Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

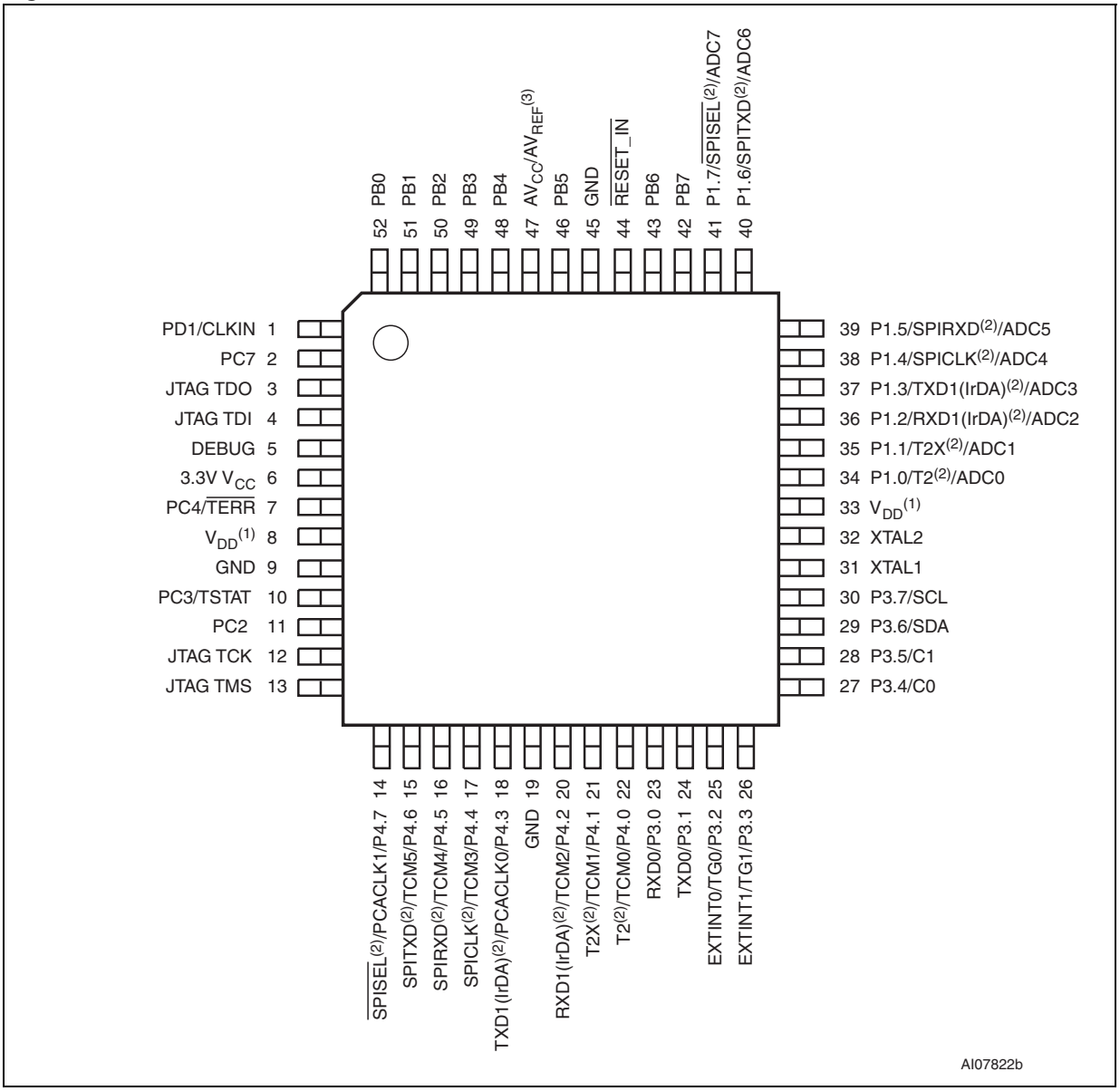
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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
Peripherals	LVD, POR, PWM, WDT
Number of I/O	45
Program Memory Size	160KB (160K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	8K x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 5.5V
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/upsd3333d-40u6

2 Pin descriptions

Figure 2. LQFP52 connections



1. 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.
2. These signals can be used on one of two different ports (Port 1 or Port 4) for flexibility. Default is Port1.
3. AV_{REF} and 3.3 V AV_{CC} are shared in the 52-pin package only. ADC channels must use AV_{CC} as AV_{REF} for the 52-pin package.

