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"

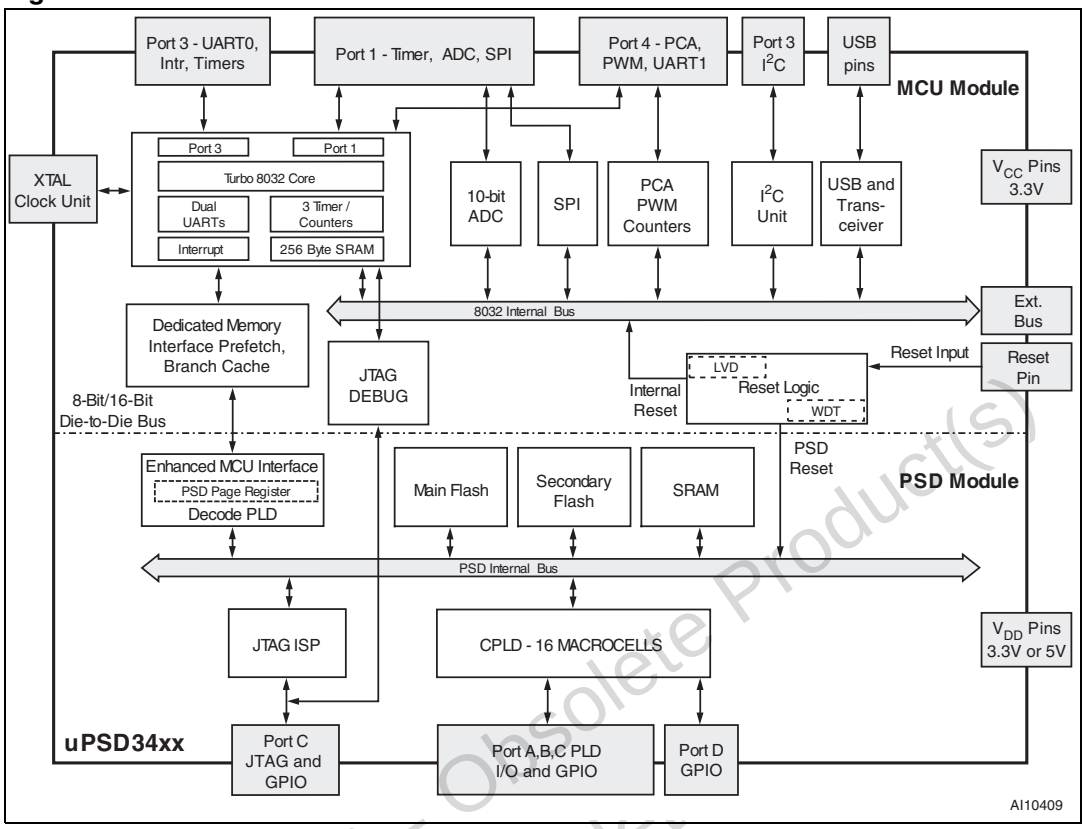
Details	
Product Status	Obsolete
Core Processor	8032
Core Size	8-Bit
Speed	40MHz
Connectivity	I²C, IrDA, SPI, UART/USART, USB
Peripherals	LVD, POR, PWM, WDT
Number of I/O	46
Program Memory Size	288KB (288K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	8K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 3.6V
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/upsd3434ev-40u6

List of tables

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)	

Figure 4. Functional modules



4.2.1 Program memory

External program memory is addressed by the 8032 using its 16-bit program counter (PC) and is accessed with the 8032 signal, $\overline{PSEN}$. Program memory can be present at any address in program space between 0000h and FFFFh.

After a power-up or reset, the 8032 begins program execution from location 0000h where the reset vector is stored, causing a jump to an initialization routine in firmware. At address 0003h, just following the reset vector are the interrupt service locations. Each interrupt is assigned a fixed interrupt service location in program memory. An interrupt causes the 8032 to jump to that service location, where it commences execution of the service routine. External Interrupt 0 (EXINT0), for example, is assigned to service location 0003h. If EXINT0 is going to be used, its service routine must begin at location 0003h. Interrupt service locations are spaced at 8-byte intervals: 0003h for EXINT0, 000Bh for Timer 0, 0013h for EXINT1, and so forth. If an interrupt service routine is short enough, it can reside entirely within the 8-byte interval. Longer service routines can use a jump instruction to somewhere else in program memory.

4.2.2 Data memory

External data is referred to as XDATA and is addressed by the 8032 using Indirect Addressing via its 16-bit data pointer register (DPTR) and is accessed by the 8032 signals, $\overline{RD}$ and $\overline{WR}$. XDATA can be present at any address in data space between 0000h and FFFFh.

Note: The UPSD34xx has dual data pointers (source and destination) making XDATA transfers much more efficient.

4.2.3 Memory placement

PSD module architecture allows the placement of its external memories into different combinations of program memory and data memory spaces. This means the main Flash, the secondary Flash, and the SRAM can be viewed by the 8032 MCU in various combinations of program memory or data memory as defined by PSDsoft Express.

As an example of this flexibility, for applications that require a great deal of Flash memory in data space (large lookup tables or extended data recording), the larger main Flash memory can be placed in data space and the smaller secondary Flash memory can be placed in program space. The opposite can be realized for a different application if more Flash memory is needed for code and less Flash memory for data.
By default, the SRAM and csiop memories on the PSD module must always reside in data memory space and they are treated by the 8032 as XDATA.
The main Flash and secondary Flash memories may reside in program space, data space, or both. These memory placement choices specified by PSDsoft Express are programmed into non-volatile sections of the UPSD34xx, and are active at power-up and after reset. It is possible to override these initial settings during runtime for In-Application Programming (IAP).

Standard 8032 MCU architecture cannot write to its own program memory space to prevent accidental corruption of firmware. However, this becomes an obstacle in typical 8032 systems when a remote update to firmware in Flash memory is required using IAP. The PSD module provides a solution for remote updates by allowing 8032 firmware to temporarily “reclassify” Flash memory to reside in data space during a remote update, then returning Flash memory back to program space when finished. See the VM register (*Table 160 on page 203*) in the PSD module section of this document for more details.

7.4 Accumulator (ACC)

This is an 8-bit general purpose register which holds a source operand and receives the result of arithmetic operations. The ACC register can also be the source or destination of logic and data movement operations. For MUL and DIV instructions, ACC is combined with the B register to hold 16-bit operands. The ACC is referred to as “A” in the MCU instruction set.

7.5 B register (B)

The B register is a general purpose 8-bit register for temporary data storage and also used as a 16-bit register when concatenated with the ACC register for use with MUL and DIV instructions.

