6 MCU_CLK periods) after a reset condition. It is important that the post-reset initialization firmware sets the bus cycle times appropriately to get the most performance, according to *Table 51 on page 90*. Keep in mind that the PSD module has a faster Turbo mode (default) and a slower but less power consuming Non-Turbo mode. The bus cycle times must be programmed in BUSCON to optimize for each mode as shown in *Table 51*. See *Section 28.5: PSD module detailed operation on page 207* for more details.

20.4 SFR, TMOD

Timer 0 and Timer 1 have four modes of operation controlled by the SFR named TMOD (*Table 58*).

20.5 Timer 0 and Timer 1 operating modes

The “Timer” or “Counter” function is selected by the C/$\overline{T}$ control bits in TMOD. The four operating modes are selected by bit-pairs M[1:0] in TMOD. Modes 0, 1, and 2 are the same for both Timer/Counters. Mode 3 is different.

20.5.1 Mode 0

Putting either Timer/Counter into Mode 0 makes it an 8-bit Counter with a divide-by-32 pre-scaler. *Figure 24* shows Mode 0 operation as it applies to Timer 1 (same applies to Timer 0).

In this mode, the Timer register is configured as a 13-bit register. As the count rolls over from all '1s' to all '0s,' it sets the Timer Interrupt flag TF1. The counted input is enabled to the Timer when TR1 = 1 and either GATE = 0 or EXTINT1 = 1. (Setting GATE = 1 allows the Timer to be controlled by external input pin, EXTINT1, to facilitate pulse width measurements). TR1 is a control bit in the SFR, TCON. GATE is a bit in the SFR, TMOD.

The 13-bit register consists of all 8 bits of TH1 and the lower 5 bits of TL1. The upper 3 bits of TL1 are indeterminate and should be ignored. Setting the run flag, TR1, does not clear the registers.

Mode 0 operation is the same for the Timer 0 as for Timer 1. Substitute TR0, TF0, C0, TL0, TH0, and EXTINT0 for the corresponding Timer 1 signals in *Figure 24*. There are two different GATE Bits, one for Timer 1 and one for Timer 0.

20.5.2 Mode 1

Mode 1 is the same as Mode 0, except that the Timer register is being run with all 16 bits.

20.5.3 Mode 2

Mode 2 configures the Timer register as an 8-bit Counter (TL1) with automatic reload, as shown in *Figure 25 on page 100*. Overflow from TL1 not only sets TF1, but also reloads TL1 with the contents of TH1, which is preset with firmware. The reload leaves TH1 unchanged. Mode 2 operation is the same for Timer/Counter 0.

20.5.4 Mode 3

Timer 1 in Mode 3 simply holds its count. The effect is the same as setting TR1 = 0.

Timer 0 in Mode 3 establishes TL0 and TH0 as two separate counters. The logic for Mode 3 on Timer 0 is shown in *Figure 26 on page 100*. TL0 uses the Timer 0 control Bits: C/$\overline{T}$, GATE, TR0, and TF0, as well as the pin EXTINT0. TH0 is locked into a timer function (counting at a rate of 1/12 f_{OSC}) and takes over the use of TR1 and TF1 from Timer 1. Thus, TH0 now controls the “Timer 1“ interrupt flag.

Mode 3 is provided for applications requiring an extra 8-bit timer on the counter (see *Figure 26 on page 100*). With Timer 0 in Mode 3, a UPSD34xx device can look like it has three Timer/Counters (not including the PCA). When Timer 0 is in Mode 3, Timer 1 can be

Table 69. Commonly used baud rates generated from timer 1 (continued)

UART mode	f_{OSC} MHz	Desired baud rate	Resultant baud rate	Baud rate deviation	SMOD bit in PCON	Timer 1		
						C/$\overline{T}$ Bit in TMOD	Timer mode in TMOD	TH1 reload value (hex)
Modes 1 or 3	3.6864	19200	19200	0	1	0	2	FF
Modes 1 or 3	3.6864	9600	9600	0	1	0	2	FE
Modes 1 or 3	1.8432	9600	9600	0	1	0	2	FF
Modes 1 or 3	1.8432	4800	4800	0	1	0	2	FE

21.4 More about UART mode 0

Refer to the block diagram in *Figure 30 on page 113*, and timing diagram in *Figure 31 on page 113*.