8 Special function registers (SFR)

A group of registers designated as Special Function register (SFR) is shown in *Table 5 on page 41*. SFRs control the operating modes of the MCU core and also control the peripheral interfaces and I/O pins on the MCU module. The SFRs can be accessed only by using the Direct Addressing method within the address range from 80h to FFh of internal 8032 SRAM. Sixteen addresses in SFR address space are both byte- and bit-addressable. The bit-addressable SFRs are noted in *Table 5*.

86 of a possible 128 SFR addresses are occupied. The remaining unoccupied SFR addresses (designated as “RESERVED” in *Table 5*) should not be written. Reading unoccupied locations will return an undefined value.

Note: There is a separate set of control registers for the PSD module, designated as csiop, and they are described in the Section 27: PSD module on page 164. The I/O pins, PLD, and other functions on the PSD module are NOT controlled by SFRs.

SFRs are categorized as follows:

- MCU core registers: IP, A, B, PSW, SP, DPTL, DPTH, DPTC, DPTM
- MCU module I/O port registers: P1, P3, P4, P1SFS0, P1SFS1, P3SFS, P4SFS0, P4SFS1
- Standard 8032 timer registers: TCON, TMOD, T2CON, TH0, TH1, TH2, TL0, TL1, TL2, RCAP2L, RCAP2H
- Standard serial interfaces (UART): SCON0, SBUF0, SCON1, SBUF1
- Power, clock, and bus timing registers: PCON, CCON0, BUSCON
- Hardware watchdog timer registers: WDKEY, WDRST
- Interrupt system registers: IP, IPA, IE, IEA
- Program counter array (PCA) control registers: PCACL0, PCACH0, PCACON0, PCASTA, PCACL1, PCACH1, PCACON1, CCON2, CCON3
- PCA capture/compare and PWM registers

  CAPCOML0, CAPCOMH0, TCMMODE0, CAPCOML1, CAPCOMH1, TCMMODE2, CAPCOML2, CAPCOMH2, TCMMODE2, CAPCOML3, CAPCOMH3, TCMMODE3, CAPCOML4, CAPCOMH4, TCMMODE4, CAPCOML5, CAPCOMH5, TCMMODE5, PWMF0, PMWF1
- SPI interface registers: SPICLKD, SPISTAT, SPITDR, SPIRDR, SPICON0, SPICON1
- I^2C interface registers: S1SETUP, S1CON, S1STA, S1DAT, S1ADR
- Analog-to-digital converter registers: ACON, ADCPS, ADAT0, ADAT1
- IrDA interface register: IRDACON

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

9.7 Indexed addressing

This mode is used for the MOVC instruction which allows the 8032 to read a constant from program memory (not data memory). MOVC is often used to read look-up tables that are embedded in program memory. The final address produced by this mode is the result of adding either the 16-bit PC or DPTR value to the contents of the accumulator. The value in the accumulator is referred to as an index. The data fetched from the final location in program memory is stored into the accumulator, overwriting the index value that was previously stored there. For example:

```
MOVC A, @A+DPTR               ; Move code byte relative to
                               ; DPTR into accumulator
MOVC A, @A+PC                 ; Move code byte relative to PC
                               ; into accumulator
```

9.8 Relative addressing

This mode will add the two's-compliment number stored in the second byte of the instruction to the program counter for short jumps within +128 or –127 addresses relative to the program counter. This is commonly used for looping and is very efficient since no additional bus cycle is needed to fetch the jump destination address. For example:

```
SJMP 34h                      ; Jump 34h bytes ahead (in program
                               ; memory) of the address at which
                               ; the SJMP instruction is stored. If
                               ; SJMP is at 1000h, program
                               ; execution jumps to 1034h.
```