7.6 General purpose registers (R0 - R7)

There are four banks of eight general purpose 8-bit registers (R0 - R7), but only one bank of eight registers is active at any given time depending on the setting in the PSW word (described next). R0 - R7 are generally used to assist in manipulating values and moving data from one memory location to another. These register banks physically reside in the first 32 locations of 8032 internal DATA SRAM, starting at address 00h. At reset, only the first bank of eight registers is active (addresses 00h to 07h), and the stack begins at address 08h.

7.7 Program status word (PSW)

The PSW is an 8-bit register which stores several important bits, or flags, that are set and cleared by many 8032 instructions, reflecting the current state of the MCU core. *Figure 11 on page 40* shows the individual flags.

7.7.1 Carry flag (CY)

This flag is set when the last arithmetic operation that was executed results in a carry (addition) or borrow (subtraction). It is cleared by all other arithmetic operations. The CY flag is also affected by Shift and Rotate Instructions.

7.7.2 Auxiliary carry flag (AC)

This flag is set when the last arithmetic operation that was executed results in a carry into (addition) or borrow from (subtraction) the high-order nibble. It is cleared by all other arithmetic operations.

7.7.3 General purpose flag (F0)

This is a bit-addressable, general-purpose flag for use under software control.

7.7.4 Register bank select flags (RS1, RS0)

These bits select which bank of eight registers is used during R0 - R7 register accesses (see *Table 4*)

9.4 Immediate addressing

This mode uses 8-bits of data (a constant) contained in the second byte of the instruction, and stores it into the memory location or register indicated by the first byte of the instruction. Thus, the data is immediately available within the instruction. This mode is commonly used to initialize registers and SFRs or to perform mask operations.

There is also a 16-bit version of this mode for loading the DPTR register. In this case, the two bytes following the instruction byte contain the 16-bit value. For example:

```
MOV A, 40#          ; Move the constant, 40h, into
                    ; the accumulator
MOV DPTR, 1234#     ; Move the constant, 1234h, into
                    ; DPTR
```

9.5 External direct addressing

This mode will access external memory (XDATA) by using the 16-bit address stored in the DPTR register. There are only two instructions using this mode and both use the accumulator to either receive a byte from external memory addressed by DPTR or to send a byte from the accumulator to the address in DPTR. The UPSD34xx has a special feature to alternate the contents (source and destination) of DPTR rapidly to implement very efficient memory-to-memory transfers. For example:

```
MOVX A, @DPTR       ; Move contents of accumulator to
                    ; XDATA at address contained in
                    ; DPTR
MOVX @DPTR, A       ; Move XDATA to accumulator
```

Note: See details in Section 11: Dual data pointers on page 57.

9.6 External indirect addressing

This mode will access external memory (XDATA) by using the 8-bit address stored in either register R0 or R1. This is the fastest way to access XDATA (least bus cycles), but because only 8-bits are available for address, this mode limits XDATA to a size of only 256 bytes (the traditional Port 2 of the 8032 MCU is not available in the UPSD34xx, so it is not possible to write the upper address byte).

For example:

```
MOVX @R0,A          ; Move into the accumulator the
                    ; XDATA that is pointed to by
                    ; the address contained in R0.
```

Note: This mode is not supported by UPSD34xx.

14 MCU clock generation

Internal system clocks generated by the clock generation unit are derived from the signal, XTAL1, shown in *Figure 13*. XTAL1 has a frequency f_{OSC}, which comes directly from the external crystal or oscillator device. The SFR named CCON0 (*Table 27 on page 70*) controls the clock generation unit.

There are two clock signals produced by the clock generation unit:

- MCU_CLK
- PERIPH_CLK

14.1 MCU_CLK

This clock drives the 8032 MCU core and the Watchdog Timer (WDT). The frequency of MCU_CLK is equal to f_{OSC} by default, but it can be divided by as much as 2048, shown in *Figure 13*. The bits CPUPS[2:0] select one of eight different divisors, ranging from 2 to 2048. The new frequency is available immediately after the CPUPS[2:0] bits are written. The final frequency of MCU_CLK is f_{MCU}.

MCU_CLK is blocked by either bit, PD or IDL, in the SFR named PCON during MCU Power-down mode or Idle mode respectively.

MCU_CLK clock can be further divided as required for use in the WDT. See details of the WDT in *Section 19: Supervisory functions on page 91*.

14.2 PERIPH_CLK

This clock drives all the UPSD34xx peripherals except the WDT. The Frequency of PERIPH_CLK is always f_{OSC}. Each of the peripherals can independently divide PERIPH_CLK to scale it appropriately for use.

PERIPH_CLK runs at all times except when blocked by the PD bit in the SFR named PCON during MCU Power-down mode.

14.2.1 JTAG interface clock

The JTAG interface for ISP and for Debugging uses the externally supplied JTAG clock, coming in on pin TCK. This means the JTAG ISP interface is always available, and the JTAG Debug interface is available when enabled, even during MCU Idle mode and Power-down mode.

However, since the MCU participates in the JTAG debug process, and MCU_CLK is halted during Idle and Power-down modes, the majority of debug functions are not available during these low power modes. But the JTAG debug interface is capable of executing a reset command while in these low power modes, which will exit back to normal operating mode where all debug commands are available again.

The CCON0 SFR contains a bit, DBGCE, which enables the breakpoint comparators inside the JTAG Debug Unit when set. DBGCE is set by default after reset, and firmware may clear this bit at run-time. Disabling these comparators will reduce current consumption on the MCU module, and it is recommended to do so if the Debug Unit will not be used (such as in the production version of an end-product).

CPL, INC, DEC, DJNZ, MOV, CLR, and SETB. All other types of reads to port SFRs will read the actual pin logic level and not the port latch. This is consistent with 8051 architecture.

