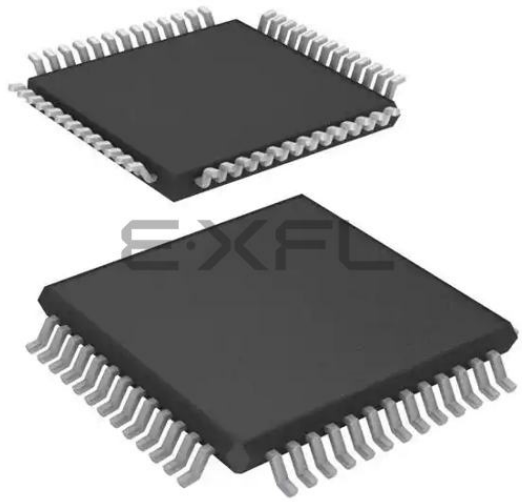


Welcome to E-XFL.COM

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	35
Program Memory Size	80KB (80K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	52-TQFP
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/upsd3422e-40t6

1 Description

The *Turbo Plus* UPSD34xx Series combines a powerful 8051-based microcontroller with a flexible memory structure, programmable logic, and a rich peripheral mix to form an ideal embedded controller. At its core is a fast 4-cycle 8032 MCU with a 4-byte instruction prefetch queue (PFQ) and a 4-entry fully associative branching cache (BC). The MCU is connected to a 16-bit internal instruction path to maximize performance, enabling loops of code in smaller localities to execute extremely fast. The 16-bit wide instruction path in the *Turbo Plus* Series allows double-byte instructions to be fetched from memory in a single memory cycle. This keeps the average performance near its peak performance (peak performance for 5 V, 40 MHz Turbo *Plus* UPSD34xx is 10 MIPS for single-byte instructions, and average performance will be approximately 9 MIPS for mix of single- and multi-byte instructions).

USB 2.0 (full speed, 12Mbps) is included, providing 10 endpoints, each with its own 64-byte FIFO to maintain high data throughput. Endpoint 0 (control endpoint) uses two of the 10 endpoints for In and Out directions, the remaining eight endpoints may be allocated in any mix to either type of transfers: Bulk or Interrupt.

Code development is easily managed without a hardware in-circuit emulator by using the serial JTAG debug interface. JTAG is also used for in-system programming (ISP) in as little as 10 seconds, perfect for manufacturing and lab development. The 8032 core is coupled to programmable system device (PSD) architecture to optimize the 8032 memory structure, offering two independent banks of Flash memory that can be placed at virtually any address within 8032 program or data address space, and easily paged beyond 64 Kbytes using on-chip programmable decode logic.

Dual Flash memory banks provide a robust solution for remote product updates in the field through in-application programming (IAP). Dual Flash banks also support EEPROM emulation, eliminating the need for external EEPROM chips.

General-purpose programmable logic (PLD) is included to build an endless variety of glue-logic, saving external logic devices. The PLD is configured using the software development tool, PSDsoft Express, available from the web at **www.st.com/psm**, at no charge.

The UPSD34xx also includes supervisor functions such as a programmable watchdog timer and low-voltage reset.

Note: For a list of known limitations of the UPSD34xx devices, please refer to Section 34: Important notes.

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)	

3 Hardware description

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

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

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

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

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

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

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

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

Table 3. Port type and voltage source combinations

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

13 Interrupt system

The UPSD34xx has an 12-source, two priority level interrupt structure summarized in *Table 17*.

Firmware may assign each interrupt source either high or low priority by writing to bits in the SFRs named, IP and IPA, shown in *Table 17*. An interrupt will be serviced as long as an interrupt of equal or higher priority is not already being serviced. If an interrupt of equal or higher priority is being serviced, the new interrupt will wait until it is finished before being serviced. If a lower priority interrupt is being serviced, it will be stopped and the new interrupt is serviced. When the new interrupt is finished, the lower priority interrupt that was stopped will be completed. If new interrupt requests are of the same priority level and are received simultaneously, an internal polling sequence determines which request is selected for service. Thus, within each of the two priority levels, there is a second priority structure determined by the polling sequence.

Firmware may individually enable or disable interrupt sources by writing to bits in the SFRs named, IE and IEA, shown in *Table 17 on page 63*. The SFR named IE contains a global disable bit (EA), which can be cleared to disable all 12 interrupts at once, as shown in *Table 18 on page 66*. *Figure 12 on page 64* illustrates the interrupt priority, polling, and enabling process.

Each interrupt source has at least one interrupt flag that indicates whether or not an interrupt is pending. These flags reside in bits of various SFRs shown in *Table 17 on page 63*.

All of the interrupt flags are latched into the interrupt control system at the beginning of each MCU machine cycle, and they are polled at the beginning of the following machine cycle. If polling determines one of the flags was set, the interrupt control system automatically generates an LCALL to the user's Interrupt Service Routine (ISR) firmware stored in program memory at the appropriate vector address.

- The specific vector address for each of the interrupt sources are listed in *Table 17 on page 63*. However, this LCALL jump may be blocked by any of the following conditions:
- An interrupt of equal or higher priority is already in progress
- The current machine cycle is not the final cycle in the execution of the instruction in progress
- The current instruction involves a write to any of the SFRs: IE, IEA, IP, or IPA
- The current instruction is an RETI

Note: Interrupt flags are polled based on a sample taken in the previous MCU machine cycle. If an interrupt flag is active in one cycle but is denied serviced due to the conditions above, and then later it is not active when the conditions above are finally satisfied, the previously denied interrupt will not be serviced. This means that active interrupts are not remembered. Every poling cycle is new.

