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	160KB (160K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	8K 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/upsd3433ev-40u6

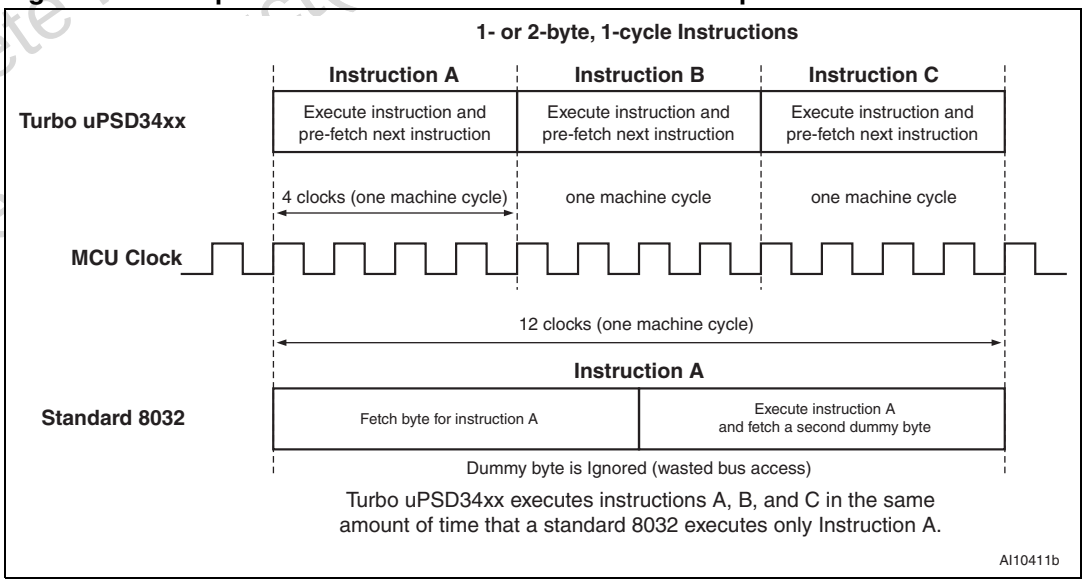
5 8032 MCU core performance enhancements

Before describing performance features of the UPSD34xx, let us first look at standard 8032 architecture. The clock source for the 8032 MCU creates a basic unit of timing called a machine-cycle, which is a period of 12 clocks for standard 8032 MCUs. The instruction set for traditional 8032 MCUs consists of 1, 2, and 3 byte instructions that execute in different combinations of 1, 2, or 4 machine-cycles. For example, there are one-byte instructions that execute in one machine-cycle (12 clocks), one-byte instructions that execute in four machine-cycles (48 clocks), two-byte, two-cycle instructions (24 clocks), and so on. In addition, standard 8032 architecture will fetch two bytes from program memory on almost every machine-cycle, regardless if it needs them or not (dummy fetch). This means for one-byte, one-cycle instructions, the second byte is ignored. These one-byte, one-cycle instructions account for half of the 8032's instructions (126 out of 255 opcodes). There are inefficiencies due to wasted bus cycles and idle bus times that can be eliminated.

The UPSD34xx 8032 MCU core offers increased performance in a number of ways, while keeping the exact same instruction set as the standard 8032 (all opcodes, the number of bytes per instruction, and the native number a machine-cycles per instruction are identical to the original 8032). The first way performance is boosted is by reducing the machine-cycle period to just 4 MCU clocks as compared to 12 MCU clocks in a standard 8032. This shortened machine-cycle improves the instruction rate for one- or two-byte, one-cycle instructions by a factor of three (*Figure 6 on page 33*) compared to standard 8051 architectures, and significantly improves performance of multiple-cycle instruction types.

The example in *Figure 6 on page 33* shows a continuous execution stream of one- or two-byte, one-cycle instructions. The 5 V UPSD34xx will yield 10 MIPS peak performance in this case while operating at 40 MHz clock rate. In a typical application however, the effective performance will be lower since programs do not use only one-cycle instructions, but special techniques are implemented in the UPSD34xx to keep the effective MIPS rate as close as possible to the peak MIPS rate at all times. This is accomplished with an instruction pre-fetch queue (PFQ), a branch cache (BC), and a 16-bit program memory bus as shown in *Figure 7 on page 34*.

Figure 6. Comparison of UPSD34xx with standard 8032 performance



CAPCOML4, CAPCOMH4, TCMMODE4, CAPCOML5, CAPCOMH5, TCMMODE5, PWMF0, PMWF1

- SPI interface registers
  SPICLKD, SPISTAT, SPITDR, SPIRDR, SPICON0, SPICON1
- I^2C interface registers
  S1SETUP, S1CON, S1STA, S1DAT, S1ADR
- Analog to digital converter registers
  ACON, ADCPS, ADAT0, ADAT1
- IrDA interface register
  IRDACON
- USB interface registers
  UADDR, UPAIR, WE0-3, UIF0-3, UCTL, USTA, USEL, UCON, USIZE, UBASEH, UBASEL, USCI, USCV