Figure 15. MCU module port pin function routing

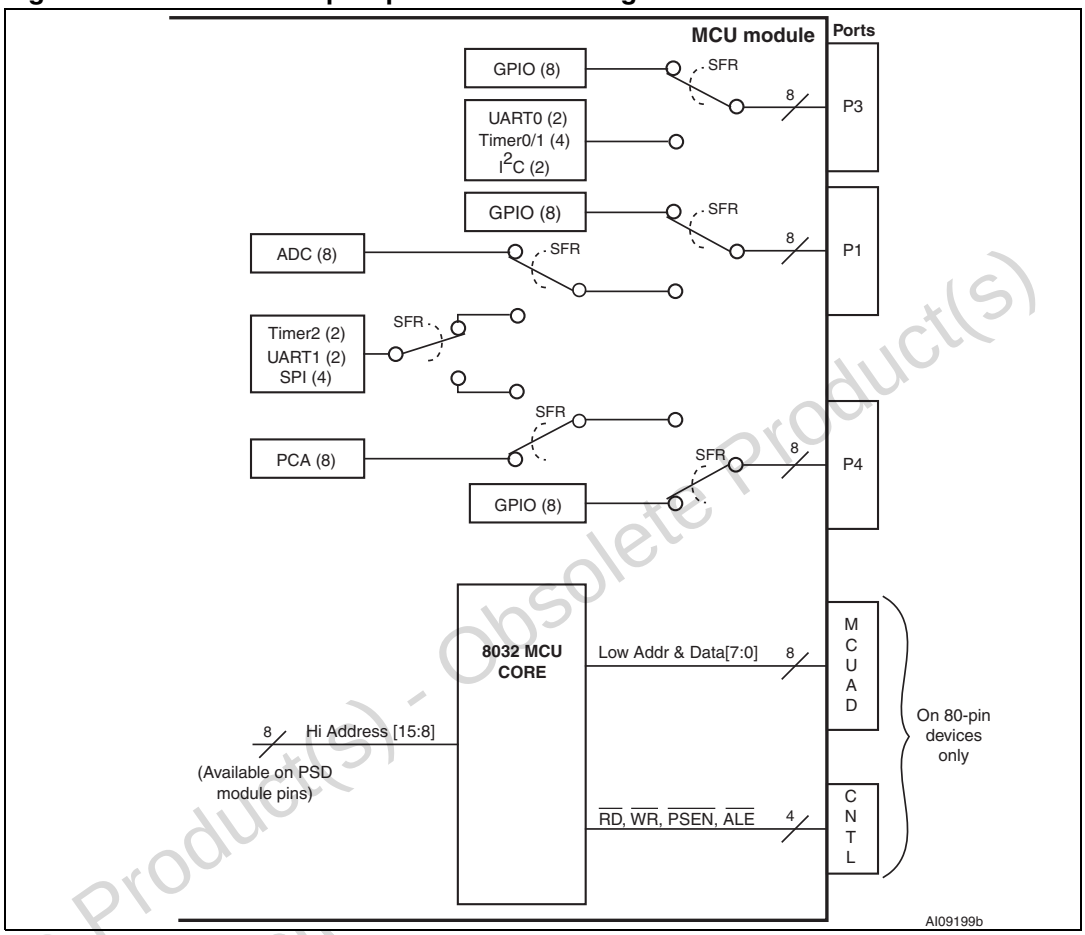


Figure 16. MCU I/O cell block diagram for port 1

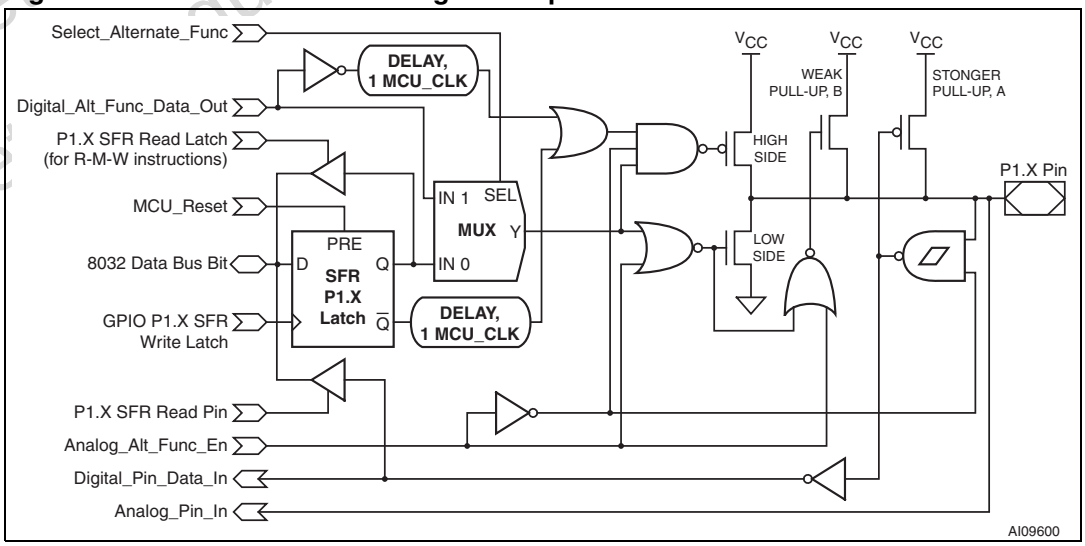


Figure 17. MCU I/O cell block diagram for port 3

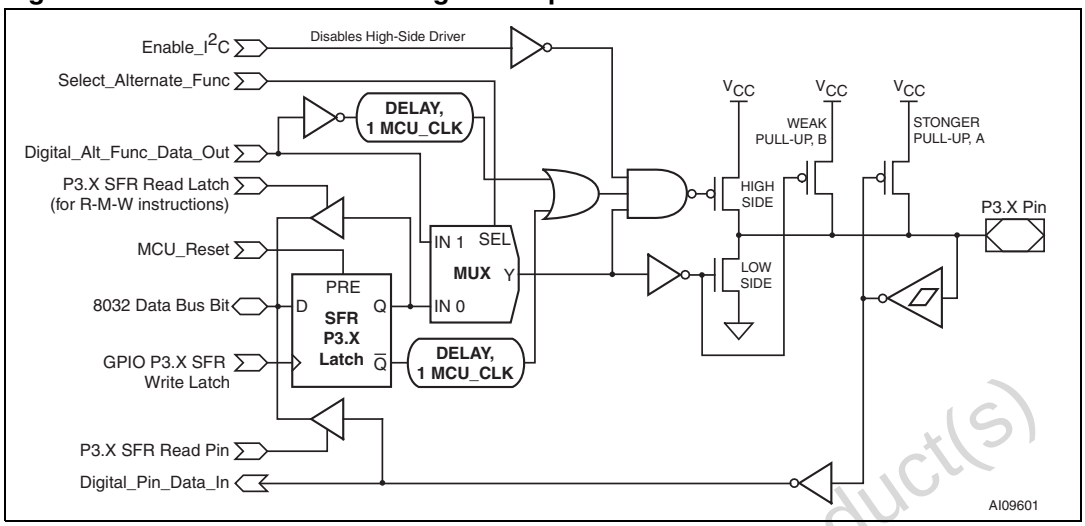


Figure 18. MCU I/O cell block diagram for port 4

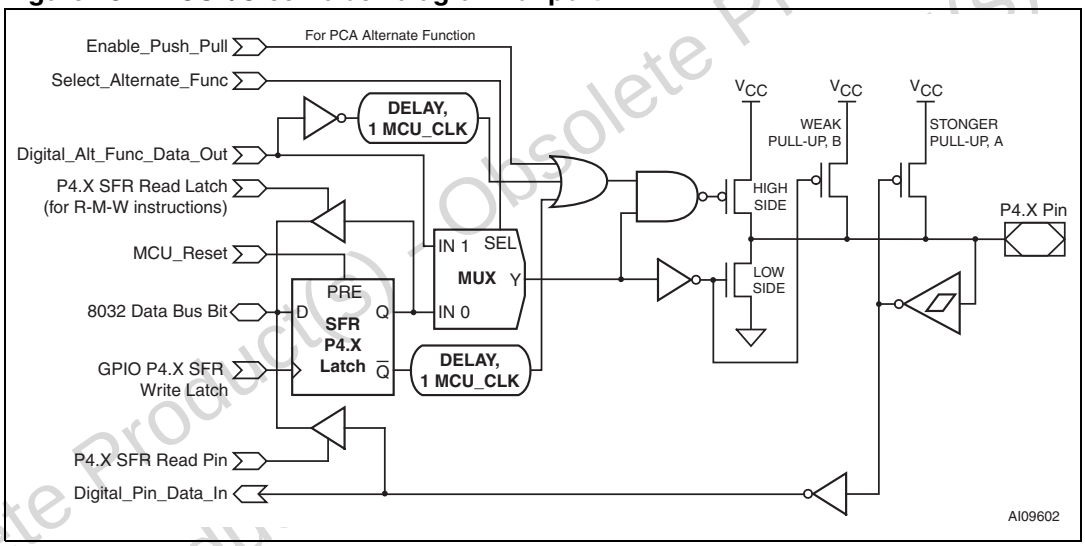