Transmission is initiated by any instruction which writes to the SFR named SBUF. At the end of a write operation to SBUF, a 1 is loaded into the 9th position of the transmit shift register and tells the TX Control unit to begin a transmission. Transmission begins on the following MCU machine cycle, when the “SEND” signal is active in *Figure 31*.

SEND enables the output of the shift register to the alternate function on the port containing pin RxD, and also enables the SHIFT CLOCK signal to the alternate function on the port containing the pin, TxD. At the end of each SHIFT CLOCK in which SEND is active, the contents of the transmit shift register are shifted to the right one position.

As data bits shift out to the right, zeros come in from the left. When the MSB of the data byte is at the output position of the shift register, then the '1' that was initially loaded into the 9th position, is just to the left of the MSB, and all positions to the left of that contain zeros. This condition flags the TX Control unit to do one last shift, then deactivate SEND, and then set the interrupt flag TI. Both of these actions occur at S1P1.

Reception is initiated by the condition REN = 1 and RI = 0. At the end of the next MCU machine cycle, the RX Control unit writes the bits 11111110 to the receive shift register, and in the next clock phase activates RECEIVE. RECEIVE enables the SHIFT CLOCK signal to the alternate function on the port containing the pin, TxD. Each pulse of SHIFT CLOCK moves the contents of the receive shift register one position to the left while RECEIVE is active. The value that comes in from the right is the value that was sampled at the RxD pin. As data bits come in from the right, 1s shift out to the left. When the 0 that was initially loaded into the right-most position arrives at the left-most position in the shift register, it flags the RX Control unit to do one last shift, and then it loads SBUF. After this, RECEIVE is cleared, and the receive interrupt flag RI is set.

The interface may operate as either a Master or a Slave within a given application, controlled by firmware writing to SFRs.

By default after a reset, the I²C interface is in Master Receiver mode, and the SDA/P3.6 and SCL/P3.7 pins default to GPIO input mode, high impedance, so there is no I²C bus interference. Before using the I²C interface, it must be initialized by firmware, and the pins must be configured. This is discussed in *Section 23.13: I2C operating sequences on page 135*.

23.4 Bus arbitration

A Master device always samples the I²C bus to ensure a bus line is high whenever that Master is asserting a logic 1. If the line is low at that time, the Master recognizes another device is overriding its own transmission.

A Master may start a transfer only if the I²C bus is not busy. However, it is possible that two or more Masters may generate a Start condition simultaneously. In this case, arbitration takes place on the SDA line each time SCL is high. The Master that first senses that its bus sample does not correspond to what it is driving (SDA line is low while it is asserting a high) will immediately change from Master-Transmitter to Slave-Receiver mode. The arbitration process can carry on for many bit times if both Masters are addressing the same Slave device, and will continue into the data bits if both Masters are trying to be Master-Transmitter. It is also possible for arbitration to carry on into the acknowledge bits if both Masters are trying to be Master-Receiver. Because address and data information on the bus is determined by the winning Master, no information is lost during the arbitration process.

23.5 Clock synchronization

Clock synchronization is used to synchronize arbitrating Masters, or used as a handshake by a devices to slow down the data transfer.

23.5.1 Clock sync during arbitration

During bus arbitration between competing Masters, Master_X, with the longest low period on SCL, will force Master_Y to wait until Master_X finishes its low period before Master_Y proceeds to assert its high period on SCL. At this point, both Masters begin asserting their high period on SCL simultaneously, and the Master with the shortest high period will be the first to drive SCL for the next low period. In this scheme, the Master with the longest low SCL period paces low times, and the Master with the shortest high SCL period paces the high times, making synchronized arbitration possible.