Table 5. SFR memory map with direct address and reset value

SFR addr (hex)	SFR name	Bit name and <bit address>								Reset value (hex)	Reg. descr. with link
		7	6	5	4	3	2	1	0		
80	RESERVED										
81	SP	SP[7:0]								07	*Section 7.1*
82	DPL	DPL[7:0]								00	*Section 7.2*
83	DPH	DPH[7:0]								00	
84	RESERVED										
85	DPTC	–	AT	–	–	–	DPSEL[2:0]			00	*Table 13*
86	DPTM	–	–	–	–	MD1[1:0]		MD0[1:0]		00	*Table 15*
87	PCON	SMOD0	SMOD1	–	POR	RCLK1	TCLK1	PD	IDLE	00	*Table 33*
88(1)	TCON	TF1 <8Fh>	TR1 <8Eh>	TF0 <8Dh>	TR0 <8Ch>	IE1 <8Bh>	IT1 <8Ah>	IE0 <89h>	IT0 <88h>	00	*Table 56*
89	TMOD	GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0	00	*Table 58*
8A	TL0	TL0[7:0]								00	*Section 20.1*
8B	TL1	TL1[7:0]								00	
8C	TH0	TH0[7:0]								00	
8D	TH1	TH1[7:0]								00	
8E	P1SFS0	P1SFS0[7:0]								00	*Table 43*
8F	P1SFS1	P1SFS1[7:0]								00	*Table 44*
90(1)	P1	P1.7 <97h>	P1.6 <96h>	P1.5 <95h>	P1.4 <94h>	P1.3 <93h>	P1.2 <92h>	P1.1 <91h>	P1.0 <90h>	FF	*Table 35*
91	P3SFS	P3SFS[7:0]								00	*Table 41*
92	P4SFS0	P4SFS0[7:0]								00	*Table 46*
93	P4SFS1	P4SFS1[7:0]								00	*Table 47*

18 MCU bus interface

The MCU module has a programmable bus interface which is a modified 8032 bus with 16 multiplexed address and data lines. The bus supports four types of data transfer (16- or 8-bit), each transfer is to/from a memory location external to the MCU module:

- Code Fetch cycle using the PSEN signal: fetch a 16-bit code word for filling the pre-fetch queue. The CPU fetches a code byte from the PFQ for execution;
- Code Read cycle using PSEN: read a 16-bit code word using the MOVC (Move Constant) instruction. The code word is routed directly to the CPU and by-pass the PFQ;
- XDATA Read cycle using the RD signal: read a data byte using the MOVX (Move eXternal) instruction; and
- XDATA Write cycle using the WR signal: write a data byte using the MOVX instruction

18.1 PSEN bus cycles

In a PSEN bus cycle, the MCU module fetches the instruction from the 16-bit program memory in the PSD module. The multiplexed address/data bus AD[15:0] is connected to the PSD module for 16-bit data transfer. The UPSD34xx does not support external PSEN cycles and cannot fetch instruction from other external program memory devices.

18.2 READ or WRITE bus cycles

In an XDATA READ or WRITE bus cycle, the MCU's multiplexed AD[15:0] bus is connected to the PSD module, but only the lower bytes AD[7:0] are used for the 8-bit data transfer. The AD[7:0] lines are also connected to pins in the 80-pin package for accessing external devices. If the high address byte A[15:8] is needed for external devices, Port B in the PSD module can be configured to provide the latched A[15:8] address outputs.

18.3 Connecting external devices to the MCU bus

The UPSD34xx supports 8-bit only external I/O or Data memory devices. The READ and WRITE data transfer is carried out on the AD[7:0] bus which is available in the 80-pin package. The address lines can be brought out to the external devices in one of three ways:

1. Configure Ports B and A of the PSD module in Address Output mode, as shown in *Figure 19*;
2. Use Port B together with an external latch, as shown in *Figure 20 on page 87*. The external latch latches the low address byte from the AD[7:0] bus with the ALE signal.This configuration is for design where Port A is needed for CPLD functions; and
3. Configure the microcell in the CPLD to output any address line to any of the CPLD output pins. This is the most flexible implementation but requires the use of CPLD resources.

Ports A and B in the PSD module can be configured in the PSDsoft to provide latched MCU address A[7:0] and A[15:8] (see *Section 28.5: PSD module detailed operation on page 207* for details on how to enable Address Output mode). The latched address outputs on the ports are pin configurable. For example, Port B pins PB[2:0] can be enabled to provide

Table 50. BUSCON register bit definition

Bit	Symbol	R/W	Definition
7	EPFQ	R,W	Enable pre-fetch queue 0 = PFQ is disabled 1 = PFQ is enabled (default)
6	EBC	R,W	Enable branch cache 0 = BC is disabled 1 = BC is enabled (default)
5:4	WRW[1:0]	R,W	$\overline{WR}$ Wait, number of MCU_CLK periods for $\overline{WR}$ write bus cycle during any MOVX instruction 00b: 4 clock periods 01b: 5 clock periods 10b: 6 clock periods (default) 11b: 7 clock periods
3:2	RDW[1:0]	R,W	$\overline{RD}$ Wait, number of MCU_CLK periods for $\overline{RD}$ read bus cycle during any MOVX instruction 00b: 4 clock periods 01b: 5 clock periods 10b: 6 clock periods (default) 11b: 7 clock periods
1:0	CW[1:0]	R,W	Code Wait, number of MCU_CLK periods for $\overline{PSEN}$ read bus cycle during any code byte fetch or during any MOVC code byte read instruction. Periods will increase with PFQ stall 00b: 3 clock periods - exception, for MOVC instructions this setting results 4 clock periods 01b: 4 clock periods 10b: 5 clock periods 11b: 6 clock periods (default)

Table 51. Number of MCU_CLK periods required to optimize bus transfer rate

MCU clock frequency, MCU_CLK (f_{MCU})	CW[1:0] Clk periods		RDW[1:0] Clk periods		WRW[1:0] Clk periods	
	3.3 V(1)	5 V(1)	3.3 V(1)	5 V(1)	3.3 V(1)	5 V(1)
40 MHz, Turbo mode PSD(2)	5	4	5	4	5	4
40 MHz, Non-turbo mode PSD	6	5	6	5	6	5
36 MHz, Turbo mode PSD	5	4	5	4	5	4
36 MHz, Non-turbo mode PSD	6	4	6	4	6	4
32 MHz, Turbo mode PSD	5	4	5	4	5	4
32 MHz, Non-turbo mode PSD	5	4	5	4	5	4
28 MHz, Turbo mode PSD	4	3	4	4	4	4
28 MHz, Non-turbo mode PSD	5	4	5	4	5	4
24MHz, Turbo mode PSD	4	3	4	4	4	4
24MHz, Non-turbo mode PSD	4	3	4	4	4	4
20 MHz and below, Turbo mode PSD	3	3	4	4	4	4
20 MHz and below, Non-turbo mode PSD	3	3	4	4	4	4

1. V_{DD} of the PSD module.
2. “Turbo mode PSD” means that the PSD module is in the faster, Turbo mode (default condition). A PSD module in Non-Turbo mode is slower, but consumes less current. See PSD module section, titled “PLD Non-Turbo mode” for details.