Assuming all of the listed conditions are satisfied, the MCU executes the hardware generated LCALL to the appropriate ISR. This LCALL pushes the contents of the PC onto the stack (but it does not save the PSW) and loads the PC with the appropriate interrupt vector address. Program execution then jumps to the ISR at the vector address.

Execution precedes in the ISR. It may be necessary for the ISR firmware to clear the pending interrupt flag for some interrupt sources, because not all interrupt flags are automatically cleared by hardware when the ISR is called, as shown in *Table 17 on page 63*.

13.1.2 Timer 0 and 1 overflow interrupt

Timer 0 and Timer 1 interrupts are generated by the flag bits TF0 and TF1 when there is an overflow condition in the respective Timer/Counter register (except for Timer 0 in Mode 3).

13.1.3 Timer 2 overflow interrupt

This interrupt is generated to the MCU by a logical OR of flag bits, TF2 and EXE2. The ISR must read the flag bits to determine the cause of the interrupt.

- TF2 is set by an overflow of Timer 2.
- EXE2 is generated by the falling edge of a signal on the external pin, T2X (pin P1.1).

13.1.4 UART0 and UART1 interrupt

Each of the UARTs have identical interrupt structure. For each UART, a single interrupt is generated to the MCU by the logical OR of the flag bits, RI (byte received) and TI (byte transmitted).

The ISR must read flag bits in the SFR named SCON0 for UART0, or SCON1 for UART1 to determine the cause of the interrupt.

13.1.5 SPI interrupt

The SPI interrupt has four interrupt sources, which are logically ORed together when interrupting the MCU. The ISR must read the flag bits to determine the cause of the interrupt.

A flag bit is set for: end of data transmit (TEISF); data receive overrun (RORISF); transmit buffer empty (TISF); or receive buffer full (RISF).

13.1.6 I^2C interrupt

The flag bit INTR is set by a variety of conditions occurring on the I^2C interface: received own slave address (ADDR flag); received general call address (GC flag); received Stop condition (STOP flag); or successful transmission or reception of a data byte.The ISR must read the flag bits to determine the cause of the interrupt.

13.1.7 ADC interrupt

The flag bit AINTF is set when an A-to-D conversion has completed.

13.1.8 PCA interrupt

The PCA has eight interrupt sources, which are logically ORed together when interrupting the MCU.The ISR must read the flag bits to determine the cause of the interrupt.

- Each of the six TCMs can generate a "match or capture" interrupt on flag bits OFV5..0 respectively.
- Each of the two 16-bit counters can generate an overflow interrupt on flag bits INTF1 and INTF0 respectively.

Tables *18* through *Table 24 on page 67* have detailed bit definitions of the interrupt system SFRs.

15 Power saving modes

The UPSD34xx is a combination of two die, or modules, each module having its own current consumption characteristics. This section describes reduced power modes for the MCU module. See *Section 28.1.16: Power management on page 197* for reduced power modes of the PSD module. Total current consumption for the combined modules is determined in the DC specifications at the end of this document.

The MCU module has three software-selectable modes of reduced power operation.

- Idle mode
- Power-down mode
- Reduced frequency mode

15.1 Idle mode

Idle mode will halt the 8032 MCU core while leaving the MCU peripherals active (Idle mode blocks MCU_CLK only). For lowest current consumption in this mode, it is recommended to disable all unused peripherals, before entering Idle mode (such as the ADC and the Debug Unit breakpoint comparators). The following functions remain fully active during Idle mode (except if disabled by SFR settings).

- External Interrupts INT0 and INT1
- Timer 0, Timer 1 and Timer 2
- Supervisor reset from: LVD, JTAG Debug, External RESET_IN_, but **not** the WTD
- ADC
- I^2C Interface
- UART0 and UART1 Interfaces
- SPI Interface
- Programmable Counter Array
- USB Interface

An interrupt generated by any of these peripherals, or a reset generated from the supervisor, will cause Idle mode to exit and the 8032 MCU will resume normal operation.

The output state on I/O pins of MCU ports 1, 3, and 4 remain unchanged during Idle mode.

To enter Idle mode, the 8032 MCU executes an instruction to set the IDL bit in the SFR named PCON, shown in *Table 33 on page 74*. This is the last instruction executed in normal operating mode before Idle mode is activated. Once in Idle mode, the MCU status is entirely preserved, and there are no changes to: SP, PSW, PC, ACC, SFRs, DATA, IDATA, or XDATA.

The following are factors related to Idle mode exit:

- Activation of any enabled interrupt will cause the IDL bit to be cleared by hardware, terminating Idle mode. The interrupt is serviced, and following the Return from Interrupt instruction (RETI), the next instruction to be executed will be the one which follows the instruction that set the IDL bit in the PCON SFR.
- After a reset from the supervisor, the IDL bit is cleared, Idle mode is terminated, and the MCU restarts after three MCU machine cycles.

16 Oscillator and external components

The oscillator circuit of UPSD34xx devices is a single stage, inverting amplifier in a Pierce oscillator configuration. The internal circuitry between pins XTAL1 and XTAL2 is basically an inverter biased to the transfer point. Either an external quartz crystal or ceramic resonator can be used as the feedback element to complete the oscillator circuit. Both are operated in parallel resonance. Ceramic resonators are lower cost, but typically have a wider frequency tolerance than quartz crystals. Alternatively, an external clock source from an oscillator or other active device may drive the UPSD34xx oscillator circuit input directly, instead of using a crystal or resonator.

The minimum frequency of the quartz crystal, ceramic resonator, or external clock source is 3MHz if the USB is used. The minimum is 8MHz if I^2C is used. The maximum is 40MHz in all cases. This frequency is f_{OSC}, which can be divided internally as described in *Section 14: MCU clock generation on page 68*.