9.9 Absolute addressing

This mode will append the 5 high-order bits of the address of the next instruction to the 11 low-order bits of an ACALL or AJUMP instruction to produce a 16-bit jump address. The jump will be within the same 2 Kbyte page of program memory as the first byte of the following instruction. For example:

```
AJMP 0500h                    ; If next instruction is located at
                               ; address 4000h, the resulting jump
                               ; will be made to 4500h.
```

Table 8. Data transfer instruction set (continued)

Mnemonic[1] and use		Description	Length/cycles
MOV	@Ri, direct	Move direct byte to indirect SRAM	2 byte/2 cycle
MOV	@Ri, #data	Move immediate data to indirect SRAM	2 byte/1 cycle
MOV	DPTR, #data16	Load Data Pointer with 16-bit constant	3 byte/2 cycle
MOVC	A, @A+DPTR	Move code byte relative to DPTR to ACC	1 byte/2 cycle
MOVC	A, @A+PC	Move code byte relative to PC to ACC	1 byte/2 cycle
MOVX	A, @Ri	Move XDATA (8-bit addr) to ACC	1 byte/2 cycle
MOVX	A, @DPTR	Move XDATA (16-bit addr) to ACC	1 byte/2 cycle
MOVX	@Ri, A	Move ACC to XDATA (8-bit addr)	1 byte/2 cycle
MOVX	@DPTR, A	Move ACC to XDATA (16-bit addr)	1 byte/2 cycle
PUSH	direct	Push direct byte onto stack	2 byte/2 cycle
POP	direct	Pop direct byte from stack	2 byte/2 cycle
XCH	A, Rn	Exchange register with ACC	1 byte/1 cycle
XCH	A, direct	Exchange direct byte with ACC	2 byte/1 cycle
XCH	A, @Ri	Exchange indirect SRAM with ACC	1 byte/1 cycle
XCHD	A, @Ri	Exchange low-order digit indirect SRAM with ACC	1 byte/1 cycle

1. All mnemonics copyrighted ©Intel Corporation 1980.

Table 9. Boolean variable manipulation instruction set

Mnemonic[1] and use		Description	Length/cycles
CLR	C	Clear carry	1 byte/1 cycle
CLR	bit	Clear direct bit	2 byte/1 cycle
SETB	C	Set carry	1 byte/1 cycle
SETB	bit	Set direct bit	2 byte/1 cycle
CPL	C	Compliment carry	1 byte/1 cycle
CPL	bit	Compliment direct bit	2 byte/1 cycle

Table 43. P1SFS0 and P1SFS1 details (continued)

Port 1 pin	R/W	Default port function		Alternate 1 port function
		P1SFS0[i] = 0, P1SFS1[i] = x		P1SFS0[i] = 1, P1SFS1[i] = 0
		Port 1 Pin, i = 0.. 7		Port 1 Pin, i = 0.. 7
3	R,W	GPIO	UART1 Transmit, TXD1	ADC Chn 3 Input, ADC3
4	R,W	GPIO	SPI Clock, SPICLK	ADC Chn 4 Input, ADC4
5	R,W	GPIO	SPI Receive, SPIRXD	ADC Chn 5 Input, ADC5
6	R,W	GPIO	SPI Transmit, SPITXD	ADC Chn 6 Input, ADC6
7	R,W	GPIO	SPI Select, SPISEL_	ADC Chn 7 Input, ADC7

Table 44. P4SFS0: Port 4 Special Function Select 0 register (SFR 92h, reset value 00h)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
P4SF07	P4SF06	P4SF05	P4SF04	P4SF03	P4SF02	P4SF01	P4SF00

Table 45. P4SFS1: Port 4 Special Function Select 1 register (SFR 93h, reset value 00h)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
P4SF17	P4SF16	P4SF15	P4SF14	P4SF13	P4SF12	P4SF11	P4SF10