Table 35. P1: I/O port 1 register (SFR 90h, reset value FFh)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
P1.7	P1.6	P1.5	P1.4	P1.3	P1.2	P1.1	P1.0

Table 36. P1 register bit definition

Bit	Symbol	R/W	Function(1)
7	P1.7	R,W	Port pin 1.7
6	P1.6	R,W	Port pin 1.6
5	P1.5	R,W	Port pin 1.5
4	P1.4	R,W	Port pin 1.4
3	P1.3	R,W	Port pin 1.3

Figure 36. UART mode 3, block diagram

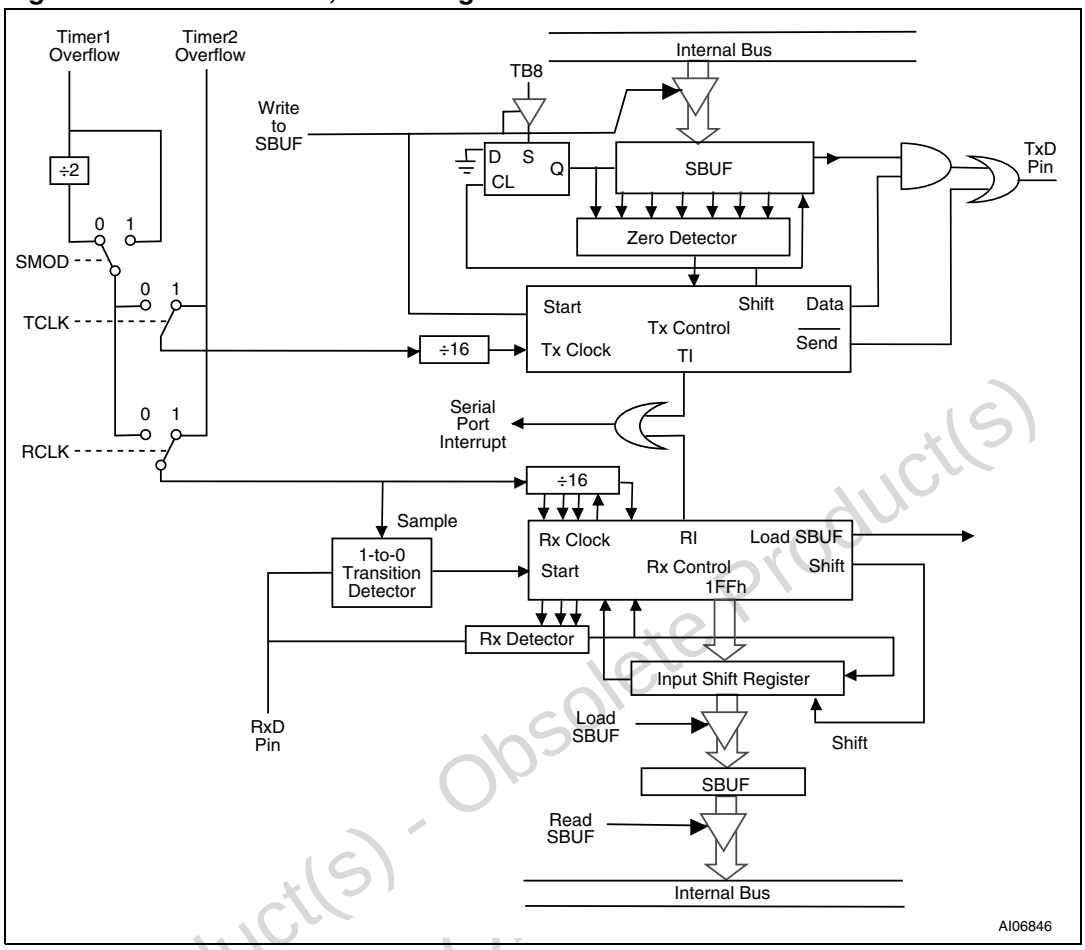
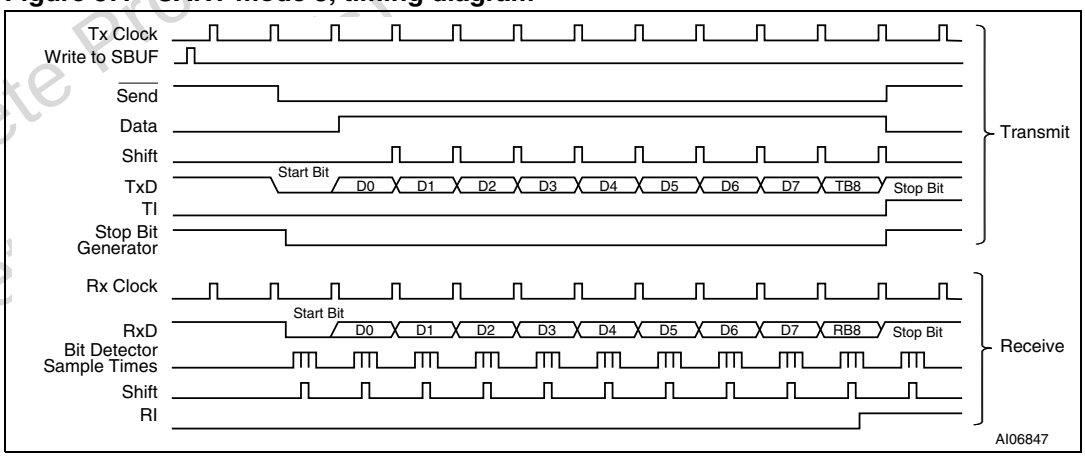