The pin XTAL1 is the high gain amplifier input, and XTAL2 is the output. To drive the UPSD34xx device externally from an oscillator or other active device, XTAL1 is driven and XTAL2 is left open-circuit. This external source should drive a logic low at the voltage level of 0.3 V_{CC} or below, and logic high at 0.7V V_{CC} or above, up to 5.5 V V_{CC}. The XTAL1 input is 5 V tolerant.

Most of the quartz crystals in the range of 25MHz to 40 MHz operate in the third overtone frequency mode. An external LC tank circuit at the XTAL2 output of the oscillator circuit is needed to achieve the third overtone frequency, as shown in *Figure 14 on page 77*. Without this LC circuit, the crystal will oscillate at a fundamental frequency mode that is about 1/3 of the desired overtone frequency.

Note: In Figure 14 on page 77 crystals which are specified to operate in fundamental mode (not overtone mode) do not need the LC circuit components. Since quartz crystals and ceramic resonators have their own characteristics based on their manufacturer, it is wise to also consult the manufacturer's recommended values for external components.

Figure 32. UART mode 1, block diagram

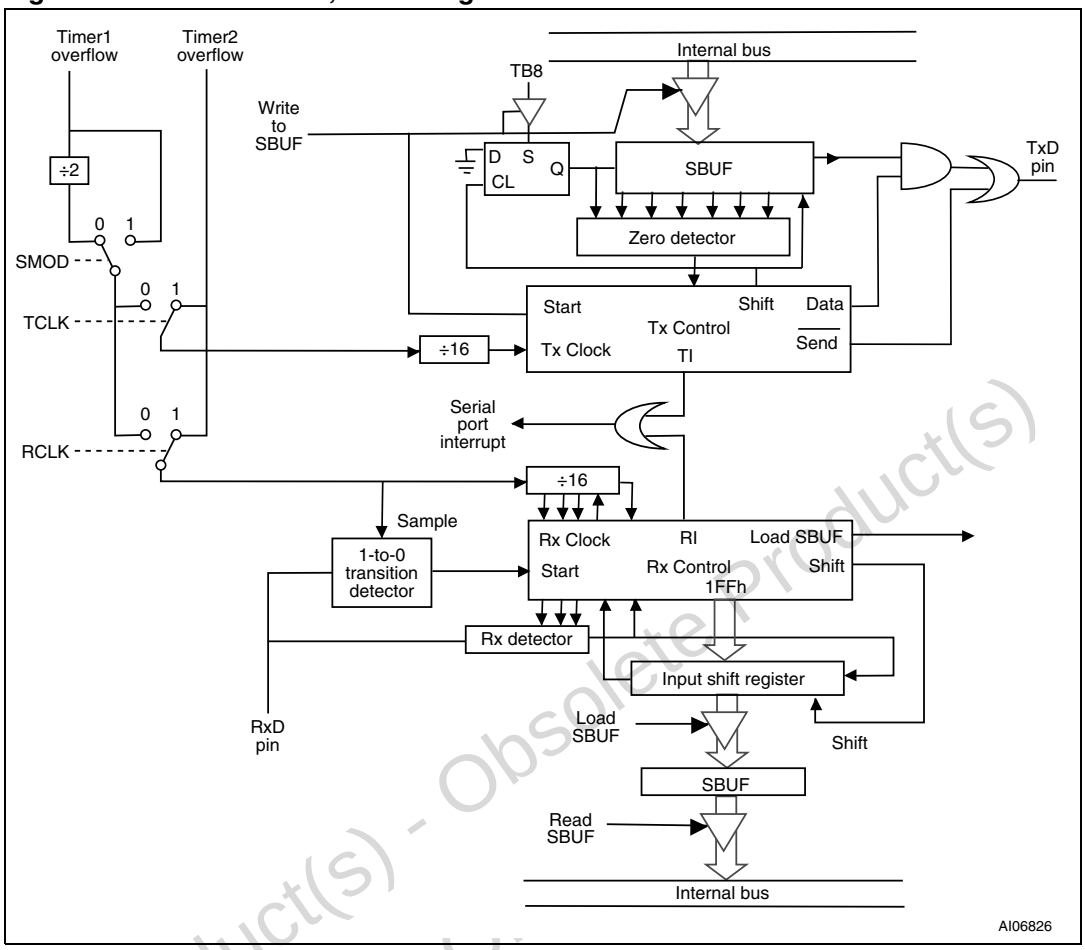


Figure 33. UART mode 1, timing diagram

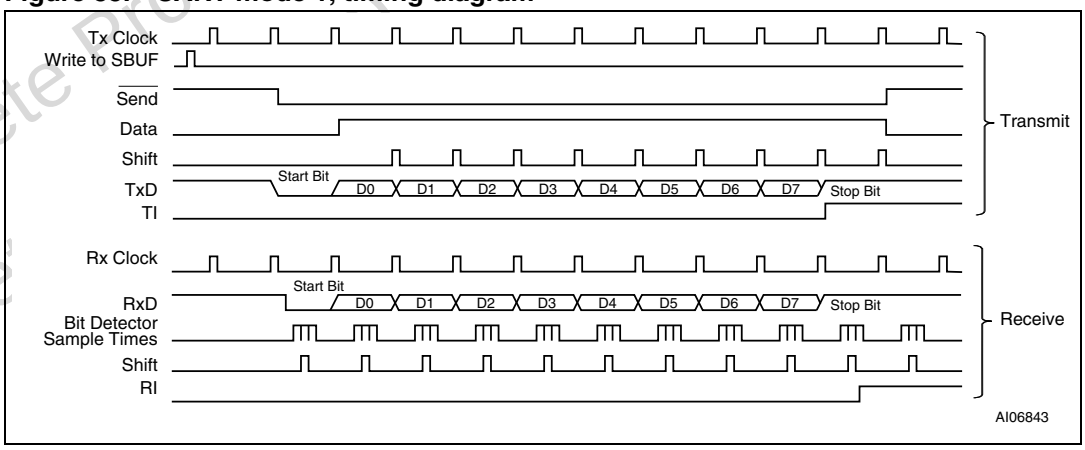


Table 74. Recommended CDIV[4:0] values to generate SIRClk (default CDIV[4:0] = 0Fh, 15 decimal)

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.