23.5.2 Clock sync during handshaking

This allows receivers in different devices to handle various transfer rates, either at the byte-level, or bit-level.

At the byte-level, a device may pause the transfer between bytes by holding SCL low to have time to store the latest received byte or fetch the next byte to transmit.

At the bit-level, a Slave device may extend the low period of SCL by holding it low. Thus the speed of any Master device will adapt to the internal operation of the Slave.

23.6 General call address

A General Call (GC) occurs when a Master-Transmitter initiates a transfer containing a Slave address of 0000000b, and the R/$\overline{W}$ bit is logic 0. All Slave devices capable of responding to this broadcast message will acknowledge the GC simultaneously and then behave as a Slave-Receiver. The next byte transmitted by the Master will be accepted and acknowledged by all Slaves capable of handling the special data bytes. A Slave that cannot handle one of these data bytes must ignore it by not acknowledging it. The I^2C specification lists the possible meanings of the special bytes that follow the first GC address byte, and the actions to be taken by the Slave device(s) upon receiving them. A common use of the GC by a Master is to dynamically assign device addresses to Slave devices on the bus capable of a programmable device address.

The UPSD34xx can generate a GC as a Master-Transmitter, and it can receive a GC as a Slave. When receiving a GC address (00h), an interrupt will be generated so firmware may respond to the special GC data bytes if desired.

23.7 Serial I/O engine (SIOE)

At the heart of the I^2C interface is the hardware SIOE, shown in *Figure 42*. The SIOE automatically handles low-level I^2C bus protocol (data shifting, handshaking, arbitration, clock generation and synchronization) and it is controlled and monitored by five SFRs.

The five SFRs shown in *Figure 42* are:

- S1CON - Interface control (*Table 75 on page 129*)
- S1STA - Interface status (*Table 78 on page 130*)
- S1DAT - Data shift register (*Table 80 on page 132*)
- S1ADR - Device address (*Table 82 on page 132*)
- S1SETUP - Sampling rate (*Table 84 on page 133*)

Enable individual I2C interrupt and set priority
- SFR IEA.I2C = 1
- SFR IPA.I2C = 1 if high priority is desired

Set the Device address for Slave mode
- SFR S1ADR = XXh, desired address

Enable SIOE (as Slave) to return an ACK signal
- SFR S1CON.AA = 1

Master-Transmitter

Disable all interrupts
- SFR IE.EA = 0

Set pointer to global data xmit buffer, set count
- *xmit_buf = *pointer to data
- buf_length = number of bytes to xmit

Set global variables to indicate Master-Xmitter
- I2C_master = 1, I2C_xmitter = 1

Disable Master from returning an ACK
- SFR S1CON.AA = 0

Enable I2C SIOE
- SFR S1CON.INI1 = 1

Transmit Address and R/W bit = 0 to Slave
- Is bus not busy? (SFR S1STA.BBUSY = 0?)

<If busy, then test until not busy>
- SFR S1DAT[7:0] = Load Slave Address & FEh
- SFR S1CON.STA = 1, send Start on bus

<bus transmission begins>

Enable All Interrupts and go do something else
- SFR IE.EA = 1

Master-Receiver

Disable all interrupts
- SFR IE.EA = 0

Set pointer to global data recv buffer, set count
- *recv_buf = *pointer to data
- buf_length = number of bytes to recv

Set global variables to indicate Master-Xmitter
- I2C_master = 1, I2C_xmitter = 0

Disable Master from returning an ACK
- SFR S1CON.AA = 0

16, 20, all the way up to 252. For example, if SPICLKD contains 24h, SPICLK has the frequency of PERIH_CLK divided by 36 decimal.

The SPICLK frequency must be set low enough to allow the MCU time to read received data bytes without loosing data. This is dependent upon many things, including the crystal frequency of the MCU and the efficiency of the SPI firmware.

24.6 Dynamic control

At runtime, bits in registers SPICON0, SPICON1, and SPISTAT are managed by firmware for dynamic control over the SPI interface. The bits Transmitter Enable (TE) and Receiver Enable (RE) when set will allow transmitting and receiving respectively. If TE is disabled, both transmitting and receiving are disabled because SPICLK is driven to constant output logic ‘0’ (when SPO = 0) or logic '1' (when SPO = 1).

