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	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/upsd3434evb40u6

13.1.6 I^2C interrupt . . . 65
13.1.7 ADC interrupt . . . 65
13.1.8 PCA interrupt . . . 65
13.1.9 USB interrupt . . . 66

14 MCU clock generation . . . 68
14.1 MCU_CLK . . . 68
14.2 PERIPH_CLK . . . 68
14.2.1 JTAG interface clock . . . 68
14.2.2 USB_CLK . . . 69

15 Power saving modes . . . 72
15.1 Idle mode . . . 72
15.2 Power-down mode . . . 73
15.3 Reduced frequency mode . . . 73

16 Oscillator and external components . . . 76

17 I/O ports of mcu module . . . 78
17.1 MCU port operating modes . . . 78
17.1.1 GPIO function . . . 79
17.1.2 GPIO output . . . 79
17.1.3 GPIO input . . . 79
17.1.4 Alternate functions . . . 83

18 MCU bus interface . . . 86
18.1 PSEN bus cycles . . . 86
18.2 READ or WRITE bus cycles . . . 86
18.3 Connecting external devices to the MCU bus . . . 86
18.4 Programmable bus timing . . . 87
18.5 Controlling the PFQ and BC . . . 88

19 Supervisory functions . . . 91
19.1 External reset input pin, RESET_IN . . . 91
19.2 Low V_{CC} voltage detect, LVD . . . 92
19.3 Power-up reset . . . 92

22.2 Pulse width selection . . . 121

23 I²C interface . . . 123

23.1 I2C interface main features . . . 123
23.2 Communication flow . . . 124
23.3 Operating modes . . . 125
23.4 Bus arbitration . . . 126
23.5 Clock synchronization . . . 126
23.5.1 Clock sync during arbitration . . . 126
23.5.2 Clock sync during handshaking . . . 126
23.6 General call address . . . 127
23.7 Serial I/O engine (SIOE) . . . 127
23.8 I²C interface control register (S1CON) . . . 129
23.9 I²C interface status register (S1STA) . . . 130
23.9.1 Interrupt conditions . . . 130
23.10 I2C data shift register (S1DAT) . . . 131
23.10.1 Bus wait condition . . . 132
23.11 I²C address register (S1ADR) . . . 132
23.12 I²C Start sample setting (S1SETUP) . . . 133
23.13 I²C operating sequences . . . 135
23.13.1 Interrupt service routine (ISR) . . . 138

24 SPI (synchronous peripheral interface) . . . 142

24.1 SPI bus features and communication flow . . . 143
24.2 Full-duplex operation . . . 143
24.3 Bus-level activity . . . 143
24.4 SPI SFR registers . . . 145
24.5 SPI configuration . . . 146
24.6 Dynamic control . . . 147