23 I²C interface

UPSD34xx devices support one serial I²C interface. This is a two-wire communication channel, having a bidirectional data signal (SDA, pin P3.6) and a clock signal (SCL, pin P3.7) based on open-drain line drivers, requiring external pull-up resistors, R_P, each with a typical value of 4.7kΩ (see *Figure 40*).

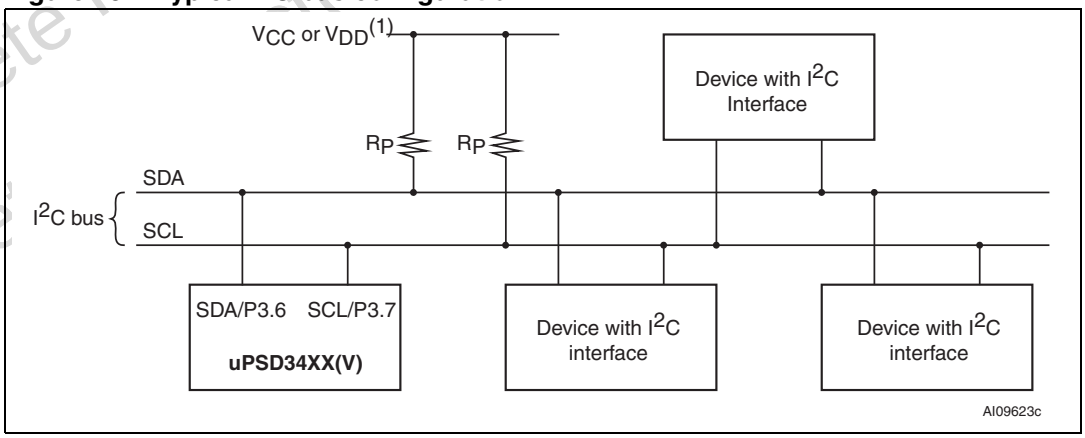
23.1 I²C interface main features

Byte-wide data is transferred, MSB first, between a Master device and a Slave device on two wires. More than one bus Master is allowed, but only one Master may control the bus at any given time. Data is not lost when another Master requests the use of a busy bus because I²C supports collision detection and arbitration. The bus Master initiates all data movement and generates the clock that permits the transfer. Once a transfer is initiated by the Master, any device addressed is considered a Slave. Automatic clock synchronization allows I²C devices with different bit rates to communicate on the same physical bus. A single device can play the role of Master or Slave, or a single device can be a Slave only. Each Slave device on the bus has a unique address, and a general broadcast address is also available. A Master or Slave device has the ability to suspend data transfers if the device needs more time to transmit or receive data.

This I²C interface has the following features:

- Serial I/O Engine (SIOE): serial/parallel conversion; bus arbitration; clock generation and synchronization; and handshaking are all performed in hardware
- Interrupt or Polled operation
- Multi-master capability
- 7-bit Addressing
- Supports standard speed I²C (SCL up to 100kHz), fast mode I²C (101KHz to 400kHz), and high-speed mode I²C (401KHz to 833kHz)

Figure 40. Typical I²C bus configuration



1. For 3.3 V system, connect R_P to 3.3 V V_{CC}. For 5.0 V system, connect R_P to 5.0 V V_{DD}.

23.8 I²C interface control register (S1CON)

Table 75. Serial control register S1CON (SFR DCh, reset value 00h)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CR2	ENI1	STA	STO	ADDR	AA	CR[1:0]	

Table 76. S1CON register bit definition

Bit	Symbol	R/W	Function
7	CR2	R,W	This bit, along with bits CR1 and CR0, determine the SCL clock frequency (f_{SCL}) when SIOE is in Master mode. These bits create a clock divisor for f_{OSC}. See *Table 77*.
6	ENI1	R,W	I²C Interface Enable 0 = SIOE disabled, 1 = SIOE enabled. When disabled, both SDA and SCL signals are in high impedance state.
5	STA	R,W	START flag. When set, Master mode is entered and SIOE generates a Start condition only if the I²C bus is not busy. When a Start condition is detected on the bus, the STA flag is cleared by hardware. When the STA bit is set during an interrupt service, the Start condition will be generated after the interrupt service.
4	STO	R,W	STOP flag When STO is set in Master mode, the SIOE generates a Stop condition. When a Stop condition is detected, the STO flag is cleared by hardware. When the STO bit is set during an interrupt service, the Stop condition will be generated after the interrupt service.
3	ADDR	R,W	This bit is set when an address byte received in Slave mode matches the device address programmed into the S1ADR register. The ADDR bit must be cleared with firmware.
2	AA	R,W	Assert Acknowledge enable If AA = 1, an acknowledge signal (low on SDA) is automatically returned during the acknowledge bit-time on the SCL line when any of the following three events occur: 1. SIOE in Slave mode receives an address that matches contents of S1ADR register 2. A data byte has been received while SIOE is in Master Receiver mode 3. A data byte has been received while SIOE is a selected Slave Receiver When AA = 0, no acknowledge is returned (high on SDA during acknowledge bit-time).
1, 0	CR1, CR0	R,W	These bits, along with bit CR2, determine the SCL clock frequency (f_{SCL}) when SIOE is in Master mode. These bits create a clock divisor for f_{OSC}. See *Table 77* for values.