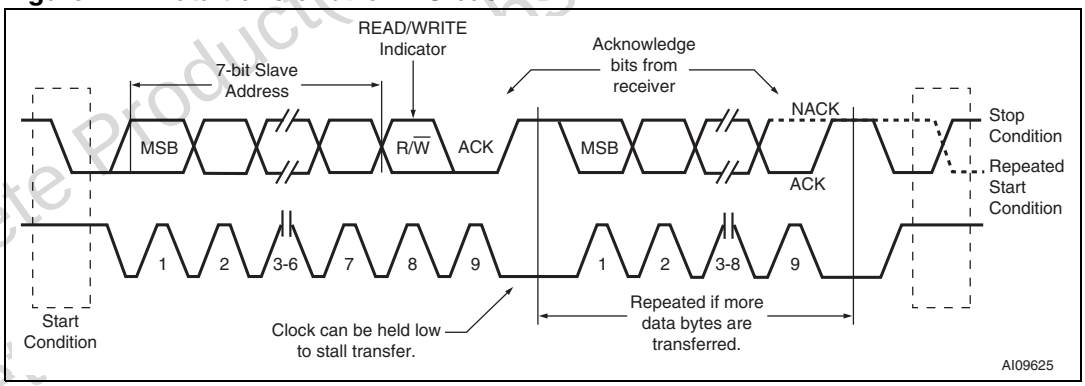
Figure 37. UART mode 3, timing diagram



A few things to know related to these transfers:

- Either the Master or Slave device can hold the SCL clock line low to indicate it needs more time to handle a byte transfer. An indefinite holding period is possible.
- A Start condition is generated by a Master and recognized by a Slave when SDA has a 1-to-0 transition while SCL is high (*Figure 41 on page 125*).
- A Stop condition is generated by a Master and recognized by a Slave when SDA has a 0-to1 transition while SCL is high (*Figure 41 on page 125*).
- A Re-Start (repeated Start) condition generated by a Master can have the same function as a Stop condition when starting another data transfer immediately following the previous data transfer (*Figure 41 on page 125*).
- When transferring data, the logic level on the SDA line must remain stable while SCL is high, and SDA can change only while SCL is low. However, when not transferring data, SDA may change state while SCL is high, which creates the Start and Stop bus conditions.
- An Acknowlegde bit is generated from a Master or a Slave by driving SDA low during the “ninth” bit time, just following each 8-bit byte that is transfered on the bus (*Figure 41 on page 125*). A Non-Acknowledge occurs when SDA is asserted high during the ninth bit time. All byte transfers on the I²C bus include a 9th bit time reserved for an Acknowlege (ACK) or Non-Acknowledge (NACK).
- An additional Master device that desires to control the bus should wait until the bus is not busy before generating a Start condition so that a possible Slave operation is not interrupted.
- If two Master devices both try to generate a Start condition simultaneously, the Master who looses arbitration will switch immediately to Slave mode so it can recoginize its own Slave address should it appear on the bus.

Figure 41. Data transfer on an I²C bus



23.3 Operating modes

The I²C interface supports four operating modes:

- Master-transmitter
- Master-receiver
- Slave-transmitter
- Slave-receiver

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

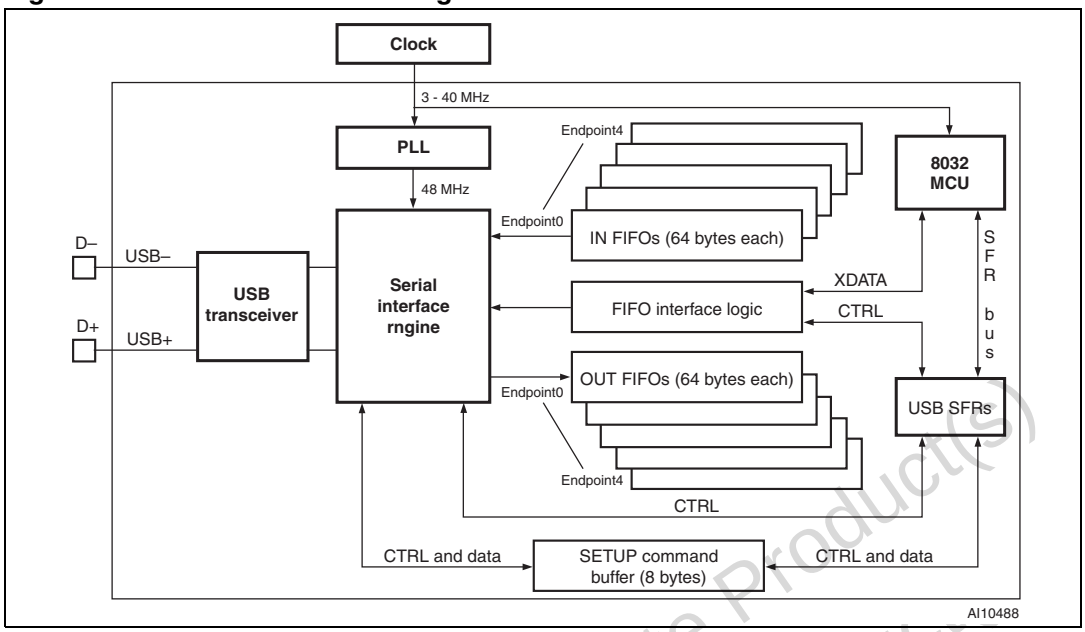
When an SPI transfer occurs, 8 bits of data are shifted out on one pin while a different 8 bits of data are simultaneously shifted in on a second pin. Another way to view this transfer is that an 8-bit shift register in the Master and another 8-bit shift register in the Slave are connected as a circular 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.

Figure 48. USB module block diagram



25.1 Basic USB concepts

The Universal Serial Bus (USB) is more complex than the standard serial port and requires familiarity with the specification to fully understand how to use the USB peripheral in the UPSD34xx. The USB specification is available on the Internet at **http://www.usb.org.** Some basic concepts will be presented in this section but knowledge of the USB specification is required.

In a USB system, there is only one master and the master is the host computer. The host controls all activity on the bus and devices respond to requests from the host. The only exception is when a device has been put into a low power suspend mode by the host. In this case, the device can signal a remote wakeup. Outside of that exception, all activity is controlled and initiated by the host. The host-centric model versus a peer-to-peer model provides the best way to develop low cost peripherals by keeping the complex control logic on the host side. The UPSD34xx is a peripheral (non-host) device.