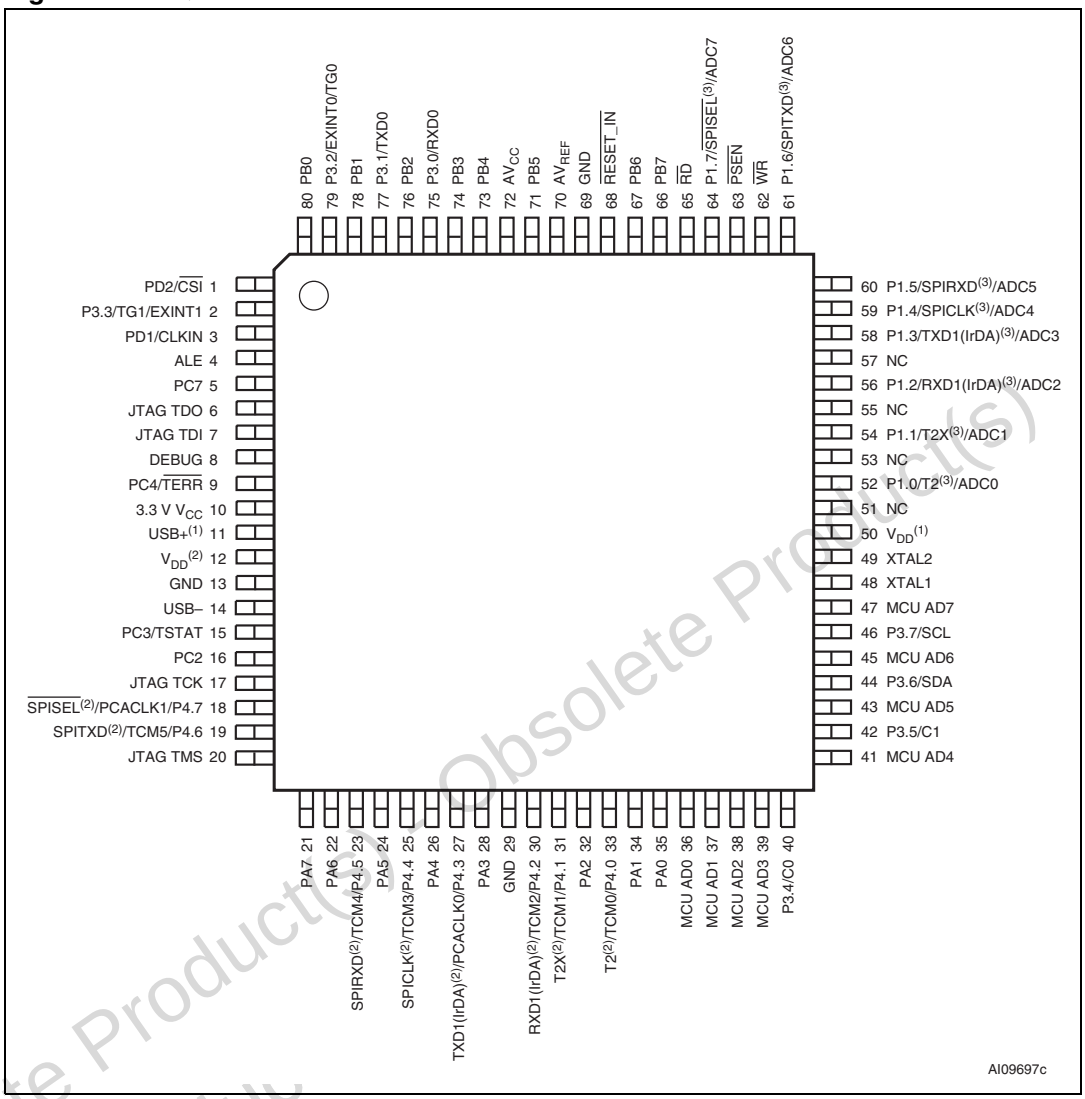
25 USB interface . . . 150

25.1 Basic USB concepts . . . 151
25.1.1 Communication flow . . . 151
25.1.2 Endpoints . . . 152
25.1.3 Packets . . . 152

List of tables

Table 1. Device summary . . . 1
Table 2. Pin definitions . . . 24
Table 3. Port type and voltage source combinations . . . 28
Table 4. Register bank select addresses . . . 40
Table 5. SFR memory map with direct address and reset value . . . 42
Table 6. Arithmetic instruction set . . . 52
Table 7. Logical instruction set . . . 53
Table 8. Data transfer instruction set . . . 54
Table 9. Boolean variable manipulation instruction set . . . 55
Table 10. Program branching instruction set . . . 55
Table 11. Miscellaneous instruction set . . . 56
Table 12. Notes on instruction set and addressing modes . . . 56
Table 13. DPTC: data pointer control register (SFR 85h, reset value 00h) . . . 57
Table 14. DPTC register bit definition . . . 57
Table 15. DPTM: data pointer mode register (SFR 86h, reset value 00h) . . . 58
Table 16. DPTM register bit definition . . . 58
Table 17. Interrupt summary . . . 63
Table 18. IE: interrupt enable register (SFR A8h, reset value 00h) . . . 66
Table 19. IE register bit definition . . . 66
Table 20. IEA: interrupt enable addition register (SFR A7h, reset value 00h) . . . 66
Table 21. IEA register bit definition . . . 66
Table 22. IP: interrupt priority register (SFR B8h, reset value 00h) . . . 67
Table 23. IP register bit definition . . . 67
Table 24. IPA: Interrupt Priority Addition register (SFR B7h, reset value 00h) . . . 67
Table 25. IPA register bit definition . . . 67
Table 26. PLLM and PLLD values for different f_{OSC} frequencies . . . 69
Table 27. CCON0: clock control register (SFR F9h, reset value 50h) . . . 70
Table 28. CCON0 register bit definition . . . 70
Table 29. CCON1 PLL control register (SFR FAh, reset value 00h) . . . 71
Table 30. CCON1 register bit definition . . . 71
Table 31. MCU module port and peripheral status during reduced power modes . . . 74
Table 32. State of 8032 MCU bus signals during power-down and idle modes . . . 74
Table 33. PCON: power control register (SFR 87h, reset value 00h) . . . 74
Table 34. PCON register bit definition . . . 74
Table 35. P1: I/O port 1 register (SFR 90h, reset value FFh) . . . 81
Table 36. P1 register bit definition . . . 81
Table 37. P3: I/O port 3 register (SFR B0h, reset value FFh) . . . 82
Table 38. P3 register bit definition . . . 82
Table 39. P4: I/O port 4 register (SFR C0h, reset value FFh) . . . 82
Table 40. P4 register bit definition . . . 82
Table 41. P3SFS: Port 3 special function select register (SFR 91h, reset value 00h) . . . 84
Table 42. P3SFS register bit definition . . . 84
Table 43. P1SFS0: Port 1 special function select 0 register (SFR 8Eh, reset value 00h) . . . 84
Table 44. P1SFS1: Port 1 special function select 1 register (SFR 8Fh, reset value 00h) . . . 84
Table 45. P1SFS0 and P1SFS1 details . . . 84
Table 46. P4SFS0: Port 4 special function select 0 register (SFR 92h, reset value 00h) . . . 85
Table 47. P4SFS1: Port 4 special function select 1 register (SFR 93h, reset value 00h) . . . 85
Table 48. P4SFS0 and P4SFS1 details . . . 85