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

I²C bus speed, f_{SCL}	Parameter	Oscillator frequency, f_{OSC}				
		6 MHz	12 MHz	24 MHz	33 MHz	40 MHz
Fast	Recommended S1SETUP value	82h	85h	8Bh	90h	93h
	Number of samples	3	6	12	17	20
	Time between samples	166.6 ns	83.3 ns	41.6 ns	30 ns	25 ns
	Total sampled period	500 ns	500 ns	500 ns	510 ns	500 ns
High	Recommended S1SETUP value	(1)	80	82	83	84
	Number of samples	-	1	3	4	5
	Time between samples	-	83.3 ns	41.6 ns	30 ns	25 ns
	Total sampled period	-	83.3	125 ns	120 ns	125 ns

1. Not compatible with High Speed I²C.

23.13 I²C operating sequences

The following pseudo-code explains hardware control for these I²C functions on the UPSD34xx:

- Initialize the Interface
- Function as Master-Transmitter
- Function as Master-Receiver
- Function as Slave-Transmitter
- Function as Slave-Receiver
- Interrupt Service Routine

Full C code drivers for the UPSD34xx I²C interface, and other interfaces are available from the web at www.st.com\psm.

Initialization after a UPSD34xx reset

```
Ensure pins P3.6 and P3.7 are GPIO inputs
    -   SFR P3.7 = 1 and SFR P3.6 = 1
Configure pins P3.6 and P3.7 as I2C
    -   SFR P3SFS.6 = 1 and P3SFS.7 = 1
Set I2C clock prescaler to determine fSCL
    -   SFR S1CON.CR[2:0] = desired SCL freq.
Set bus Start condition sampling
    -   SFR S1SETUP[7:0] = number of samples
```

- USB FIFO base address high and low registers (UBASEH and UBASEL)

  All 10 Endpoint FIFOs share the same 64-byte address range. The 16-bit base address for the FIFOs is specified using the USB base address registers (see *Table 130* and *Table 132*). The USB endpoint select register (see *Table 124 on page 172*) selects the direction and the Endpoint for the FIFO that is accessed when addressing the 64-bytes of XDATA space starting with the base address specified in the base address registers. The base address is a 64-byte segment where the lower 6 bits of the base register are hardwired to '0.'

  Important note: The USB FIFO base address must be set to an open 64-byte segment in the XDATA space. Care should be taken to ensure that there is no overlap of addresses between the USB FIFOs and the flash memory, SRAM, csiop registers, and anything else accessed in the XDATA space. While the logic in the PSD module handles overlap of flash memory, SRAM, and the csiop registers with a fixed priority (see Section 28.1: PSD module functional description on page 192), this is not the case with the USB FIFOs. Unpredictable results as well as potential damage to the device may occur if there is an overlap of addresses.

Table 130. USB FIFO base address high register (UBASEH 0F3h, reset value 00h)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
BASEADDR[15:8]							

Table 131. UBASEH register bit definition

Bit	Symbol	R/W	Definition
7:0	BASEADDR [15:8]	R/W	The upper 8 bits of the 16-bit base address for USB FIFOs to be mapped in XDATA space

Table 132. USB FIFO base address low register (UBASEL 0F4h, reset value 00h)

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

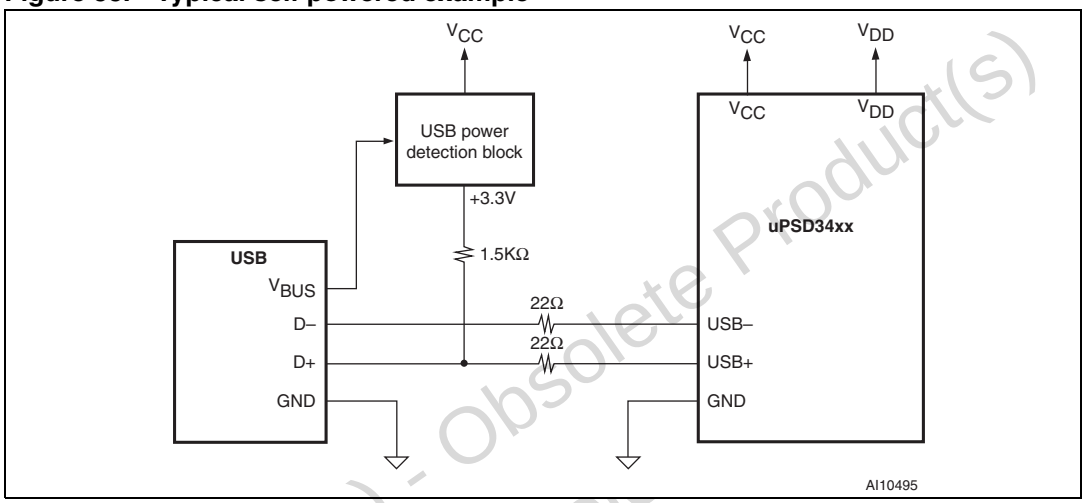
Table 133. UBASEL register bit definition

Bit	Symbol	R/W	Definition
7:6	BASEADDR [7:6]	R/W	Bits 7 and 6 of the 16-bit base address for the USB FIFOs to be mapped in XDATA space
5:0	BASEADDR [5:0]	R	Hardwired '0'

25.5 Typical connection to USB

Connecting the UPSD34xx to the USB is simple and straightforward. *Figure 55* shows a typical self-powered example requiring only three resistors and a USB power detection circuit. The USB power detection circuit detects when the device has been connected to the USB. When V_{BUS} is detected, it switches 3.3 V to the pull-up resistor on the D+ line. Per the USB specification, the pull-up resistor on D+ is required to signal to the upstream USB port when a full speed device has been connected to the bus. The resistors in series in the D+ and D– lines are recommended per the USB specification to reduce transients on the data lines.