When the SSEL Bit is set, the SPISEL pin will drive to logic '0' (active) to select a connected slave device at the appropriate time before the first data bit of a byte is transmitted, and SPISEL will automatically return to logic '1' (inactive) after transmitting the eight bit of data, as shown in *Figure 46 on page 145*. SPISEL will continue to automatically toggle this way for each byte data transmission while the SSEL bit is set by firmware. When the SSEL Bit is cleared, the SPISEL pin will drive to constant logic '1' and stay that way (after a transmission in progress completes).

The Interrupt Enable Bits (TEIE, RORIE,TIE, and RIE) when set, will allow an SPI interrupt to be generated to the MCU upon the occurrence of the condition enabled by these bits. Firmware must read the four corresponding flags in the SPISTAT register to determine the specific cause of interrupt. These flags are automatically cleared when firmware reads the SPISTAT register.

Table 89. SPICON0: control register 0 (SFR D6h, reset value 00h)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
–	TE	RE	SPIEN	SSEL	FLSB	SBO	–

Table 90. SPICON0 register bit definition

Bit	Symbol	R/W	Definition
7	–	–	Reserved
6	TE	RW	Transmitter Enable 0 = Transmitter is disabled 1 = Transmitter is enabled
5	RE	RW	Receiver Enable 0 = Receiver is disabled 1 = Receiver is enabled
4	SPIEN	RW	SPI Enable 0 = Entire SPI Interface is disabled 1 = Entire SPI Interface is enabled

- USB IN FIFO NAK interrupt flag (UIF3)

  The USB IN FIFO NAK Interrupt Flag register (see *Table 118*) contains flags that indicate when an IN Endpoint FIFO is not ready. The Endpoint FIFO is not ready when data has not been loaded into its FIFO and the USIZE register has not been written to (writing to the USIZE register puts the FIFO in a “ready” to send data state). Until the FIFO is ready, the SIE will continue to NAK all IN requests to the respective Endpoint. Once set, firmware must clear the flag by writing a '0' to the appropriate bit. When FIFOs are paired, only the odd numbered FIFO Interrupt Flags are active.

Table 118. USB IN FIFO NAK interrupt flag (UIF3 0EBh, reset value 00h)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
–	–	–	NAK4F	NAK3F	NAK2F	NAK1F	NAK0F

Table 119. UIF3 register bit definition

Bit	Symbol	R/W	Definition
7	–	–	Reserved
6	–	–	Reserved
5	–	–	Reserved
4	NAK4F	R/W	Endpoint 4 IN FIFO NAK Interrupt flag This bit is set when the SIE responded to an IN request with a NAK since the FIFO was not ready.
3	NAK3F	R/W	Endpoint 3 IN FIFO NAK Interrupt flag This bit is set when the SIE responded to an IN request with a NAK since the FIFO was not ready.
2	NAK2F	R/W	Endpoint 2 IN FIFO NAK Interrupt flag This bit is set when the SIE responded to an IN request with a NAK since the FIFO was not ready.
1	NAK1F	R/W	Endpoint 1 IN FIFO NAK Interrupt flag This bit is set when the SIE responded to an IN request with a NAK since the FIFO was not ready.
0	NAK0F	R/W	Endpoint 0 IN FIFO NAK Interrupt flag This bit is set when the SIE responded to an IN request with a NAK since the FIFO was not ready.

- USB endpoint0 status (USTA)

  The USB Endpoint0 Status register (see *Table 122*) provides the status for events that occur on the USB that are directed to endpoint0.