Figure 3. LQFP80 connections



1. NC = Not connected
2. The USB+ pin needs a 1.5 kΩ pull-up resistor.
3. For 5 V applications, V_{DD} must be connected to a 5.0 V source. For 3.3 V applications, V_{DD} must be connected to a 3.3 V source.
4. These signals can be used on one of two different ports (Port 1 or Port 4) for flexibility. Default is Port1.

Table 2. Pin definitions

Port pin	Signal name	80-pin No.	52-pin No.(1)	In/out	Function		
					Basic	Alternate 1	Alternate 2
MCUAD0	AD0	36	N/A	I/O	External bus multiplexed address/data bus A0/D0		
MCUAD1	AD1	37	N/A	I/O	Multiplexed address/data bus A1/D1		
MCUAD2	AD2	38	N/A	I/O	Multiplexed address/data bus A2/D2		
MCUAD3	AD3	39	N/A	I/O	Multiplexed address/data bus A3/D3		
MCUAD4	AD4	41	N/A	I/O	Multiplexed address/data bus A4/D4		
MCUAD5	AD5	43	N/A	I/O	Multiplexed address/data bus A5/D5		
MCUAD6	AD6	45	N/A	I/O	Multiplexed address/data bus A6/D6		
MCUAD7	AD7	47	N/A	I/O	Multiplexed address/data bus A7/D7		
P1.0	T2 ADC0	52	34	I/O	General I/O port pin	Timer 2 Count input (T2)	ADC Channel 0 input (ADC0)
P1.1	T2X ADC1	54	35	I/O	General I/O port pin	Timer 2 Trigger input (T2X)	ADC Channel 1 input (ADC1)
P1.2	RxD1 ADC2	56	36	I/O	General I/O port pin	UART1 or IrDA Receive (RxD1)	ADC Channel 2 input (ADC2)
P1.3	TXD1 ADC3	58	37	I/O	General I/O port pin	UART or IrDA Transmit (TxD1)	ADC Channel 3 input (ADC3)
P1.4	SPICLK ADC4	59	38	I/O	General I/O port pin	SPI Clock Out (SPICLK)	ADC Channel 4 input (ADC4)
P1.5	SPIRxD ADC5	60	39	I/O	General I/O port pin	SPI Receive (SPIRxD)	ADC Channel 5 input (ADC5)
P1.6	SPITXD ADC6	61	40	I/O	General I/O port pin	SPI Transmit (SPITxD)	ADC Channel 6 input (ADC6)
P1.7	SPISEL ADC7	64	41	I/O	General I/O port pin	SPI Slave Select (SPISEL)	ADC Channel 7 input (ADC7)
P3.0	RxD0	75	23	I/O	General I/O port pin	UART0 Receive (RxD0)	

Table 2. Pin definitions (continued)