Figure 55. Typical self powered example



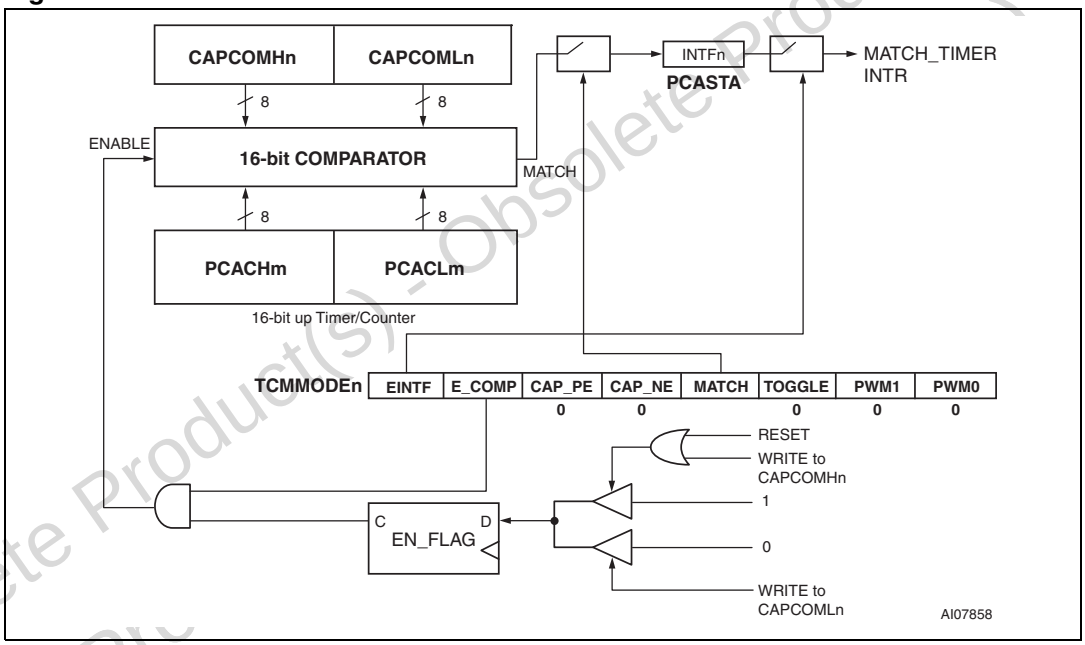
27.6 Toggle mode

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

27.7 PWM mode - (x8), fixed frequency

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

Figure 58. Timer mode



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

Table 155. TCMMODE0 - TCMMODE5 register bit definition (continued)

Bit	Symbol	Function
2	TOGGLE	1 - A match on the comparator results in a toggling output on CEXn pin.
1-0	PWM[1:0]	01 Enable PWM Mode (x8), fixed frequency. Enable the CEXn pin as a PWM output. 10 Enable PWM Mode (x8) with programmable frequency. Enable the CEXn pin as a PWM output. 11 Enable PWM Mode (x10 or x16), fixed frequency. Enable the CEXn pin as a PWM output.

Table 156. TCMMODE register configurations

EINTF	E_COMP	CAP_PE	CAP_NE	MATCH	TOGGLE	PWM1	PWM0	TCM FUNCTION
0	0	0	0	0	0	0	0	No operation (reset value)
0	1	0	0	0	0	0	1	8-bit PWM, fixed frequency
0	1	0	0	0	0	1	0	8-bit PWM, programmable frequency
0	1	0	0	0	0	1	1	10-bit or 16-bit PMW, fixed frequency(1)
X	1	0	0	1	1	0	0	16-bit toggle
X	1	0	0	1	0	0	0	16-bit Software Timer
X	X	0	1	0	0	0	0	16-bit capture, negative trigger
X	X	1	0	0	0	0	0	16-bit capture, positive trigger
X	X	1	1	0	0	0	0	16-bit capture, transition trigger

1. 10-bit PWM mode requires the 10B_PWM Bit in the PCACON register set to '1.'

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.