Table 122. USB endpoint0 status (USTA 0EDh, reset value 00h)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
–	–	–	–	RCVT	SETUP	IN	OUT

Table 123. USTA register bit definition

Bit	Symbol	R/W	Definition
7	–	–	Reserved
6	–	–	Reserved
5	–	–	Reserved
4	–	–	Reserved
3	RCVT	R	Received Data Toggle Bit This bit indicates the toggle bit of the received data packet: 0 = Data0, and 1 = Data1
2	SETUP	R/W	SETUP Token Detect Bit This bit is set when Endpoint0 receives a SETUP token. This bit is not cleared when Endpoint0 receives an IN or OUT token following the SETUP token that set this bit. This bit is cleared by software or a reset.
1	IN	R	IN Token Detect Bit This bit is set when Endpoint0 receives an IN token. This bit is cleared when Endpoint0 receives a SETUP or OUT token.
0	OUT	R	OUT Token Detect Bit This bit is set when Endpoint0 receives an OUT token. This bit is cleared when Endpoint0 receives a SETUP or IN token.

Caution: Disabling and enabling the USB SIE using the USBEN bit in the UCTL register clears the RCVT, IN, and OUT bits in this register.

- USB endpoint select register (USEL)

  Endpoints share the same XDATA space for FIFOs as well as the same SFR addresses for Control and FIFO Valid Size registers. The USB endpoint select register (see *Table 124*) is used to select the desired direction and endpoint that is accessed when reading or writing to the FIFO XDATA address space. This register is also used to select the direction and Endpoint when accessing the USB endpoint control register.

Table 124. USB endpoint select register (USEL 0EFh, reset value 00h)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DIR	–	–	–	–	EP[2]	EP[1]	EP[0]

Table 125. USEL register bit definition

Bit	Symbol	R/W	Definition
7	DIR	R/W	FIFO's Direction Select Bit: 0: IN FIFO select 1: OUT FIFO select
6	–	–	Reserved
5	–	–	Reserved
4	–	–	Reserved
3	–	–	Reserved
2:0	EP	R/W	Endpoint Selects Bits: 0: Endpoint0 1: Endpoint1 2: Endpoint2 3: Endpoint3 4: Endpoint4

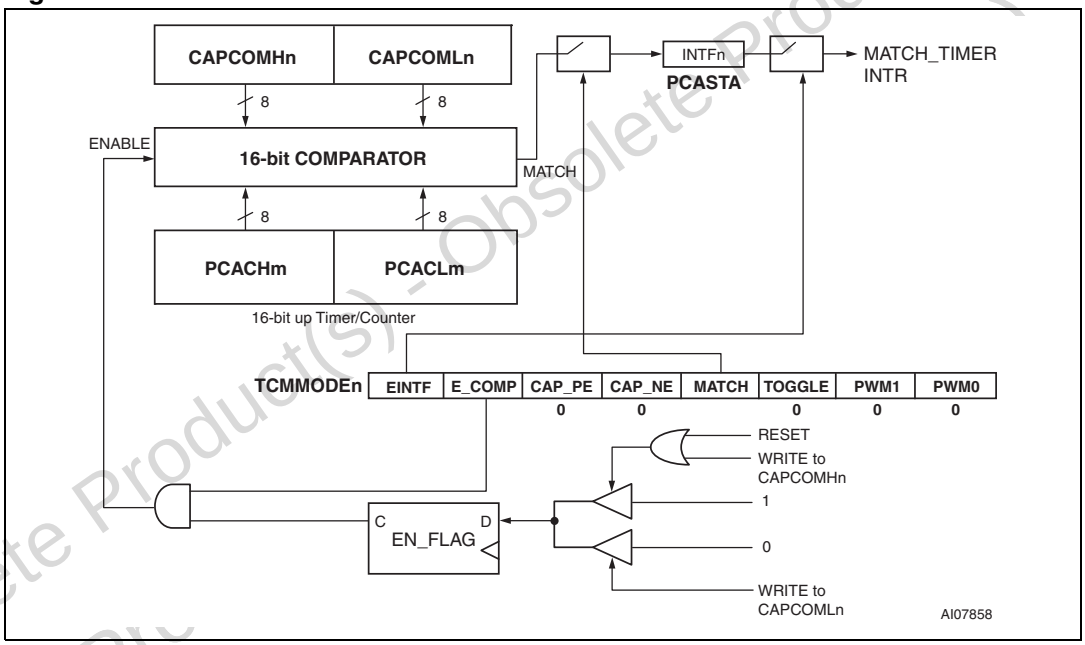
27.6 Toggle mode

In this mode, the user writes a value to the TCM's CAPCOM registers and enables the comparator. When there is a match with the Counter output, the output of the TCM pin toggles. This mode is a simple extension of the Timer Mode.