Port pin	Signal name	80-pin No.	52-pin No.(1)	In/out	Function		
					Basic	Alternate 1	Alternate 2
P3.1	TXD0	77	24	I/O	General I/O port pin	UART0 Transmit (TxD0)	
P3.2	EXINT0 TGO	79	25	I/O	General I/O port pin	Interrupt 0 input (EXTINT0)/Timer 0 gate control (TG0)	
P3.3	INT1	2	26	I/O	General I/O port pin	Interrupt 1 input (EXTINT1)/Timer 1 gate control (TG1)	
P3.4	C0	40	27	I/O	General I/O port pin	Counter 0 input (C0)	
P3.5	C1	42	28	I/O	General I/O port pin	Counter 1 input (C1)	
P3.6	SDA	44	29	I/O	General I/O port pin	I^2C bus serial data (I^2CSDA)	
P3.7	SCL	46	30	I/O	General I/O port pin	I^2C bus clock (I^2CSCL)	
P4.0	T2 TCM0	33	22	I/O	General I/O port pin	Program counter array0 PCA0-TCM0	Timer 2 count input (T2)
P4.1	T2X TCM1	31	21	I/O	General I/O port pin	PCA0-TCM1	Timer 2 trigger input (T2X)
P4.2	RXD1 TCM2	30	20	I/O	General I/O port pin	PCA0-TCM2	UART1 or IrDA Receive (RxD1)
P4.3	TXD1 PCACLK0	27	18	I/O	General I/O port pin	PCACLK0	UART1 or IrDA Transmit (TxD1)
P4.4	SPICLK TCM3	25	17	I/O	General I/O port pin	Program counter Array1 PCA1-TCM3	SPI clock out (SPICLK)
P4.5	SPIRXD TCM4	23	16	I/O	General I/O port pin	PCA1-TCM4	SPI Receive (SPIRxD)
P4.6	SPITXD	19	15	I/O	General I/O port pin	PCA1-TCM5	SPI Transmit (SPITxD)
P4.7	SPISEL PCACLK1	18	14	I/O	General I/O port pin	PCACLK1	SPI Slave Select (SPISEL)
AV_{REF}		70	N/A	I	Reference Voltage input for ADC. Connect AV_{REF} to V_{CC} if the ADC is not used.		
RD		65	N/A	O	READ signal, external bus		
WR		62	N/A	O	WRITE signal, external bus		
PSEN		63	N/A	O	PSEN signal, external bus		

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

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		
F9	CCON0	PLLM[4]	PLLEN	UPLLCE	DBGCE	CPU_AR	CPUPS[2:0]			50	*Table 27*
FA	CCON1	PLLM[3:0]				PLLD[3:0]				00	
FB	CCON2	–	–	–	PCA0CE	PCA0PS[3:0]				10	*Table 144*
FC	CCON3	–	–	–	PCA1CE	PCA1PS[3:0]				10	*Table 146*
FD	RESERVED										
FE	RESERVED										
FF	RESERVED										
FE	RESERVED										
FF	RESERVED										

1. This SFR can be addressed by individual bits (bit address mode) or addressed by the entire byte (direct address mode).

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)

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)

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 61. T2CON register bit definition (continued)

Bit	Symbol	R/W	Definition
4	TCLK(1)	R,W	UART0 Transmit Clock control. When TCLK = 1, UART0 uses Timer 2 overflow pulses for its transmit clock in Modes 1 and 3. TCLK=0, Timer 1 overflow is used for transmit clock
3	EXEN2	R,W	Timer 2 External Enable. When EXEN2 = 1, capture or reload results when negative edge on pin T2X occurs. EXEN2 = 0 causes Timer 2 to ignore events at pin T2X.
2	TR2	R,W	Timer 2 run control. 1 = Timer/Counter 2 is on, 0 = Timer Counter 2 is off.
1	C/$\overline{T2}$	R,W	Counter or Timer function select. When C/$\overline{T2}$ = 0, function is timer, clocked by internal clock. When C/$\overline{T2}$ = 1, function is counter, clocked by signal sampled on external pin, T2.
0	CP/$\overline{RL2}$	R,W	Capture/Reload. When CP/$\overline{RL2}$ = 1, capture occurs on negative transition at pin T2X if EXEN2 = 1. When CP/$\overline{RL2}$ = 0, auto-reload occurs when Timer 2 overflows, or on negative transition at pin T2X when EXEN2=1. When RCLK = 1 or TCLK = 1, CP/$\overline{RL2}$ is ignored, and Timer 2 is forced to auto-reload upon Timer 2 overflow

Note: 1 The RCLK1 and TCLK1 Bits in the SFR named PCON control UART1, and have the exact same function as RCLK and TCLK.