Table 46. P4SFS0 and P4SFS1 details

Port 4 pin	R/W	Default port function	Alternate 1 port function	Alternate 2 port function
		P4SFS0[i] = 0, P4SFS1[i] = x	P4SFS0[i] = 1, P4SFS1[i] = 0	P4SFS0[i] = 1, P4SFS1[i] = 1
		Port 4 Pin, i = 0.. 7	Port 4 Pin, i = 0.. 7	Port 4 Pin, i = 0.. 7
0	R,W	GPIO	PCA0 module 0, TCM0	Timer 2 Count Input, T2
1	R,W	GPIO	PCA0 module 1, TCM1	Timer 2 Trigger Input, TX2
2	R,W	GPIO	PCA0 module 2, TCM2	UART1 Receive, RXD1
3	R,W	GPIO	PCA0 Ext Clock, PCACLK0	UART1 Transmit, TXD1
4	R,W	GPIO	PCA1 module 3, TCM3	SPI Clock, SPICLK
5	R,W	GPIO	PCA1 module 4, TCM4	SPI Receive, SPIRXD
6	R,W	GPIO	PCA1 module 5, TCM5	SPI Transmit, SPITXD
7	R,W	GPIO	PCA1 Ext Clock, PCACLK1	SPI Select, SPISEL_

18 MCU bus interface

The MCU module has a programmable bus interface. It is based on a standard 8032 bus, with eight data signals multiplexed with eight low-order address signals (AD[7:0]). It also has eight high-order non-multiplexed address signals (A[15:8]). Time multiplexing is controlled by the address latch signal, ALE.

This bus connects the MCU module to the PSD module, and also connects to external pins only on 80-pin devices. See the *Section 28: AC/DC parameters on page 242* at the end of this document for external bus timing on 80-pin devices.

Four types of data transfers are supported, each transfer is to/from a memory location external to the MCU module:

- Code Fetch cycle using the $\overline{\text{PSEN}}$ signal: fetch a code byte for execution
- Code Read cycle using $\overline{\text{PSEN}}$: read a code byte using the MOVC (Move Constant) instruction
- XDATA Read cycle using the $\overline{\text{RD}}$ signal: read a data byte using the MOVX (Move eXternal) instruction
- XDATA Write cycle using the $\overline{\text{WR}}$ signal: write a data byte using the MOVX instruction

The number of MCU_CLK periods for these transfer types can be specified at runtime by firmware writing to the SFR register named BUSCON (*Table 47 on page 87*). Here, the number of MCU_CLK clock pulses per bus cycle are specified to maximize performance.

Note: ***Important:*** *By default, the BUSCON register is loaded with long bus cycle times (6 MCU_CLK periods) after a reset condition. It is important that the post-reset initialization firmware sets the bus cycle times appropriately to get the most performance, according to Table 49 on page 88. Keep in mind that the PSD module has a faster Turbo mode (default) and a slower but less power consuming Non-Turbo mode. The bus cycle times must be programmed in BUSCON to optimize for each mode as shown in Table 49 on page 88. See Section 27.4.55: PLD non-turbo mode on page 230 for more details.*

18.1 Bus read cycles ($\overline{\text{PSEN}}$ or $\overline{\text{RD}}$)

When the $\overline{\text{PSEN}}$ signal is used to fetch a byte of code, the byte is read from the PSD module or external device and it enters the MCU Pre-Fetch Queue (PFQ). When $\overline{\text{PSEN}}$ is used during a MOVC instruction, or when the $\overline{\text{RD}}$ signal is used to read a byte of data, the byte is routed directly to the MCU, bypassing the PFQ.

Bits in the BUSCON register determine the number of MCU_CLK periods per bus cycle for each of these kinds of transfers to all address ranges.

It is not possible to specify in the BUSCON register a different number of MCU_CLK periods for various address ranges. For example, the user cannot specify 4 MCU_CLK periods for $\overline{\text{RD}}$ read cycles to one address range on the PSD module, and 5 MCU_CLK periods for $\overline{\text{RD}}$ read cycles to a different address range on an external device. However, the user can specify one number of clock periods for $\overline{\text{PSEN}}$ read cycles and a different number of clock periods for $\overline{\text{RD}}$ read cycles.

In this example,

t_{MACH_CYC} = 100ns (4 MCU_CLK periods x 25ns)

$N_{OVERFLOW} = 2^{24}$ = 16777216 up-counts

WDT_{PERIOD} = 100ns X 16777216 = 1.67 seconds

The actual value will be slightly longer due to PFQ/BC.