27.7 PWM mode - (x8), fixed frequency

In this mode, one or all the TCM's can be configured to have a fixed frequency PWM output on the port pins. The PWM frequency depends on when the low byte of the Counter overflows (modulo 256). The duty cycle of each TCM module can be specified in the CAPCOMHn register. When the PCA_Counter_L value is equal to or greater than the value in CAPCOMLn, the PWM output is switched to a high state. When the PCA_Counter_L register overflows, the content in CAPCOMHn is loaded to CAPCOMLn and a new PWM pulse starts.

Figure 58. Timer mode



1. m = 0: n = 0, 1, or 2
   m = 1: n = 3, 4, or 5

27.10 PWM mode - fixed frequency, 10-bit

The 10-bit PWM logic requires that all 3 TCMs in PCA0 or PCA1 operate in the same 10-bit PWM mode. The 10-bit PWM operates in a similar manner as the 16-bit PWM, except the PCACHm and PCACLm counters are reconfigured as 10-bit counters. The CAPCOMHn and CAPCOMLn registers become 10-bit registers.

PWM duty cycle of each TCM module can be specified in the 10-bit CAPCOMHn and CAPCOMLn registers. When the 10-bit PCA counter is equal or greater than the values in the 10-bit registers CAPCOMHn and CAPCOMLn, the PWM output switches to a high state. When the 10-bit PCA counter overflows, the PWM pin is switched to a logic low and starts the next PWM pulse.

The most-significant 6 bits in the PCACHm counter and CAPCOMH register are “Don’t cares” and have no effect on the PWM generation.

27.11 Writing to capture/compare registers

When writing a 16-bit value to the PCA Capture/Compare registers, the low byte should always be written first. Writing to CAPCOMLn clears the E_COMP Bit to '0'; writing to CAPCOMHn sets E_COMP to '1' the largest duty cycle is 100% (CAPCOMHn CAPCOMLn = 0000h), and the smallest duty cycle is 0.0015% (CAPCOMHn CAPCOMLn = FFFFh). A 0% duty cycle may be generated by clearing the E_COMP Bit to ‘0’.

27.12 Control register bit definition

Each PCA has its own PCA_CONFIGn, and each module within the PCA block has its own TCM_Mode register which defines the operation of that module (see *Table 148 on page 187* through *Table 150 on page 188*). There is one PCA_STATUS register that covers both PCA0 and PCA1 (see *Table 152 on page 188*).

Table 148. PCA0 control register PCACON0 (SFR 0A4h, reset value 00h)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EN-ALL	EN_PCA	EOVFI	PCAIDLE	–	–	CLK_SEL[1:0]	

Table 149. PCA0 register bit definition

Bit	Symbol	Function
7	EN-ALL	0 = No impact on TCM modules 1 = Enable both PCA counters simultaneously (override the EN_PCA Bits) This bit is to start the two 16-bit counters in the PCA. For customers who want 5 PWM, for example, this bit can start all of the PWM outputs.
6	EN_PCA	0 = PCA counter is disabled 1 = PCA counter is enabled EN_PCA Counter Run Control Bit. Set with software to turn the PCA counter on. Must be cleared with software to turn the PCA counter off.
5	EOVFI	1 = Enable Counter Overflow Interrupt if overflow flag (OVF) is set
4	PCAIDLE	0 = PCA operates when CPU is in Idle Mode 1 = PCA stops running when CPU is in Idle Mode

Note: The 8032 data bus, D0 - D7, does not route directly to PLD inputs. Instead, the 8032 data bus has indirect access to the GPLD (not the DPLD) when the 8032 reads and writes the OMC and IMC registers within csiop address space.