Figure 24. Timer/counter mode 0: 13-bit counter

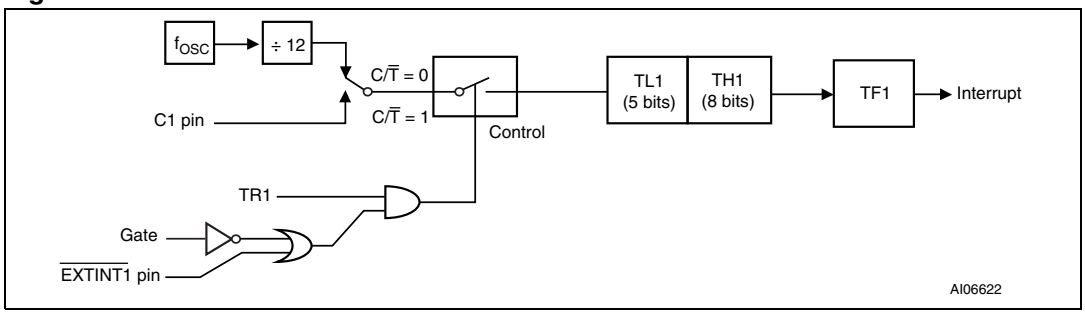


Figure 25. Timer/counter mode 2: 8-bit Auto-reload

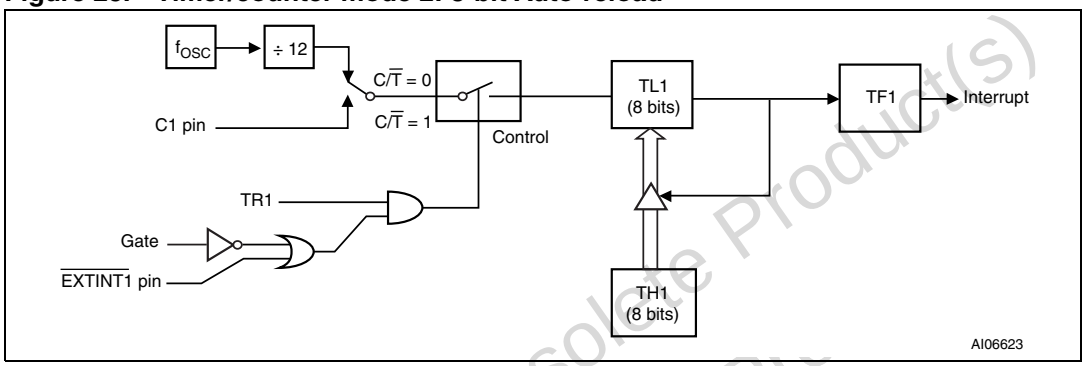
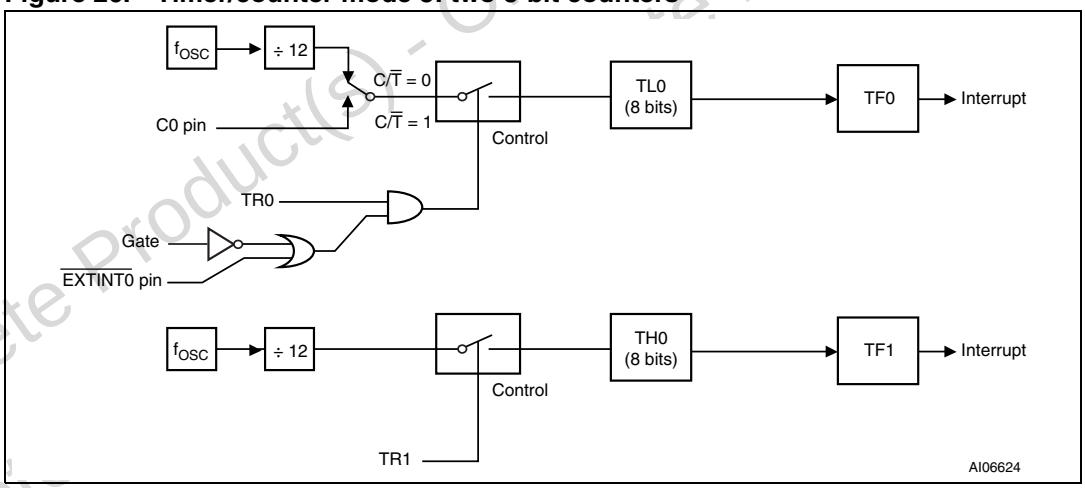


Figure 26. Timer/counter mode 3: two 8-bit counters



20.6 Timer 2

Timer 2 can operate as either an event timer or as an event counter. This is selected by the bit C/$\overline{T2}$ in the SFR named, T2CON (*Table 60 on page 101*). Timer 2 has three operating modes selected by bits in T2CON, according to *Table 62 on page 102*. The three modes are:

- Capture mode
- Auto re-load mode
- Baud rate generator mode

21.1.3 Mode 2

Mode 2 provides asynchronous, full-duplex communication using a total of 11 bits per data byte. Data is transmitted through TxD and received through RxD with: a Start Bit (logic '0'); eight data bits (LSB first); a programmable 9th data bit; and a Stop Bit (logic '1'). Upon Transmit, the 9th data bit (from bit TB8 in SCON) can be assigned the value of '0' or '1.' Or, for example, the Parity Bit (P, in the PSW) could be moved into TB8. Upon receive, the 9th data bit goes into RB8 in SCON, while the Stop Bit is ignored. The baud rate is programmable to either 1/32 or 1/64 of f_{OSC}.