Table 62. Timer/counter 2 operating modes

Mode	Bits in T2CON SFR				Pin T2X (1)	Remarks	Input clock	
	RCLK or TCLK	CP/$\overline{RL2}$	TR2	EXEN2			Timer, internal	Counter, external (Pin T2, P1.0)
16-bit Auto-reload	0	0	1	0	x	reload [RCAP2H, RCAP2L] to [TH2, TL2] upon overflow (upcounting)	$f_{OSC}/12$	MAX $f_{OSC}/24$
	0	0	1	1	↓	reload [RCAP2H, RCAP2L] to [TH2, TL2] at falling edge on pin T2X		
16-bit capture	0	1	1	0	x	16-bit timer/counter (upcounting)	$f_{OSC}/12$	MAX $f_{OSC}/24$
	0	1	1	1	↓	Capture [TH2, TL2] and store to [RCAP2H, RCAP2L] at falling edge on pin T2X		

Table 74. Recommended CDIV[4:0] values to generate SIRClk (default CDIV[4:0] = 0Fh, 15 decimal)

f_{OSC} (MHz)	Value in CDIV[4:0]	Resulting f_{SIRCLK} (MHz)
40.00	16h, 22 decimal	1.82
36.864, or 36.00	14h, 20 decimal	1.84, or 1.80
24.00	0Dh, 13 decimal	1.84
11.059, or 12.00	06h, 6 decimal	1.84, or 2.00
7.3728(1)	04h, 4 decimal	1.84

1. When PULSE bit = 0 (fixed data pulse width), this is minimum recommended f_{OSC} because CDIV[4:0] must be 4 or greater.

24.1 SPI bus features and communication flow

The SPICLK signal is a gated clock generated from the UPSD34xx (Master) and regulates the flow of data bits. The Master may transmit at a variety of baud rates, and the SPICLK signal will clock one period for each bit of transmitted data. Data is shifted on one edge of SPICLK and sampled on the opposite edge.

The SPITxD signal is generated by the Master and received by the Slave device. The SPIRxD signal is generated by the Slave device and received by the Master. There may be no more than one Slave device transmitting data on SPIRxD at any given time in a multi-Slave configuration. Slave selection is accomplished when a Slave's "Slave Select" (SS) input is permanently grounded or asserted active-low by a Master device. Slave devices that are not selected do not interfere with SPI activities. Slave devices ignore SPICLK and keep their MISO output pins in high-impedance state when not selected.

The SPI specification allows a selection of clock polarity and clock phase with respect to data. The UPSD34xx supports the choice of clock polarity, but it does not support the choice of clock phase (phase is fixed at what is typically known as CPHA = 1). See *Figure 45* and *Figure 46 on page 145* for SPI data and clock relationships.

Referring to these figures (*45* and *46*), when the phase mode is defined as such (fixed at CPHA =1), in a new SPI data frame, the Master device begins driving the first data bit on SPITxD at the very first edge of the first clock period of SPICLK.

The Slave device will use this first clock edge as a transmission start indicator, and therefore the Slave's Slave Select input signal may remain grounded in a single-Master/single-Slave configuration (which means the user does not have to use the $\overline{\text{SPISEL}}$ signal from UPSD34xx in this case).

The SPI specification does not specify high-level protocol for data exchange, only low-level bit-serial transfers are defined.

24.2 Full-duplex operation

When an SPI transfer occurs, 8 bits of data are shifted out on one pin while a different 8 bits of data are simultaneously shifted in on a second pin. Another way to view this transfer is that an 8-bit shift register in the Master and another 8-bit shift register in the Slave are connected as a circular 16-bit shift register. When a transfer occurs, this distributed shift register is shifted 8 bit positions; thus, the data in the Master and Slave devices are effectively exchanged (see *Figure 44*).

24.3 Bus-level activity

Figure 45 details an SPI receive operation (with respect to bus Master) and *Figure 46* details an SPI transmit operation. Also shown are internal flags available to firmware to manage data flow. These flags are accessed through a number of SFRs.

Note: The UPSD34xx SPI interface SFRs allow the choice of transmitting the most significant bit (MSB) of a byte first, or the least significant bit (LSB) first. The same bit-order applies to data reception. Figures 45 and 46 illustrate shifting the LSB first.

- USB control register (UCTL)

  The USB control register (see *Table 120*) is used to enable the SIE, make the Endpoint FIFOs visible in the XDATA space and for generating a remote wakeup signal. Upon a reset, the USB module is disabled and must be enabled by the CPU for communication with the host over the USB.