28.5.26 Turbo bit and PLDs

The PLDs can minimize power consumption by going to standby after ALL the PLD inputs remain unchanged for an extended time (about 70ns). When the Turbo Bit is set to logic one (Bit 3 of the csiop PMMR0 register), Turbo mode is turned off and then this automatic standby mode is achieved. Turning off Turbo mode increases propagation delays while reducing power consumption. The default state of the Turbo Bit is logic zero, meaning Turbo mode is on. Additionally, four bits are available in the csiop PMMR0 and PMMR2 registers to block the 8032 bus control signals ($\overline{RD}$, $\overline{WR}$, $\overline{PSEN}$, ALE) from entering the PLDs. This reduces power consumption and can be used only when these 8032 control signals are not used in PLD logic equations. See *Section 28.5.52: Power management on page 248*.

Table 167. DPLD and GPLD inputs

Input source	Input name	Number of signals
8032 address bus	A0-A15	16
8032 bus control signals	$\overline{PSEN}$, $\overline{RD}$, $\overline{WR}$, ALE	4
Reset from MCU module	$\overline{RESET}$	1
Power-Down from Auto-Power Down counter	PDN	1
PortA input macrocells (80-pin devices only)	PA0-PA7	8
PortB input macrocells	PB0-PB7	8
PortC input macrocells	PC2, PC3, PC4, PC7	4
Port D inputs (52-pin devices have only PD1)	PD1, PD2	2
Page register	PGR0-PGR7	8
Macrocell OMC bank AB Feedback	MCELLAB FB0-7	8
Macrocell OMC bank BC Feedback	MCELLBC FB0-7	8
Flash memory status bit	Ready/$\overline{Busy}$	1

Note: If a particular OMC output is specified as an internal node and not specified as a port pin output in PSDsoft Express, then the port pin that is associated with that OMC can be used for other I/O functions.

Figure 80. Simple PLD logic example

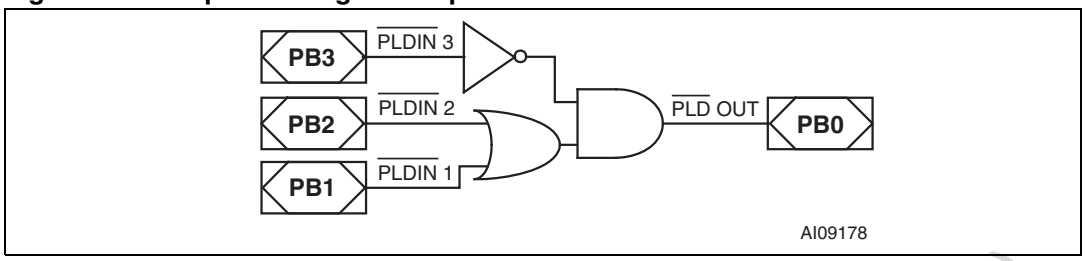
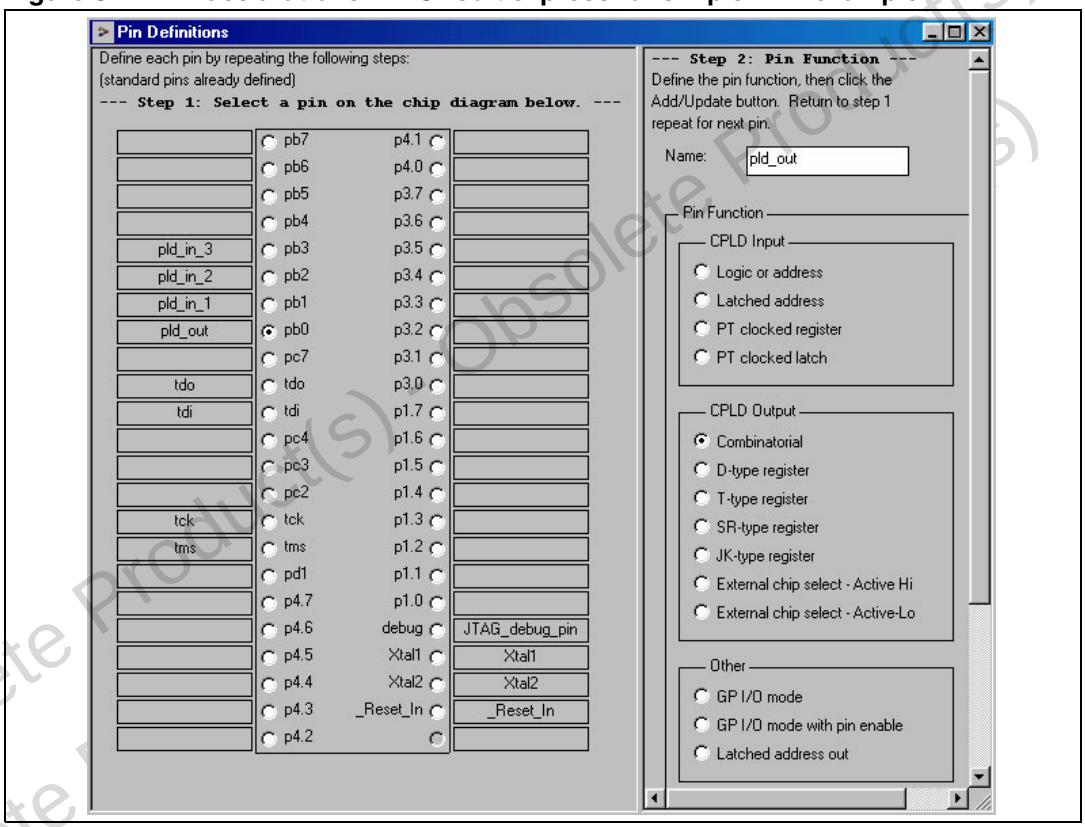