25.1.1 Communication flow

The USB provides a means for communication between host (client) software and a function on a USB device. Functions can have different requirements for the communication flow depending on the client software to the USB function interaction. With USB, the various communication flows are separated to provide better bus utilization. For example, one communication flow is used for managing the device while another is for transferring data related to the operation of the device. Some bus access is used for each communication flow with each flow terminated at an endpoint on a device. Each endpoint has various aspects associated with the communication flow. A USB device looks like a collection of endpoints to the USB system.

27 Programmable counter array (PCA) with PWM

There are two Programmable Counter Array blocks (PCA0 and PCA1) in the UPSD34xx. A PCA block consists of a 16-bit up-counter, which is shared by three TCM (Timer Counter Module). A TCM can be programmed to perform one of the following four functions:

1. Capture Mode: capture counter values by external input signals
2. Timer Mode
3. Toggle Output Mode
4. PWM Mode: fixed frequency (8-bit or 16-bit), programmable frequency (8-bit only)

27.1 PCA block

The 16-bit Up-Counter in the PCA block is a free-running counter (except in PWM Mode with programmable frequency). The Counter has a choice of clock input: from an external pin, Timer 0 Overflow, or PCA Clock.

A PCA block has 3 Timer Counter Modules (TCM) which share the 16-bit Counter output. The TCM can be configured to capture or compare counter value, generate a toggling output, or PWM functions. Except for the PWM function, the other TCM functions can generate an interrupt when an event occurs.

Every TCM is connected to a port pin in Port 4; the TCM pin can be configured as an event input, a PWMs, a Toggle Output, or as External Clock Input. The pins are general I/O pins when not assigned to the TCM.

The TCM operation is configured by Control registers and Capture/Compare registers. *Table 143 on page 182* lists the SFR registers in the PCA blocks.

Figure 57. PCA0 block diagram

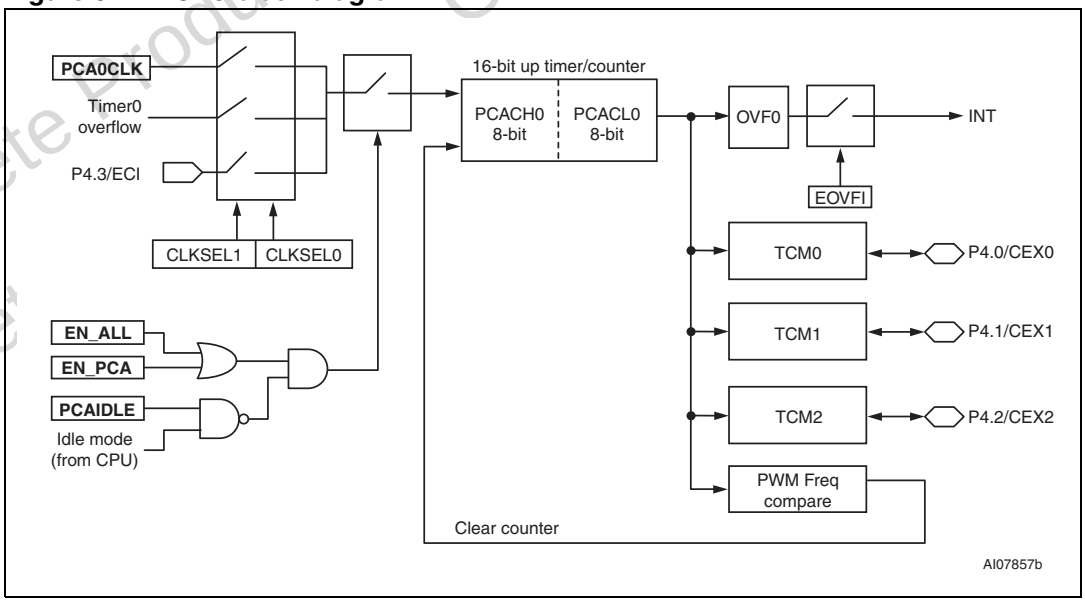


Table 158. General I/O pins on PSD module

Pkg	Port A	Port B	Port C	Port D	Total
52-pin	0	8	4	1	13
80-pin	8	8	4	2	22

Note: Four pins on Port C are dedicated to JTAG, leaving four pins for general I/O.

Each I/O pin on the PSD module can be individually configured for different functions on a pin-by-pin basis (*Figure 79 on page 232*). Following are the available functions on PSD module I/O pins.

- **MCU I/O:** 8032 controls the output state of each port pin or it reads input state of each port pin, by accessing csiop registers at run-time. The direction (in or out) of each pin is also controlled by csiop registers at run-time.
- **PLD I/O:** PSDsoft Express logic equations and pin configuration selections determine if pins are connected to OMC outputs or IMC inputs. This is a static and non-volatile configuration. Port pins connected to PLD outputs can no longer be driven by the 8032 using MCU I/O output mode.
- **Latched MCU Address Output:** Port A or Port B can output de-multiplexed 8032 address signals A0 - A7 on a pin-by-pin basis as specified in csiop registers at run-time. In addition, Port B can also be configured to output de-multiplexed A8-A15 in PSDsoft Express.
- **Data Bus Repeater:** Port A can bidirectionally buffer the 8032 data bus (de-multiplexed) for a specified address range in PSDsoft Express. This is referred to as ***Peripheral I/O Mode*** in this document.
- **Open Drain Outputs:** Some port pins can function as open-drain as specified in csiop registers at run-time.
- Pins on Port D can be used for **external chip-select** outputs originating from the DPLD, without consuming OMC resources within the GPLD.

28.1.15 JTAG port

In-System Programming (ISP) can be performed through the JTAG signals on Port C. This serial interface allows programming of the entire PSD module device or subsections of the PSD module (for example, only Flash memory but not the PLDs) without the participation of the 8032. A blank UPSD34xx device soldered to a circuit board can be completely programmed in 10 to 25 seconds. The four basic JTAG signals on Port C; TMS, TCK, TDI, and TDO form the IEEE-1149.1 interface. The PSD module does not implement the IEEE-1149.1 Boundary Scan functions, but uses the JTAG interface for ISP an 8032 debug. The PSD module can reside in a standard JTAG chain with other JTAG devices and it will remain in BYPASS mode when other devices perform JTAG functions.

ISP programming time can be reduced as much as 30% by using two optional JTAG signals on Port C, TSTAT and $\overline{\text{TERR}}$, in addition to TMS, TCK, TDI and TDO, and this is referred to as “6-pin JTAG”. The FlashLINK JTAG programming cable is available from STMicroelectronics and PSDsoft Express software is available at no charge from www.st.com/psm. More JTAG ISP information maybe found in *Section 28.6.1: JTAG ISP and JTAG debug on page 257*.