Table 120. USB control register (UCTL 0ECh, reset value 00h)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
–	–	–	–	–	USBEN	VISIBLE	WAKEUP

Table 121. UCTL register bit definition

Bit	Symbol	R/W	Definition
7	–	–	Reserved
6	–	–	Reserved
5	–	–	Reserved
4	–	–	Reserved
3	–	–	Reserved
2	USBEN	R/W	USB Enable When this bit is set, the USB function is enabled and the SIE responds to tokens from the host. When this bit is clear, the USB function is disabled and does not respond to any tokens from the host. *Note: A USB reset does not clear this bit. Disabling and enabling the SIE using this bit resets part of the USB SIE state machine and some of the bits in the USTA and UCON registers.*
1	VISIBLE	R/W	USB FIFO VISIBLE When this bit is set, the selected USB FIFO is accessible (visible) in the XDATA space.
0	WAKEUP	R/W	Remote Wakeup Enable This bit forces a resume or “K” state on the USB data lines to initiate a remote wake-up. The CPU is responsible for controlling the timing of the forced resume that must be between 10ms and 15ms. Setting this bit will not cause the RESUMF Bit to be set.

- 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

Table 149. PCA0 register bit definition (continued)

Bit	Symbol	Function
3	–	Reserved
2	10B_PWM	0 = Select 16-bit PWM 1 = Select 10-bit PWM
1-0	CLK_SEL [1:0]	00 Select Prescaler clock as Counter clock 01 Select Timer 0 Overflow 10 Select External Clock pin (P4.3 for PCA0) (MAX clock rate = $f_{OSC}/4$)

Table 150. PCA1 control register PCACON1 (SFR 0BCh, reset value 00h)

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

Table 151. PCA1 register bit definition

Bit	Symbol	Function
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
3	–	Reserved
2	10B_PWM	0 = Select 16-bit PWM 1 = Select 10-bit PWM
1-0	CLK_SEL [1:0]	00 Select Prescaler clock as Counter clock 01 Select Timer 0 Overflow 10 Select External Clock pin (P4.7 for PCA1) (MAX clock rate = $f_{OSC}/4$)

Table 152. PCA status register PCASTA (SFR 0A5h, reset value 00h)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OVF1	INTF5	INTF4	INTF3	OVF0	INTF2	INTF1	INTF0

Table 153. PCASTA register bit definition

Bit	Symbol	Function
7	OFV1	PCA1 Counter OverFlow flag Set by hardware when the counter rolls over. OVF1 flags an interrupt if Bit EOVFI in PCACON1 is set. OVF1 may be set with either hardware or software but can only be cleared with software.
6	INTF5	TCM5 Interrupt flag Set by hardware when a match or capture event occurs. Must be clear with software.

maintained by alternating between the two flash sectors. For example, a data set of 128 bytes is written and maintained by software in a distributed fashion across one 8 Kbyte sector of Secondary Flash memory until it becomes full. Then the writing continues on the other 8 Kbyte sector while erasing the first 8 Kbyte sector. This process repeats continuously, bouncing back and forth between the two 8 Kbyte sectors. This creates a wear-leveling effect, which increases the effective number of erase cycles for a data set of 128 bytes to many times more than the base 100 000 erase cycles of the Flash memory. EEPROM emulation in Flash memory is typically faster than writing to actual EEPROM memory, and more reliable because the last known value in a data set is maintained even if a WRITE cycle is corrupted by a power outage. The EEPROM emulation function can be called by the user's firmware, making it appear that the user is writing a single byte, or data record, thus hiding all of the data management that occurs within the two 8 Kbyte Flash sectors. EEPROM emulation firmware for the UPSD34xx is available from www.st.com/psm.