21.1.4 Mode 3

Mode 3 is the same as Mode 2 in all respects except the baud rate is variable like it is in Mode 1.

In all four modes, transmission is initiated by any instruction that uses SBUF as a destination register. Reception is initiated in Mode 0 by the condition RI = 0 and REN = 1. Reception is initiated in the other modes by the incoming Start Bit if REN = 1.

Table 64. UART operating modes

Mode	Synchronization	Bits of SFR, SCON		Baud clock	Data bits	Start/stop bits	See Figure
		SM0	SM1				
0	Synchronous	0	0	$f_{OSC}/12$	8	None	*Figure 30 on page 113*
1	Asynchronous	0	1	Timer 1 or Timer 2 Overflow	8	1 Start, 1 Stop	*Figure 32 on page 115*
2	Asynchronous	1	0	$f_{OSC}/32$ or $f_{OSC}/64$	9	1 Start, 1 Stop	*Figure 34 on page 117*
3	Asynchronous	1	1	Timer 1 or Timer 2 Overflow	9	1 Start, 1 Stop	*Figure 36 on page 118*

21.1.5 Multiprocessor communications

Modes 2 and 3 have a special provision for multiprocessor communications. In these modes, 9 data bits are received. The 9th one goes into bit RB8, then comes a stop bit. The port can be programmed such that when the stop bit is received, the UART interrupt will be activated only if bit RB8 = 1. This feature is enabled by setting bit SM2 in SCON. A way to use this feature in multi-processor systems is as follows: When the master processor wants to transmit a block of data to one of several slaves, it first sends out an address byte which identifies the target slave. An address byte differs from a data byte in that the 9th bit is 1 in an address byte and 0 in a data byte. With SM2 = 1, no slave will be interrupted by a data byte. An address byte, however, will interrupt all slaves, so that each slave can examine the received byte and see if it is being addressed. The addressed slave will clear its SM2 bit and prepare to receive the data bytes that will be coming. The slaves that were not being addressed leave their SM2 bits set and go on about their business, ignoring the coming data bytes.

SM2 has no effect in Mode 0, and in Mode 1, SM2 can be used to check the validity of the stop bit. In a Mode 1 reception, if SM2 = 1, the receive interrupt will not be activated unless a valid stop bit is received.

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.

Table 79. S1STA register bit definition

Bit	Symbol	R/W	Function
7	GC	R	General Call flag GC = 1 if the General Call address of 00h was received when SIOE is in Slave mode, and GC is cleared by a Start or Stop condition on the bus. If the SIOE is in Master mode when GC = 1, the Bus Lost condition exists, and BLOST = 1.
6	STOP	R	STOP flag STOP = 1 while SIOE detects a Stop condition on the bus when in Master or Slave mode.
5	INTR	R,W	Interrupt flag INTR is set to 1 by any of the five I²C interrupt conditions listed above. INTR must be cleared by firmware.
4	TX_MODE	R	Transmission Mode flag TX_MODE = 1 whenever the SIOE is in Master-Transmitter or Slave-Transmitter mode. TX_MODE = 0 when SIOE is in any receiver mode.
3	BBUSY	R	Bus Busy flag BBUSY = 1 when the I²C bus is in use. BBUSY is set by the SIOE when a Start condition exists on the bus and BBUSY is cleared by a Stop condition.
2	BLOST	R	Bus Lost flag BLOST is set when the SIOE is in Master mode and it looses the arbitration process to another Master device on the bus.
1	ACK_RESP	R	Not Acknowledge Response flag While SIOE is in Transmitter mode: – After SIOE sends a byte, $\overline{\text{ACK\_RESP}}$ = 1 whenever the external I²C device receives the byte, but that device does NOT assert an ackowledge signal (external device asserted a high on SDA during the acknowledge bit-time). – After SIOE sends a byte, $\overline{\text{ACK\_RESP}}$ = 0 whenever the external I²C device receives the byte, and that device DOES assert an ackowledge signal (external device drove a low on SDA during the acknowledge bit-time) *Note: If SIOE is in Master-Transmitter mode, and $\overline{\text{ACK\_RESP}}$ = 1 due to a Slave-Transmitter not sending an Acknowledge, a Stop condition will not automatically be generated by the SIOE. The Stop condition must be generated with S1CON.STO = 1.*
0	SLV	R	Slave Mode flag SLV = 1 when the SIOE is in Slave mode. SLV = 0 when the SIOE is in Master mode (default).

23.10 I²C data shift register (S1DAT)

The S1ADR register (*Table 80*) holds a byte of serial data to be transmitted or it holds a serial byte that has just been received. The MCU may access S1DAT while the SIOE is not in the process of shifting a byte (the INTR flag indicates shifting is complete).

Table 86. Number of I²C bus samples taken after 1-to-0 transition on SDA (Start condition)

Contents of S1SETUP		Resulting value for S1SETUP	Resulting number of samples taken after 1-to-0 on SDA line
SS_EN bit	SMPL_SET[6:0]		
0	XXXXXXXb	00h (default)	1
1	0000000b	80h	1
1	0000001b	81h	2
1	0000010b	82h	3
...	...	...	...
1	0001011b	8Bh	12
1	0010111b	97h	24
...	...	...	...
1	1111111b	FFh	128

Table 87. Start condition hold time

I²C bus speed	Range of I²C clock speed (f_{SCL})	Minimum Start condition hold time (t_{HLDSTA})
Standard	Up to 100 kHz	4000 ns
Fast	101 kHz to 400 kHz	600 ns
High	401 kHz to 833 kHz(1)	160 ns

1. 833KHz is maximum for UPSD34xx devices.