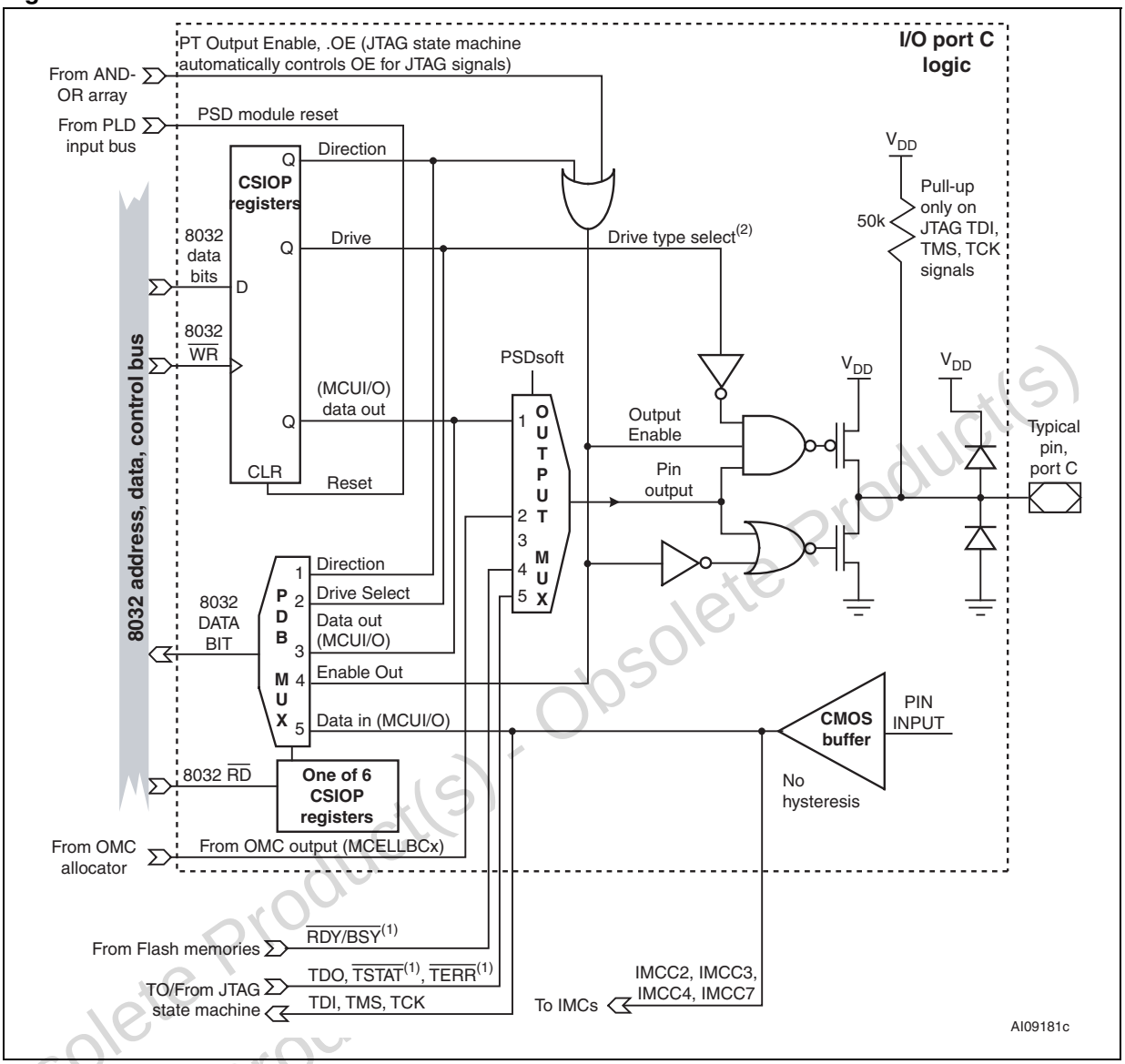
Once the 8032 issues a Flash memory program or erase instruction sequence, it must check the status bits for completion. The embedded algorithms that are invoked inside a Flash memory array provide several ways to give status to the 8032. Status may be checked using any of three methods: Data Polling, 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.

Figure 86. Port C structure



Note: 1 Optional function on a specific Port C pin.

28.5.51 Port D structure

Port D has two I/O pins (PD1, PD2) on 80-pin UPSD34xx devices, and just one pin (PD1) on 52-pin devices, supporting the following operating modes:

- MCU I/O Mode
- DPLD Output Mode for External Chip Selects, ECS1, ECS2. This does not consume OMCs in the GPLD.
- PLD Input Mode – direct input to the PLD Input Bus available to DPLD and GPLD. Does not use IMCs

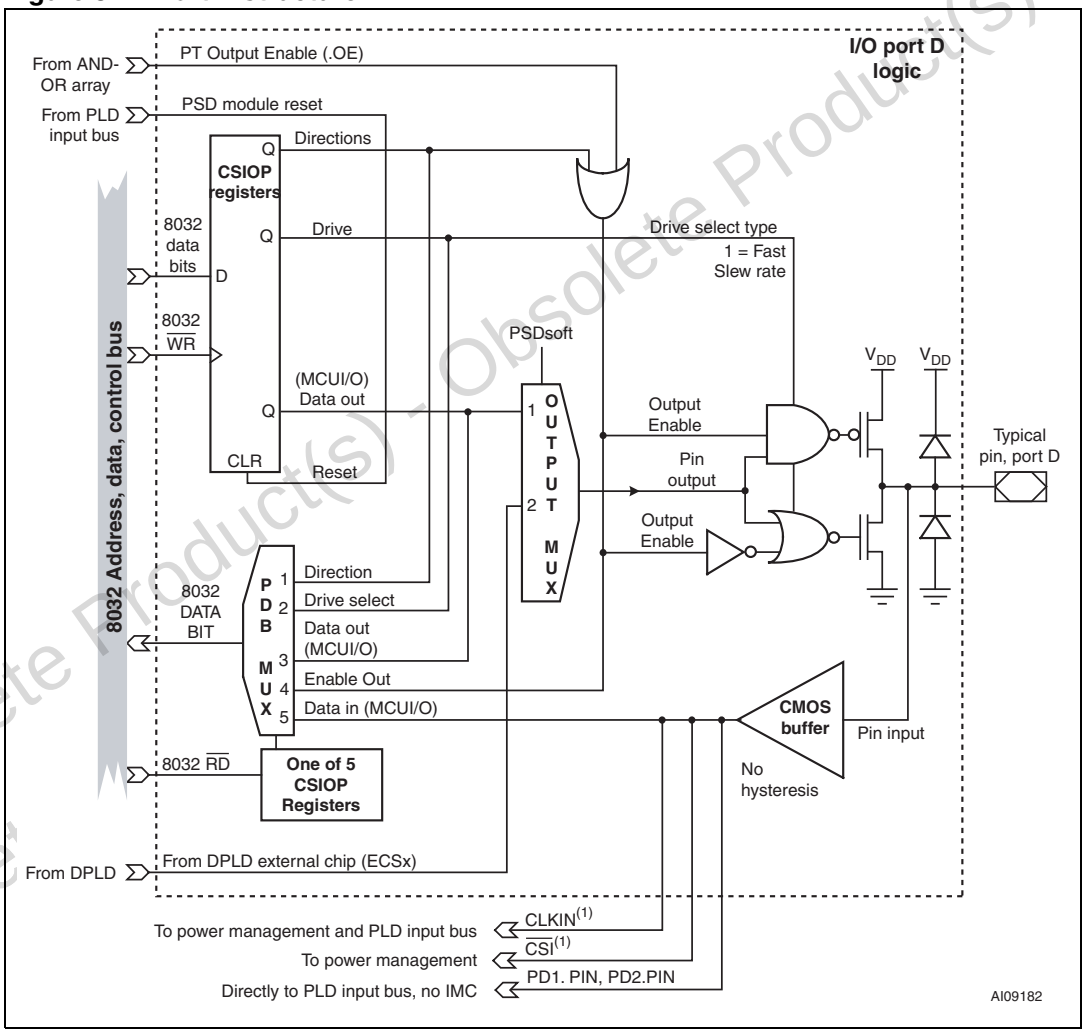
See *Figure 87 on page 247* for detail.