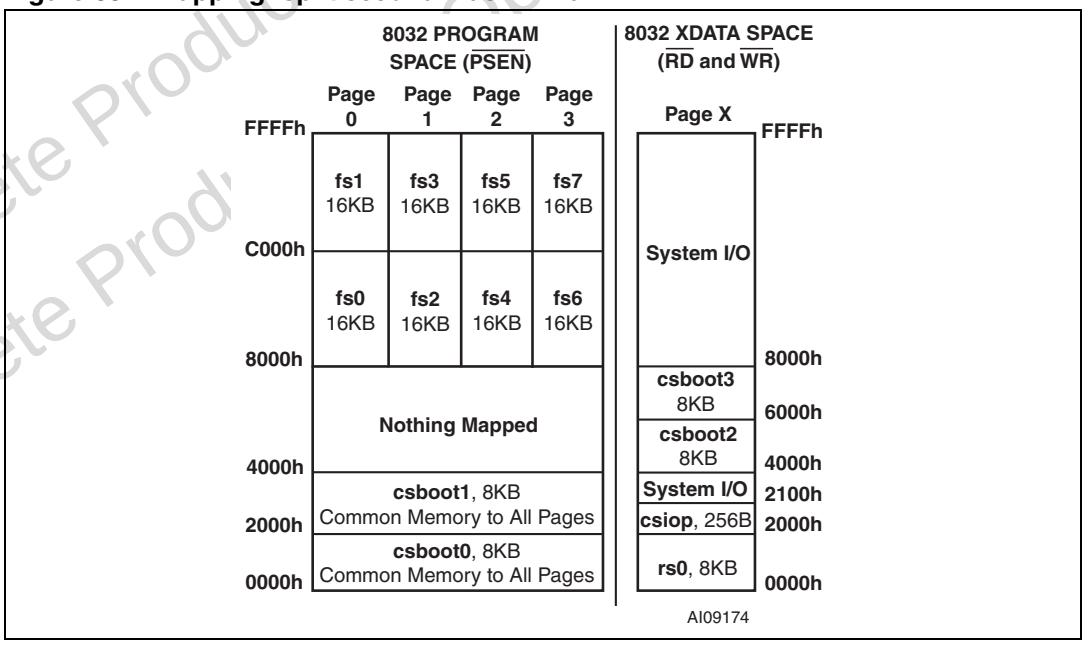
28.2.5 Alternative mapping schemes

Here are more possible memory maps for the UPSD3433.

Note: Mapping examples would be slightly different for UPSD3433 and UPSD3434, because of the different sizes of individual Flash memory sectors.

- ***Figure 65*** Place the larger Main Flash Memory into program space, but split the Secondary Flash in half, placing two of its sectors into XDATA space and remaining two sectors into program space. This method allows the designer to put IAP code (or boot code) into two sectors of Secondary Flash in program space, and use the other two Secondary Flash sectors for data storage, such as EEPROM emulation in XDATA space.
- ***Figure 66*** Place both the Main and Secondary Flash memories into program space for maximum code storage, with no Flash memory in XDATA space.

Figure 65. Mapping: split second Flash in half



28.5.9 Erase time-out flag (DQ3)

The Erase Time-out Flag Bit (DQ3) reflects the time-out period allowed between two consecutive sector erase instruction sequence bytes. If multiple sector erase commands are desired, the additional sector erase commands (30h) must be sent by the 8032 within 80us after the previous sector erase command. DQ3 is 0 before this time period has expired, indicating it is OK to issue additional sector erase commands. DQ3 will go to logic ’1’ if the time has been longer than 80µs since the previous sector erase command (time has expired), indication that is not OK to send another sector erase command. In this case, the 8032 must start a new sector erase instruction sequence (unlock and command) beginning again after the current sector erase operation has completed.

28.5.10 Programming Flash memory

When a byte of Flash memory is programmed, individual bits are programmed to logic '0.' cannot program a bit in Flash memory to a logic ’1’ once it has been programmed to a logic '0.' A bit must be erased to logic ’1’, and programmed to logic '0.' That means Flash memory must be erased prior to being programmed. A byte of Flash memory is erased to all 1s (FFh). The 8032 may erase the entire Flash memory array all at once, or erase individual sector-by-sector, but not erase byte-by-byte. However, even though the Flash memories cannot be *erased* byte-by-byte, the 8032 may *program* Flash memory byte-by-byte. This means the 8032 does not need to program group of bytes (64, 128, etc.) at one time, like some Flash memories.

Each Flash memory requires the 8032 to send an instruction sequence to program a byte or to erase sectors (see *Table 163 on page 209*).

If the byte to be programmed is in a protected Flash memory sector, the instruction sequence is ignored.

Important note: It is mandatory that a chip-select signal is active for the Flash sector where a programming instruction sequence is targeted. The user must make sure that the correct chip-select equation, FSx or CSBOOTx specified in PSDsoft Express matches the address range that the 8032 firmware is accessing, otherwise the instruction sequence will not be recognized by the Flash array. If memory paging is used, be sure that the 8032 firmware sets the page register to the correct page number before issuing an instruction sequence to the Flash memory segment on a particular memory page, otherwise the correct sector select signal will not become active.

Once the 8032 issues a Flash memory program or erase instruction sequence, it must check the status bits for completion. The embedded algorithms that are invoked inside a Flash memory array provide several ways to give status to the 8032. Status may be checked using any of three methods: Data Polling, Data Toggle, or Ready/$\overline{\text{Busy}}$ (pin PC3).