Table 88 provides recommended settings for S1SETUP based on various combinations of f_{OSC} and f_{SCL}. Note that the "Total Sample Period" times in *Table 87 on page 134* are typically slightly less than the minimum Start condition hold time, t_{HLDSTA} for a given I²C bus speed.

Important note: The SCL bit rate f_{SCL} must first be determined by bits CR[2:0] in the SFR S1CON before a value is chosen for SMPL_SET[6:0] in the SFR S1SETUP.

Table 88. S1SETUP examples for various I²C bus speeds and oscillator frequencies

I²C bus speed, f_{SCL}	Parameter	Oscillator frequency, f_{OSC}				
		6 MHz	12 MHz	24 MHz	33 MHz	40 MHz
Standard	Recommended S1SETUP value	93h	A7h	CFh	EEh	FFh
	Number of samples	20	40	80	111	128
	Time between samples	166.6 ns	83.3 ns	41.6 ns	30 ns	25 ns
	Total sampled period	3332 ns	3332 ns	3332 ns	3333 ns	3200 ns

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

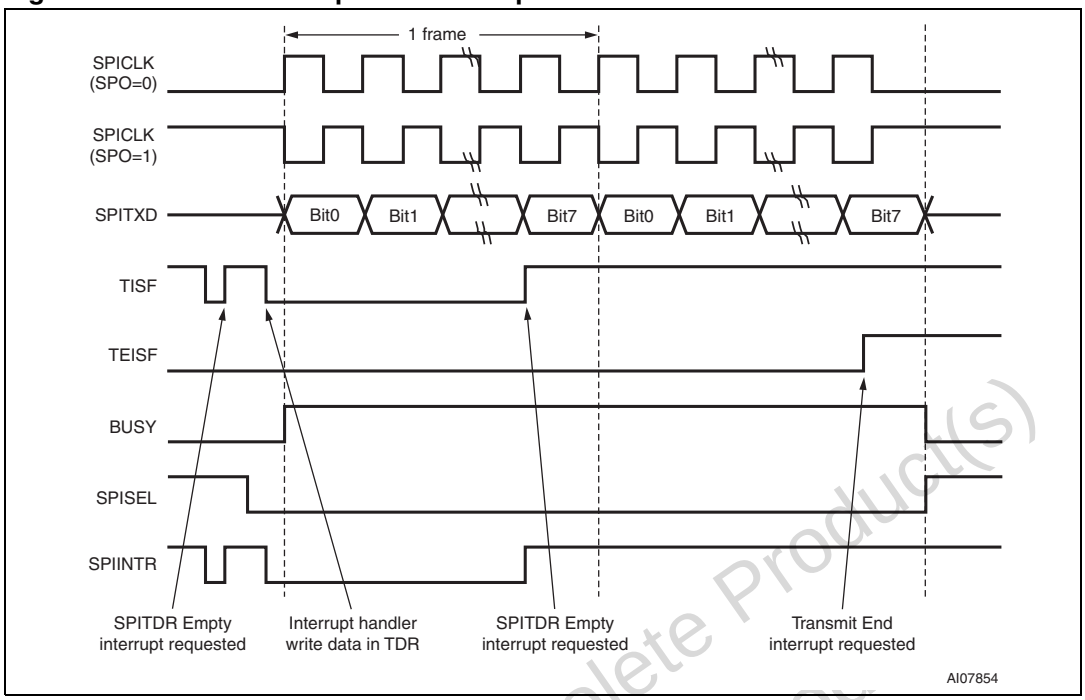
Disable all interrupts
- SFR IE.EA = 0

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

Figure 46. SPI transmit operation example



24.4 SPI SFR registers

Six SFR registers control the SPI interface:

- SPICON0 (*Table 89*) for interface control
- SPICON1 (*Table 91*) for interrupt control
- SPITDR (SFR D4h, Write only) holds byte to transmit
- SPIRDR (SFR D5h, Read only) holds byte received
- SPICLKD (*Table 93*) for clock divider
- SPISTAT (*Table 95 on page 149*) holds interface status

The SPI interface functional block diagram (*Figure 47*) shows these six SFRs. Both the transmit and receive data paths are double-buffered, meaning that continuous transmitting or receiving (back-to-back transfer) is possible by reading from SPIRDR or writing data to SPITDR while shifting is taking place. There are a number of flags in the SPISTAT register that indicate when it is full or empty to assist the 8032 MCU in data flow management. When enabled, these status flags will cause an interrupt to the MCU.

memory contents through its 8-bit data bus even while the security bit is set. The 8032 can read the status of the security bit at run-time (but it cannot change it) by reading the csiop register defined in *Table 166*.

Table 165. Main Flash memory protection register definition (address = csiop + offset C0h)(1)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Sec7_Prot	Sec6_Prot	Sec5_Prot	Sec4_Prot	Sec3_Prot	Sec2_Prot	Sec1_Prot	Sec0_Prot

1. *Bit definitions:*
Sec<i>_Prot 1 = Flash memory sector <i> is write protected, 0 = Flash memory sector <i> is not write protected.

Table 166. Secondary Flash memory protection/security register definition (csiop + offset C2h)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3(1)	Bit 2(1)	Bit 1(1)	Bit 0(1)
Security_Bit (2)	not used	not used	not used	Sec3_Prot	Sec2_Prot	Sec1_Prot	Sec0_Prot

1. *Sec<i>_Prot 1 = Flash memory sector <i> is write protected, 0 = Flash memory sector <i> is not write protected.*
2. Security_Bit = 1, device is secured, 0 = not secured.