Figure 81. Pin declarations in PSDsoft express for simple PLD example



28.6.5 6-pin JTAG ISP (optional)

The optional signals TSTAT and $\overline{\text{TERR}}$ are programming status flags that can reduce programming time by as much as 30% compared to 4-pin JTAG because this status information does not have to be scanned out of the device serially. TSTAT and $\overline{\text{TERR}}$ must be used as a pair for 6-pin JTAG operation.

- TSTAT (pin PC3) indicates when programming of a single Flash location is complete. Logic 1 = Ready, Logic 0 = busy.
- $\overline{\text{TERR}}$ (pin PC4) indicates if there was a Flash programming error. Logic 1 = no error, Logic 0 = error.

The pin functions for PC3 and PC4 must be selected as “Dedicated JTAG - TSTAT” and “Dedicated JTAG - $\overline{\text{TERR}}$” in PSDsoft Express to enable 6-pin JTAG ISP.

No 8032 firmware is needed to use 6-pin ISP because all ISP functions are controlled from the external JTAG program/test equipment.

TSTAT and $\overline{\text{TERR}}$ are functional only when JTAG ISP operations are occurring, which means they are non-functional during JTAG debugging of the 8032 on the MCU module.

Programming times vary depending on the number of locations to be programmed and the JTAG programming equipment, but typical JTAG ISP programming times are 10 to 25 seconds using 6-pin JTAG. The signals TSTAT and $\overline{\text{TERR}}$ are not included in the IEEE 1149.1 specification.

Figure 92 on page 261 shows recommended connections on a circuit board to a JTAG program/test tool using 6-pin JTAG. It is required to connect the $\overline{\text{RST}}$ output signal from the JTAG program/test equipment to the $\overline{\text{RESET\_IN}}$ input on the UPSD34xx. The $\overline{\text{RST}}$ signal is driven by the equipment with an Open Drain driver, allowing other sources (like a push button) to drive $\overline{\text{RESET\_IN}}$ without conflict.

Note: The recommended pull-up resistors and decoupling capacitor are illustrated in Figure 92.