The MCU module is also included in the JTAG chain within the UPSD34xx device for 8032 debugging and emulation. While debugging, the PSD module is in BYPASS mode. Conversely, during ISP, the MCU module is in BYPASS mode.

Table 183. MCU I/O mode port B data out register (address = csiop + offset 05h)(1)(2)

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

1. For each bit, 1 = drive port pin to logic '1,' 0 = drive port pin to logic '0'.
2. Default state of register is 00h after reset or power-up.

Table 184. MCU I/O mode port C data out register (address = csiop + offset 12h)(1)(2)

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

1. For each bit, 1 = drive port pin to logic '1,' 0 = drive port pin to logic '0'.
2. Default state of register is 00h after reset or power-up.

Table 185. MCU I/O mode port D data out register (address = csiop + offset 13h)(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)	PD1	N/A

1. For each bit, 1 = drive port pin to logic '1,' 0 = drive port pin to logic '0'.
2. Default state of register is 00h after reset or power-up.
3. Not available on 52-pin UPSD34xx devices.

Table 186. MCU I/O mode port A direction register(1) (address = csiop + offset 06h)(2)(3)

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

1. Port A not available on 52-pin UPSD34xx devices.
2. For each bit, 1 = out from UPSD34xx port pin1, 0 = in to PSD34xx port pin.
3. Default state for register is 00h after reset or power-up.

Table 187. MCU I/O mode port B direction in register (address = csiop + offset 07h)(1)(2)

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

1. For each bit, 1 = out from UPSD34xx port pin1, 0 = in to PSD34xx port pin.
2. Default state for register is 00h after reset or power-up.

Table 188. MCU I/O mode port C direction register (address = csiop + offset 14h)(1)(2)

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

1. For each bit, 1 = out from UPSD34xx port pin1, 0 = in to PSD34xx port pin.
2. Default state for register is 00h after reset or power-up.

Table 191. Latched address output, port B contro register(1) (2)(address = csiop + offset 03h)l

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PB7 (addr A7 or A15)	PB6 (addr A6 or A14)	PB5 (addr A5 or A13)	PB4 (addr A4 or A12)	PB3 (addr A3 or A11)	PB2 (Addr A2 or A10)	PB1 (addr A1 or A9)	PB0 (addr A0 or A8)

1. Default state for register is 00h after reset or power-up.
2. For each bit, 1 = drive demuxed 8032 address signal on pin, 0 = pin is default mode, MCU I/O.

28.5.41 Peripheral I/O mode

This mode will provide a data bus repeater function for the 8032 to interface with external parallel peripherals. The mode is only available on Port A (80-pin devices only) and the data bus signals, D0 - D7, are de-multiplexed (no address A0-A7). When active, this mode behaves like a bidirectional buffer, with the direction automatically controlled by the 8032 $\overline{RD}$ and $\overline{WR}$ signals for a specified address range. The DPLD signals PSEL0 and PSEL1 determine this address range. *Figure 79 on page 232* shows the action of Peripheral I/O mode on the Output Enable logic of the tri-state output driver for a single port pin. *Figure 83 on page 239* illustrates data repeater the operation. To activate this mode, choose the pin function “Peripheral I/O Mode” in PSDsoft Express on any Port A pin (all eight pins of Port A will automatically change to this mode). Next in PSDsoft, specify an address range for the PSELx signals in the “Chip-Select” section of the “Design Assistant”. The user can specify an address range for either PSEL0 or PSEL1. Always qualify the PSELx equation with “$\overline{PSEN}$ is logic ‘1’” to ensure Peripheral I/O mode is only active during 8032 data cycles, not code cycles. Only one equation is needed since PSELx signals are OR’ed together (*Figure 83*). Then in the 8032 initialization firmware, a logic ’1’ is written to the csiop VM register, Bit 7 (PIO_EN) as shown in *Table 154 on page 189*. After this, Port A will automatically perform this repeater function whenever the 8032 presents an address (and memory page number, if paging is used) that is within the range specified by PSELx. Once Port A is designated as Peripheral I/O mode in PSDsoft Express, it cannot be used for other functions.

Note: The user can alternatively connect an external parallel peripheral to the standard 8032 AD0-AD7 pins on an 80-pin uPSD device (not Port A), but these pins have multiplexed address and data signals, with a weaker fanout drive capability.

Figure 83. Peripheral I/O mode

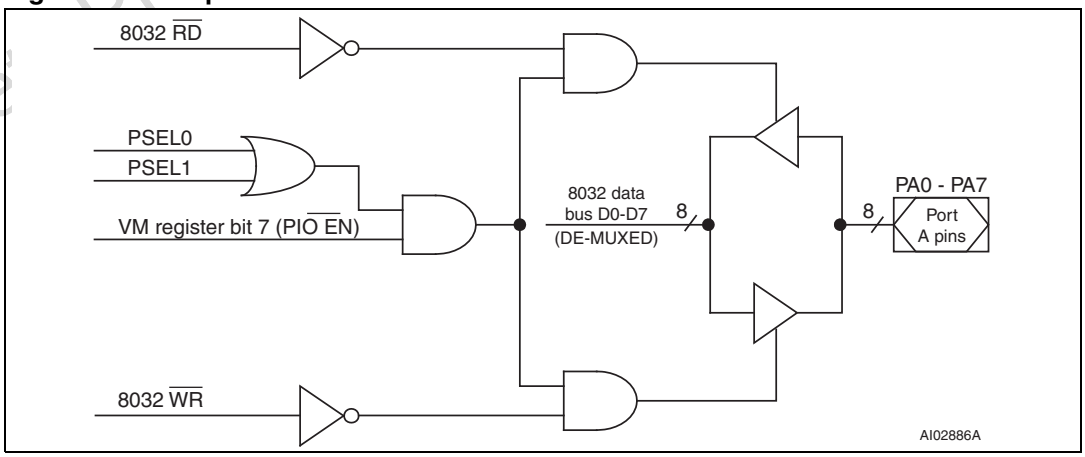
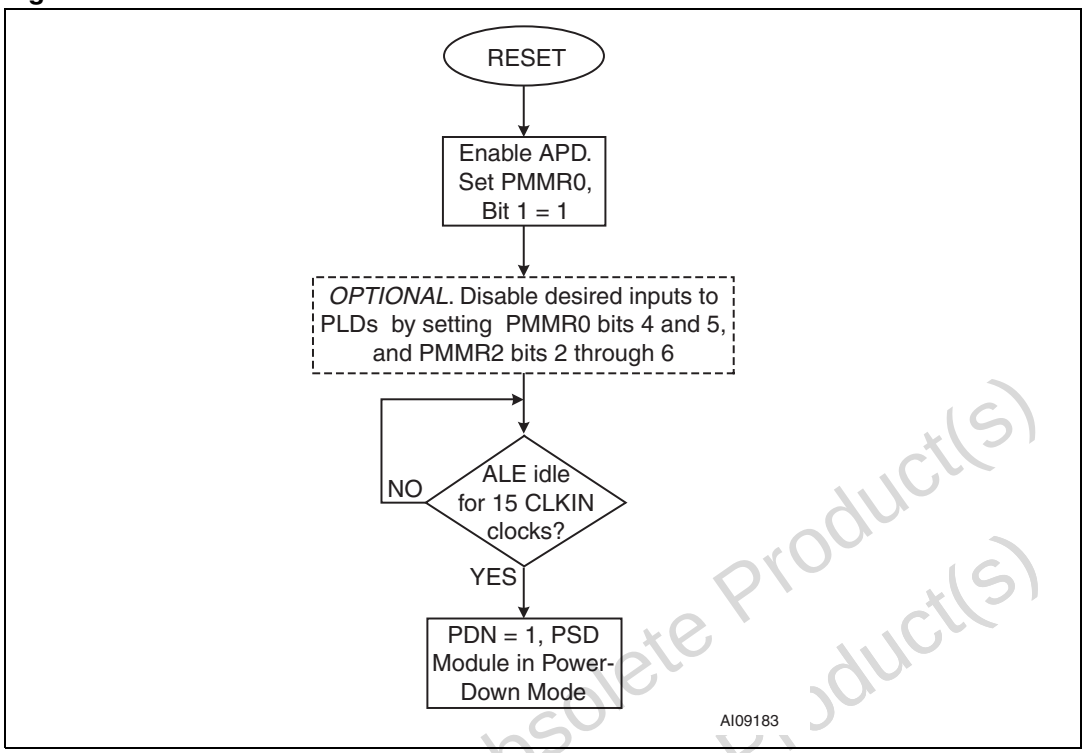