28.5.25 PLDs

The PSD module contains two PLDs: the Decode PLD (DPLD), and the General PLD (GPLD), as shown in *Figure 73 on page 221*. Both PLDs are fed by a common PLD input signal bus, and additionally, the GPLD is connected to the 8032 data bus.

PLD logic is specified using PSDsoft Express and programmed into the PSD module using the JTAG ISP channel. PLD logic is non-volatile and available at power-up. PLDs may not be programmed by the 8032. The PLDs have selectable levels of performance and power consumption.

The DPLD performs address decoding, and generates select signals for internal and external components, such as memory, registers, and I/O ports. The DPLD can generate External Chip-Select (ECS1-ECS2) signals on Port D.

The GPLD can be used for logic functions, such as loadable counters and shift registers, state machines, encoding and decoding logic. These logic functions can be constructed from a combination of 16 Output Macrocells (OMC), 20 Input Macrocells (IMC), and the AND-OR Array.

Routing of the 16 OMCs outputs can be divided between pins on three Ports A, B, or C by the OMC Allocator as shown in *Figure 77 on page 227*. Eight of the 16 OMCs that can be routed to pins on Port A or Port B and are named MCELLAB0-MCELLAB7. The other eight OMCs to be routed to pins on Port B or Port C and are named MCELLBC0-MCELLBC7. This routing depends on the pin number assignments that are specified in PSDsoft Express for “PLD Outputs” in the Pin Definition section. OMC outputs can also be routed internally (not to pins) used as buried nodes to create shifters, counters, etc.

The AND-OR Array is used to form product terms. These product terms are configured from the logic definitions entered in PSDsoft Express. A PLD Input Bus consisting of 69 signals is connected to both PLDs. Input signals are shown in *Table 167*, both the true and compliment versions of each of these signals are available at inputs to each PLD.

28.5.60 PLD blocking bits

Blocking specific signals from entering the PLDs using bits of the csiop PMMR registers can further reduce PLD AC current consumption by lowering the effective composite frequency of inputs to the PLDs.

28.5.61 Blocking 8032 bus control signals

When the 8032 is active on the MCU module, four bus control signals ($\overline{RD}$, $\overline{WR}$, $\overline{PSEN}$, and ALE) are constantly transitioning to manage 8032 bus traffic. Each time one of these signals has a transition from logic ’1’ to '0,' or 0 to '1,' it will wake up the PLDs if operating in non-Turbo mode, or when in Turbo mode it will cause the affected PLD gates to draw current. If equations in the DPLD or GPLD do not use the signals $\overline{RD}$, $\overline{WR}$, $\overline{PSEN}$, or ALE then these signals can be blocked which will reduce the AC current component substantially. These bus control signals are rarely used in DPLD equations because they are routed in silicon directly to the memory arrays of the PSD module, bypassing the PLDs. For example, it is NOT necessary to qualify a memory chip select signal with an MCU write strobe, such as “fs0 = address range & !WR_”. Only “fs0 = address range” is needed.

Each of the 8032 bus control signals may be blocked individually by writing to Bits 2, 3, 4, and 5 of the PMMR2 register shown in *Table 201 on page 249*. Blocking any of these four bus control signals only prevents them from reaching the PLDs, but they will always go to the memories directly.

However, sometimes it is necessary to use these 8032 bus control signals in the GPLD when creating interface signals to external I/O peripherals. But it is still possible to save power by dynamically unblocking the bus signals before reading/writing the external device, then blocking the signals after the communication is complete.

The user can also block an input signal coming from pin PC7 to the PLD input bus if desired by writing to Bit 6 of PMMR2.

28.5.62 Blocking common clock, CLKIN

The input CLKIN (from pin PD1) can be blocked to reduce current consumption. CLKIN is used as a common clock input to all OMC flip-flips, it is a general input to the PLD input bus, and it is used to clock the APD counter. In PSDsoft Express, the function of pin PD1 must be specified as “Common Clock Input, CLKIN” before programming the device with JTAG to get the CLKIN function.

Bit 4 of PMMR0 can be set to logic ’1’ to block CLKIN from reaching the PLD input bus, but CLKIN will still reach the APD counter.