19.5.1 Firmware example

The following 8051 assembly code illustrates how to operate the WDT. A simple statement in the reset initialization firmware enables the WDT, and then a periodic write to clear the WDT in the main firmware is required to keep the WDT from overflowing. This firmware is based on the example above (40 MHz f_{OSC}, CCON0 = 10h, BUSCON = C1h).

For example, in the reset initialization firmware (the function that executes after a jump to the reset vector):

```
MOV AE, #AA                    ; enable WDT by writing value to
                                  ; WDKEY other than 55h
```

Somewhere in the flow of the main program, this statement will execute periodically to reset the WDT before it's timeout period of 1.67 seconds. For example:

```
MOV A6, #00                    ; reset WDT, loading 000000h.
                                  ; Counting will automatically
                               ; resume as long as 55h in not in
                                           ; WDKEY
```

Table 50. WDKEY: Watchdog Timer Key register (SFR AEh, reset value 55h)

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

Table 51. WDKEY register bit definition

Bit	Symbol	R/W	Definition
[7:0]	WDKEY	W	55h disables the WDT from counting. 55h is automatically loaded in this SFR after any reset condition, leaving the WDT disabled by default. Any value other than 55h written to this SFR will enable the WDT, and counting begins.

Table 52. WDRST: Watchdog Timer Reset Counter register (SFR A6h, reset value 00h)

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

Table 73. Selection of the SCL frequency in Master mode based on f_{OSC} examples

CR2	CR1	CR0	f_{OSC} divided by:	Bit rate (kHz) @ f_{OSC}			
				12 MHz f_{OSC}	24 MHz f_{OSC}	36 MHz f_{OSC}	40 MHz f_{OSC}
0	0	0	32	375	750	X(1)	X(1)
0	0	1	48	250	500	750	833
0	1	0	60	200	400	600	666
0	1	1	120	100	200	300	333
1	0	0	240	50	100	150	166
1	0	1	480	25	50	75	83
1	1	0	960	12.5	25	37.5	41
1	1	1	1920	6.25	12.5	18.75	20

1. These values are beyond the bit rate supported by UPSD33xx.

23.9 I²C Interface Status register (S1STA)

The S1STA register provides status regarding immediate activity and the current state of operation on the I²C bus. All bits in this register are read-only except bit 5, INTR, which is the interrupt flag.

23.9.1 Interrupt conditions

If the I²C interrupt is enabled (EI²C = 1 in SFR named IEA, and EA =1 in SFR named IE), and the SIOE is initialized, then an interrupt is automatically generated when any one of the following five events occur:

- When the SIOE receives an address that matches the contents of the SFR, S1ADR. Requirements: SIOE is in Slave mode, and bit AA = 1 in the SFR S1CON.
- When the SIOE receives General Call address. Requirements: SIOE is in Slave mode, bit AA = 1 in the SFR S1CON
- When a complete data byte has been received or transmitted by the SIOE while in Master mode. The interrupt will occur even if the Master looses arbitration.
- When a complete data byte has been received or transmitted by the SIOE while in selected Slave mode.
- A STOP condition on the bus has been recognized by the SIOE while in selected Slave mode.

Selected Slave mode means the device address sent by the Master device at the beginning of the current data transfer matched the address stored in the S1ADR register.

If the I²C interrupt is not enabled, the MCU may poll the INTR flag in S1STA.

24 Synchronous peripheral interface (SPI)

UPSD33xx devices support one serial SPI interface in Master mode only. This is a three- or four-wire synchronous communication channel, capable of full-duplex operation on 8-bit serial data transfers. The four SPI bus signals are:

- SPIRxD

  Pin P1.5 or P4.5 receives data from the Slave SPI device to the UPSD33xx
- SPITxD

  Pin P1.6 or P4.6 transmits data from the UPSD33xx to the Slave SPI device
- SPICLK

  Pin P1.4 or P4.4 clock is generated from the UPSD33xx to the SPI Slave device
- $\overline{\text{SPISEL}}$

  Pin P1.7 or P4.7 selects the signal from the UPSD33xx to an individual Slave SPI device

This SPI interface supports single-Master/multiple-Slave connections. Multiple-Master connections are not directly supported by the UPSD33xx (no internal logic for collision detection).