Figure 89. Power-down mode flowchart



28.5.55 Chip select input ($\overline{CSI}$)

Pin PD2 of Port D can optionally be configured in PSDsoft Express as the PSD module Chip Select Input, $\overline{CSI}$, which is an active-low logic input. By default, pin PD2 does not have the $\overline{CSI}$ function.

When the $\overline{CSI}$ function is specified in PSDsoft Express, the $\overline{CSI}$ signal is automatically included in DPLD chip select equations for FSx, CSBOOTx, RS0, and CSIOP. When the $\overline{CSI}$ pin is driven to logic '0' from an external device, all of these memories will be available for READ and WRITE operations. When $\overline{CSI}$ is driven to logic '1,' none of these memories are available for selection, regardless of the address activity from the 8032, reducing power consumption. The state of the PLD and port I/O pins are not changed when $\overline{CSI}$ goes to logic '1' (disabled).

28.5.56 PLD non-turbo mode

The power consumption and speed of the PLDs are controlled by the Turbo Bit (Bit 3) in the csiop PMMR0 register. By setting this bit to logic '1,' the Turbo mode is turned off and both PLDs consume only standby current when ALL PLD inputs have no transitions for an extended time (65 ns for 5 V devices, 100 ns for 3.3 V devices), significantly reducing current consumption. The PLDs will latch their outputs and go to standby, drawing very little current. When Turbo mode is off, PLD propagation delay time is increased as shown in the AC specifications for the PSD module. Since this additional propagation delay also effects the DPLD, the response time of the memories on the PSD module is also lengthened by that same amount of time. If Turbo mode is off, the user should add an additional wait state to the 8032 BUSCON SFR register if the 8032 clock frequency is higher that a particular value. Please refer to *Table 51 on page 90* in the MCU module section.

Figure 96. PLD I_{CC} / frequency consumption (3 V range)

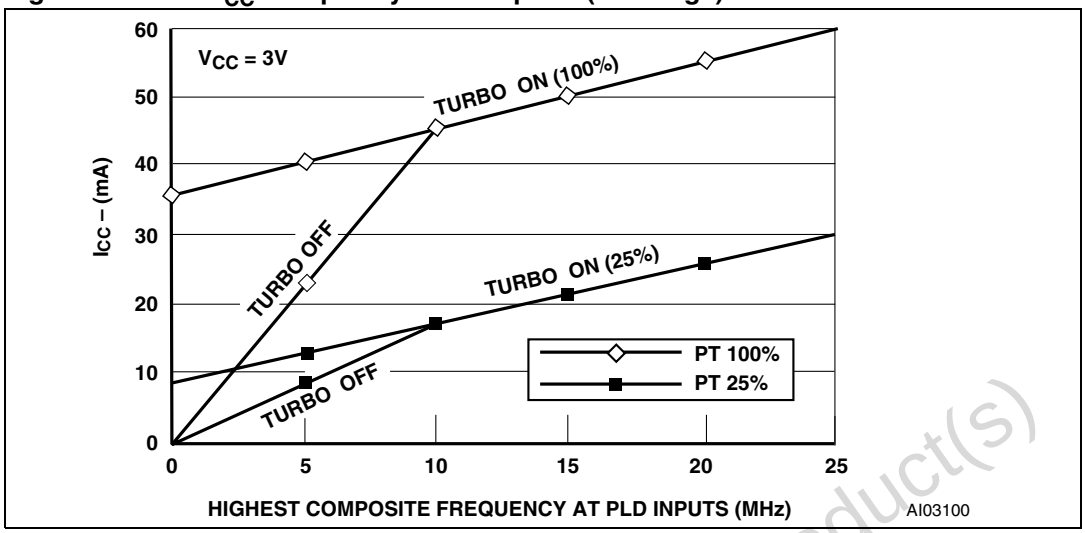


Table 204. PSD module example, typ. power calculation at V_{CC} = 5.0 V (turbo mode off)

Conditions		
	MCU clock frequency	= 12 MHz
Highest composite PLD input frequency		
	(Freq PLD)	= 8 MHz
MCU ALE frequency (Freq ALE)		= 2 MHz
	% Flash memory access	= 80%
	% SRAM access	= 15%
	% I/O access	= 5% (no additional power above base)
Operational modes		
	% Normal	= 40%
	% Power-down mode	= 60%
Number of product terms used		
	(from fitter report)	= 45 PT
	% of total product terms	= 45/182 = 24.7%
	Turbo mode	= Off
Calculation (using typical values)		
I_{CC} total	= I_{CC}(MCUactive) x %MCUactive + I_{CC}(PSDactive) x %PSDactive + I_{PD}(pwrdown) x %pwrdown	

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).