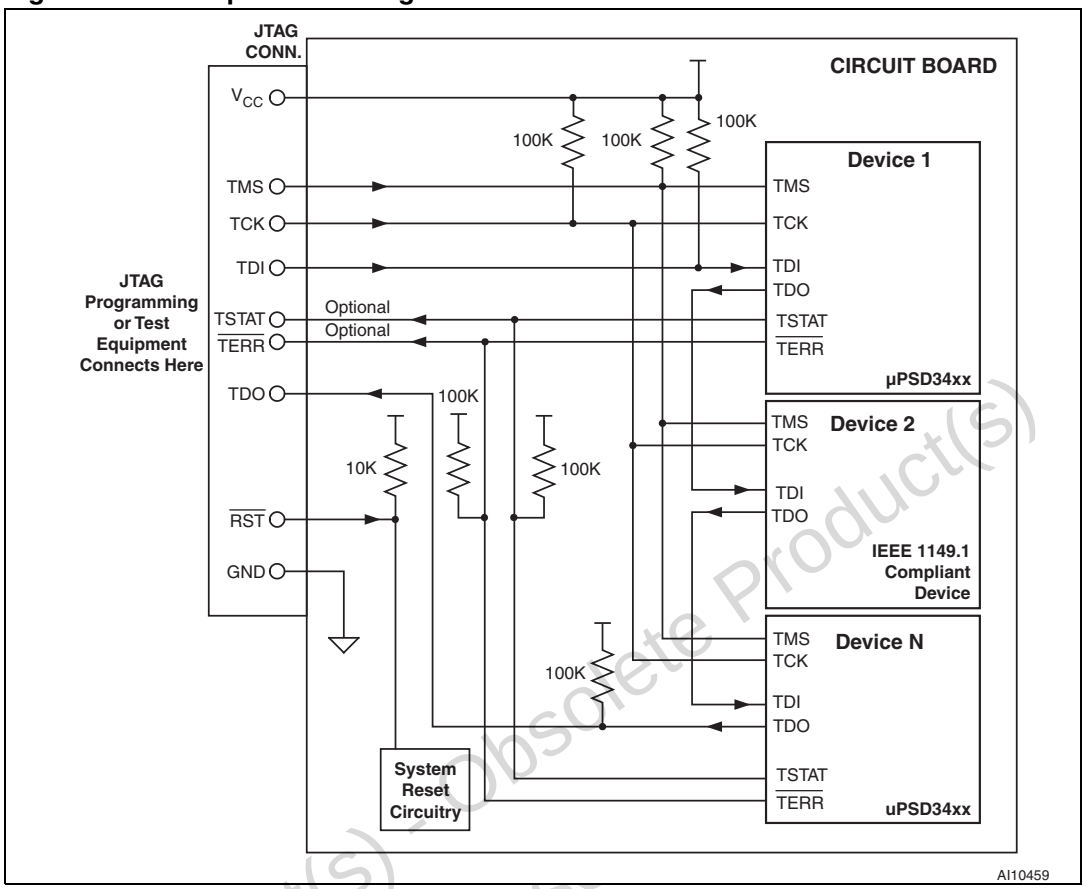
Bit 5 of PMMR0 can be set to logic ’1’ to block CLKIN from reaching the OMC flip-flops only, but CLKIN is still available to the PLD input bus and the APD counter.

See *Table 200 on page 249* for details.

28.6 PSD module reset conditions

The PSD module receives a reset signal from the MCU module. This reset signal is referred to as the “$\overline{RST}$” input in PSD module documentation, and it is active-low when asserted. The character of the $\overline{RST}$ signal generated from the MCU module is described in *Section 19: Supervisory functions on page 91*.

Figure 94. Example of chaining UPSD34xx devices



28.6.8 Debugging the 8032 MCU module

The 8032 on the MCU module may be debugged in-circuit using the same four basic JTAG signals as used for JTAG ISP (TDI, TDO, TCK, TMS). The signals TSTAT and $\overline{\text{TERR}}$ are not needed for debugging, and they will not create a problem if they exist on the circuit board while debugging. The same connector specified in *Figure 93 on page 262* can be used for ISP or for 8032 debugging. There are 3rd party suppliers of UPSD34xx JTAG debugging equipment (check www.st.com/psm). These are small pods which connect to a PC (or notebook computer) using a USB interface, and they are driven by an 8032 Integrated Development Environment (IDE) running on the PC.

Standard debugging features are provided through this JTAG interface such as single-step, breakpoints, trace, memory dump and fill, and others. There is also a dedicated Debug pin (shown in *Figure 90 on page 258*) which can be configured as an output to trigger external devices upon a programmable internal event (e.g., breakpoint match), or the pin can be configured as an input so an external device can initiate an internal debug event (e.g., break execution). The Debug pin function is configured by the 8032 IDE debug software tool. See *Section 12: Debug unit on page 60* for more details.

The Debug signal should always be pulled up externally with a weak pull-up (100 kΩ minimum) to V_{CC} even if nothing is connected to it, as shown in *Figure 91 on page 259* and *Figure 92 on page 261*.

Table 209. AC signal behavior symbols for timing

Letter	Meaning
t	Time
L	Logic level low or ALE
H	Logic level high
V	Valid
X	No longer a valid logic level
Z	Float
PW	Pulse width

Note: Example: t_{AVLX} = time from address valid to ALE invalid.