Port D pins can also be configured in PSDsoft as pins for other dedicated functions:

- PD1 can be used as a common clock input to all 16 OMC Flip-flops (see *Section 28.1.11: OMCs on page 195*) and also the *Section 28.5.53: Automatic power-down (APD) on page 250*.
- PD2 can be used as a common chip select signal ($\overline{CSI}$) for the Flash and SRAM memories on the PSD module (see *Section 28.5.55: Chip select input (CSI) on page 253*). If driven to logic '1' by an external source, $\overline{CSI}$ will force the Flash memory into standby mode regardless of what other internal memory select signals are doing on the PSD module. This is specified in PSDsoft as "PSD Chip Select Input, $\overline{CSI}$".

Port D also supports the Fast Slew Rate output drive type option using the csiop Drive Select registers.

Figure 87. Port D structure



Note: 1 Optional function on a specific Port D pin.

Table 234. ISC timing (5 V PSD module) (continued)

Symbol	Parameter	Conditions	Min	Max	Unit
t_{ISCPCO}	ISC port clock to output			21	ns
t_{ISCPZV}	ISC port high-impedance to valid output			21	ns
t_{ISCPVZ}	ISC port valid output to high-impedance			21	ns

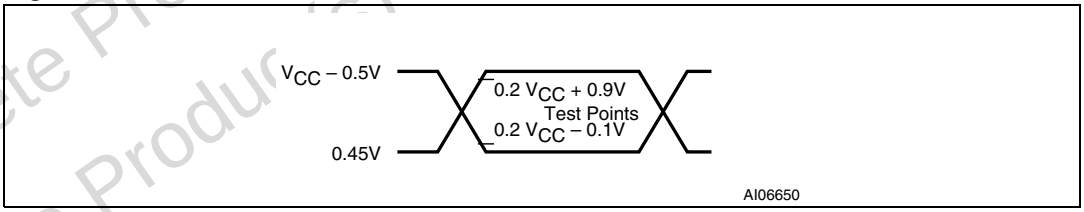
1. For non-PLD programming, erase or in ISC bypass mode.
2. For program or erase PLD only.

Table 235. ISC timing (3 V PSD module)

Symbol	Parameter	Conditions	Min	Max	Unit
t_{ISCCF}	Clock (TCK, PC1) frequency (except for PLD)	(1)		16	MHz
t_{ISCCH}	Clock (TCK, PC1) high time (except for PLD)		40		ns
t_{ISCCL}	Clock (TCK, PC1) low time (except for PLD)		40		ns
t_{ISCCFP}	Clock (TCK, PC1) frequency (PLD only)	(2)		4	MHz
t_{ISCCHP}	Clock (TCK, PC1) high time (PLD only)		90		ns
t_{ISCCLP}	Clock (TCK, PC1) low time (PLD only)		90		ns
t_{ISCPSU}	ISC port setup time		12		ns
t_{ISCPH}	ISC port hold up time		5		ns
t_{ISCPCO}	ISC port clock to output			30	ns
t_{ISCPZV}	ISC port high-impedance to valid output			30	ns
t_{ISCPVZ}	ISC port valid output to high-impedance			30	ns

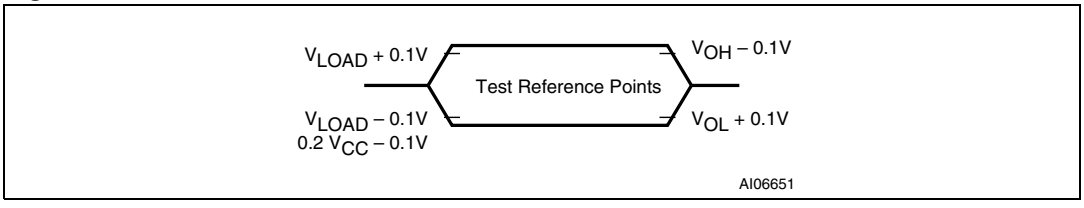
1. For non-PLD programming, erase or in ISC bypass mode.
2. For program or erase PLD only.

Figure 108. MCU module AC measurement I/O waveform



1. AC inputs during testing are driven at V_{CC}–0.5 V for a logic '1,' and 0.45 V for a logic '0.'
2. Timing measurements are made at V_{IH}(min) for a logic '1,' and V_{IL}(max) for a logic '0'

Figure 109. PSD module AC float I/O waveform



1. For timing purposes, a port pin is considered to be no longer floating when a 100 mV change from load voltage occurs, and begins to float when a 100 mV change from the loaded V_{OH} or V_{OL} level occurs
2. I_{OL} and $I_{OH} \geq 20$ mA