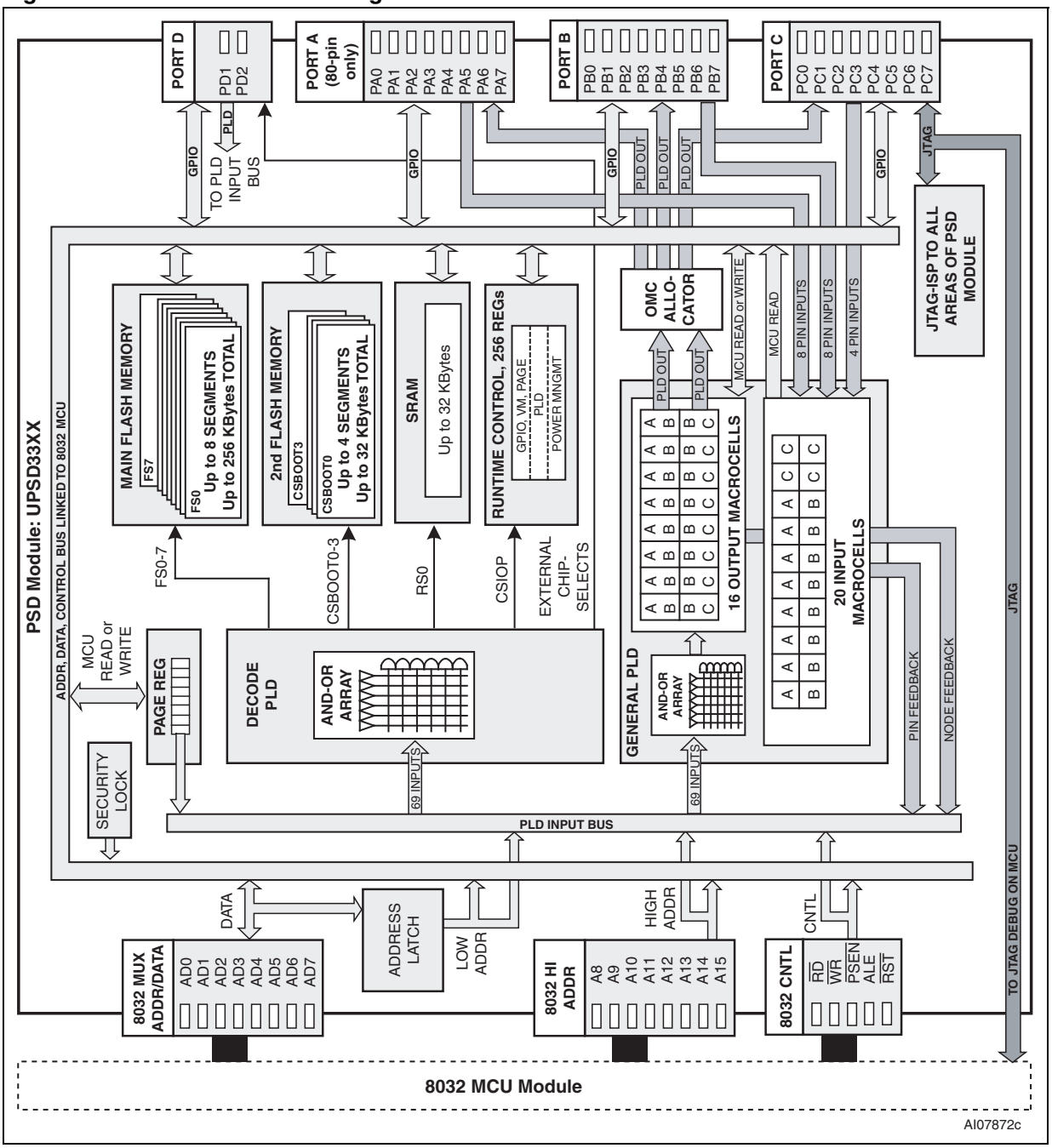
If more than one Slave device is required, the $\overline{\text{SPISEL}}$ signal may be generated from UPSD33xx GPIO outputs (one for each Slave) or from the PLD outputs of the PSD module. *Figure 40* illustrates three examples of SPI device connections using the UPSD33xx:

- Single-Master/Single-Slave with $\overline{\text{SPISEL}}$
- Single-Master/Single-Slave without $\overline{\text{SPISEL}}$
- Single-Master/Multiple-Slave without $\overline{\text{SPISEL}}$

27 PSD module

The PSD module is stacked with the MCU module to form the UPSD33xx, see *Section 3: UPSD33xx hardware description on page 26*. Details of the PSD module are shown in *Figure 50*. The two separate modules interface with each other at the 8032 Address, Data, and Control interface blocks in *Figure 50*.

Figure 50. PSD module block diagram



27.1.14 I/O ports

For 80-pin UPSD33xx devices, the PSD module has 22 individually configurable I/O pins distributed over four ports (these I/O are in addition to I/O on MCU module). For 52-pin UPSD33xx devices, the PSD module has 13 individually configurable I/O pins distributed over three ports. See *Figure 73 on page 219* for I/O port pin availability on these two packages.

I/O port pins on the PSD module (Ports A, B, C, and D) are completely separate from the port pins on the MCU module (Ports 1, 3, and 4). They even have different electrical characteristics. I/O port pins on the PSD module are accessed by csiop registers, or they are controlled by PLD equations. Conversely, I/O Port pins on the MCU module are controlled by the 8032 SFR registers.

Table 113. General I/O pins on PSD module

Pkg	Port A	Port B	Port D	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 68 on page 208*). 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 bi-directionally 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.

27.4.28 Output macrocell

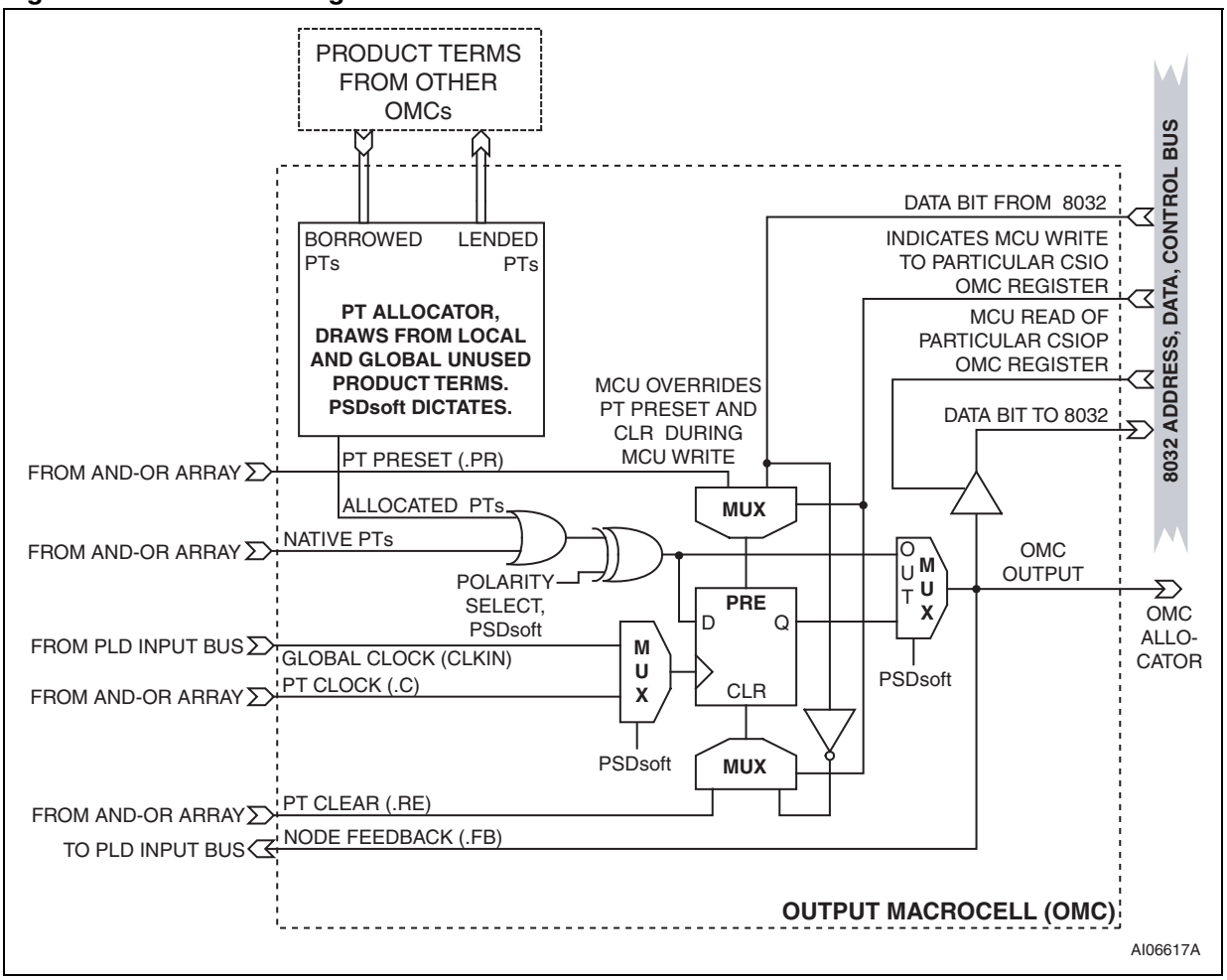
The GPLD has 16 OMCs. Architecture of one individual OMC is shown in *Figure 65*. OMCs can be used for internal node feedback (buried registers to build shift registers, etc.), or their outputs may be routed to external port pins. The user can choose any mixture of OMCs used for buried functions and OMCs used to drive port pins.