Figure 97. Switching waveforms – key

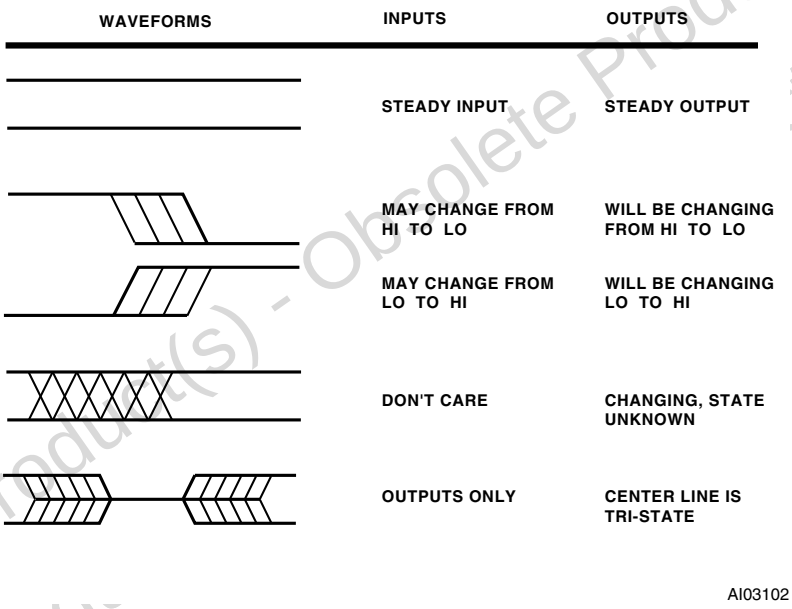


Figure 110. External clock cycle

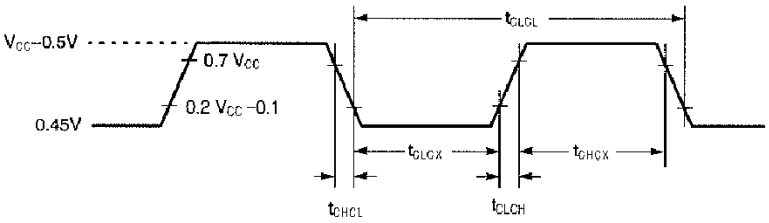


Figure 111. PSD module AC measurement I/O waveform

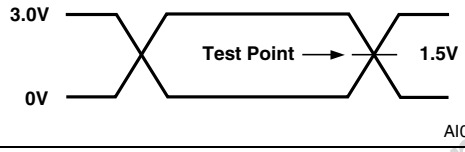


Figure 112. PSD module AC measurement load circuit

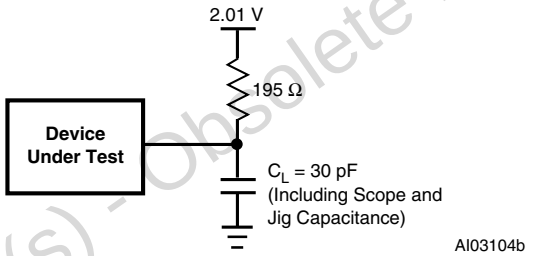


Table 236. I/O pin capacitance

Symbol	Parameter((1)	Test condition	Typ.(2)	Max.	Unit
C_{IN}	Input capacitance (for input pins)	V_{IN} = 0 V	4	6	pF
C_{OUT}	Output capacitance (for input/output pins)(3)	V_{OUT} = 0 V	8	12	pF

1. Sampled only, not 100% tested.
2. Typical values are for T_A = 25 °C and nominal supply voltages.
3. Maximum for MCU Address and Data lines is 20 pF each.

Table 240. Order codes

Part number	Max MHz	1st Flash	2nd Flash	SRAM	GPIO	8032 bus	V_{CC}	V_{DD}	Package
		(bytes)							
UPSD3422E-40T6	40	64K	32K	4K	35	No	3.3 V	5.0 V	LQFP52
UPSD3422EV-40T6	40	64K	32K	4K	35	No	3.3 V	3.3 V	LQFP52
UPSD3422E-40U6	40	64K	32K	4K	46	Yes	3.3 V	5.0 V	LQFP80
UPSD3422EV-40U6	40	64K	32K	4K	46	Yes	3.3 V	3.3 V	LQFP80
UPSD3433E-40T6	40	128K	32K	8K	35	No	3.3 V	5.0 V	LQFP52
UPSD3433EV-40T6	40	128K	32K	8K	35	No	3.3 V	3.3 V	LQFP52
UPSD3433E-40U6	40	128K	32K	8K	46	Yes	3.3 V	5.0 V	LQFP80
UPSD3433EV-40U6	40	128K	32K	8K	46	Yes	3.3 V	3.3 V	LQFP80
UPSD3434E-40T6	40	256K	32K	8K	35	No	3.3 V	5.0 V	LQFP52
UPSD3434EV-40T6	40	256K	32K	8K	35	No	3.3 V	3.3 V	LQFP52
UPSD3434E-40U6	40	256K	32K	8K	46	Yes	3.3 V	5.0 V	LQFP80
UPSD3434EV-40U6	40	256K	32K	8K	46	Yes	3.3 V	3.3 V	LQFP80
UPSD3454E-40T6	40	256K	32K	32K	35	No	3.3 V	5.0 V	LQFP52
UPSD3454EV-40T6	40	256K	32K	32K	35	No	3.3 V	3.3 V	LQFP52
UPSD3454E-40U6	40	256K	32K	32K	46	Yes	3.3 V	5.0 V	LQFP80
UPSD3454EV-40U6	40	256K	32K	32K	46	Yes	3.3 V	3.3 V	LQFP80
UPSD3422EB40T6	40	64K	32K	4K	35	No	3.3 V	5.0 V	LQFP52
UPSD3422EVB40T6	40	64K	32K	4K	35	No	3.3 V	3.3 V	LQFP52
UPSD3422EB40U6	40	64K	32K	4K	46	Yes	3.3 V	5.0 V	LQFP80
UPSD3422EVB40U6	40	64K	32K	4K	46	Yes	3.3 V	3.3 V	LQFP80
UPSD3433EB40T6	40	128K	32K	8K	35	No	3.3 V	5.0 V	LQFP52
UPSD3433EVB40T6	40	128K	32K	8K	35	No	3.3 V	3.3 V	LQFP52
UPSD3433EB40U6	40	128K	32K	8K	46	Yes	3.3 V	5.0 V	LQFP80
UPSD3433EVB40U6	40	128K	32K	8K	46	Yes	3.3 V	3.3 V	LQFP80
UPSD3434EB40T6	40	256K	32K	8K	35	No	3.3 V	5.0 V	LQFP52
UPSD3434EVB40T6	40	256K	32K	8K	35	No	3.3 V	3.3 V	LQFP52
UPSD3434EB40U6	40	256K	32K	8K	46	Yes	3.3 V	5.0 V	LQFP80
UPSD3434EVB40U6	40	256K	32K	8K	46	Yes	3.3 V	3.3 V	LQFP80
UPSD3454EB40T6	40	256K	32K	32K	35	No	3.3 V	5.0 V	LQFP52
UPSD3454EVB40T6	40	256K	32K	32K	35	No	3.3 V	3.3 V	LQFP52
UPSD3454EB40U6	40	256K	32K	32K	46	Yes	3.3 V	5.0 V	LQFP80
UPSD3454EVB40U6	40	256K	32K	32K	46	Yes	3.3 V	3.3 V	LQFP80

Note: Operating temperature is in the Industrial range (–40 °C to 85 °C).