Table 164. Flash memory status bit definition(1) (2)

Functional block	FSx, or CSBOOTx	DQ7	DQ6	DQ5	DQ4	DQ3	DQ2	DQ1	DQ0
Flash memory	Active (the desired segment is selected)	Data polling	Toggle flag	Error flag	X	Erase timeout	X	X	X

1. X = Not guaranteed value, can be read either '1' or '0.'
2. DQ7-DQ0 represent the 8032 data bus bits, D7-D0.

Table 193. Port B pin drive select register (address = csiop + offset 09h)(1) (2)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PB7 open drain	PB6 open drain	PB5 open drain	PB4 open drain	PB3 slew rate	PB2 slew rate	PB1 slew rate	PB0 slew rate

1. For each bit, 1 = pin drive type is selected, 0 = pin drive type is default mode, CMOS push/pull.
2. Default state for register is 00h after reset or power-up.

Table 194. Port C pin drive select register (address = csiop + offset 16h)(1) (2)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PC7 open drain	N/A (JTAG)	N/A (JTAG)	PC4 open drain	PC3 open drain	PC2 open drain	N/A (JTAG)	N/A (JTAG)

1. For each bit, 1 = pin drive type is selected, 0 = pin drive type is default mode, CMOS push/pull
2. Default state for register is 00h after reset or power-up

Table 195. Port D pin drive select register (address = csiop + offset 17h)(1) (2)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
N/A	N/A	N/A	N/A	N/A	PD2(3) slew rate	PD1 slew rate	N/A

1. For each bit, 1 = pin drive type is selected, 0 = pin drive type is default mode, CMOS push/pull
2. Default state for register is 00h after reset or power-up
3. Pin is not available on 52-pin UPSD34xx devices

Table 196. Port A enable out register(1) (2)(address = csiop + offset 0Ch)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PA7 OE	PA6 OE	PA5 OE	PA4 OE	PA3 OE	PA2 OE	PA1 OE	PA0 OE

1. Port A not available on 52-pin UPSD34xx devices
2. For each bit, 1 = pin drive is enabled as an output, 0 = pin drive is off (high-impedance, pin used as input)

Table 197. Port B enable out register (address = csiop + offset 0Dh)(1)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PB7 OE	PB6 OE	PB5 OE	PB4 OE	PB3 OE	PB2 OE	PB1 OE	PB0 OE

1. For each bit, 1 = pin drive is enabled as an output, 0 = pin drive is off (high-impedance, pin used as input)

Table 198. Port C enable out register (address = csiop + offset 1Ah)(1)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PC7 OE	N/A (JTAG)	N/A (JTAG)	PC4 OE	PC3 OE	PC2 OE	N/A (JTAG)	N/A (JTAG)

1. For each bit, 1 = pin drive is enabled as an output, 0 = pin drive is off (high-impedance, pin used as input)

Table 199. Port D enable out register (address = csiop + offset 1Bh)(1)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
N/A	N/A	N/A	N/A	N/A	PD2 OE(2)	PD1 OE	N/A

1. For each bit, 1 = pin drive is enabled as an output, 0 = pin drive is off (high-impedance, pin used as input)
2. Pin is not available on 52-pin UPSD34xx devices

assembly code example in *Table* , the PFQ will be loaded with the final instructions to command the MCU module to Power Down mode after the PDS Module goes to Power-Down mode. In this case, even though the code memory goes off-line in the PSD module, the last few MCU instruction are sourced from the PFQ.

Forced power-down example

```
PDOWN:  ANL   A8h, #7Fh      ; disable all interrupts
        ORL   9Dh, #C0h      ; ensure PFQ and BC are enabled
        MOV   DPTR, #xxC7    ; load XDATA pointer to select PMMR3 register (xx = base
                             ; address of csiop registers)
        CLR   A              ; clear A
        JMP   LOOP           ; first loop - fill PFQ/BQ with Power Down instructions
        NOP                  ; second loop - fetch code from PFQ/BC and set Power-
                             ; Down bits for PSD module and then MCU module
LOOP:   MOVX  @DPTR, A       ; set FORCE_PD Bit in PMMR3 in PSD module in second
                             ; loop
        MOV   87h, A         ; set PD Bit in PCON register in MCU module in second
                             ; loop
        MOV   A, #02h        ; set power-down bit in the A register, but not in PMMR3 or
                             ; PCON yet in first loop
        JMP   LOOP           ; uPSD enters into Power-Down mode in second loop
```

Figure 88. Automatic power-down (APD) unit