Referring to *Figure 65*, for each OMC there are native product terms available from the AND-OR Array to form logic, and also borrowed product terms are available (if unused) from other OMCs. The polarity of the final product term output is controlled by the XOR gate. Each OMC can implement sequential logic using the flip-flop element, or combinatorial logic when bypassing the flip-flop as selected by the output multiplexer. An OMC output can drive a port pin through the OMC Allocator, it can also drive the 8032 data bus, and also it can drive a feedback path to the AND-OR Array inputs, all at the same time.

The flip-flop in each OMC can be synthesized as a D, T, JK, or SR type in PSDsoft Express. OMC flip-flops are specified using PSDsoft Express in the “User Defined Nodes” section of the Design Assistant. Each flip-flop’s clock, preset, and clear inputs may be driven individually from a product term of the AND-OR Array, defined by equations in PSDsoft Express for signals *. c, *.pr, and *.re respectively. The preset and clear inputs on the flip-flops are level activated, active-high logic signals. The clock inputs on the flip-flops are rising-edge logic signals.

Optionally, the signal CLKIN (pin PD1) can be used for a common clock source to all OMC flip-flops. Each flip-flop is clocked on the rising edge. A common clock is specified in PSDsoft Express by assigning the function “Common Clock Input” for pin PD1 in the Pin Definition section, and then choosing the signal CLKIN when specifying the clock input (*.c) for individual flip-flops in the “User Defined Nodes” section.

Figure 65. Detail of a Single OMC



27.4.29 OMC allocator

Outputs of the 16 OMCs can be routed to a combination of pins on Port A (80-pin devices only), Port B, or Port C as shown in *Figure 66*. OMCs are routed to port pins automatically after specifying pin numbers in PSDsoft Express. Routing can occur on a bit-by-bit basis, spitting OMC assignment between the ports. However, one OMC can be routed to one only port pin, not both ports.

27.4.30 Product term allocator

Each OMC has a Product Term Allocator as shown in *Figure 65 on page 201*. PSDsoft Express uses PT Allocators to give and take product terms to and from other OMCs to fit a logic design into the available silicon resources. This happens automatically in PSDsoft Express, but understanding how PT allocation works will help the user if the logic design does not “fit,” in which case the user may try selecting a different pin or different OMC for the

Table 145. Latched Address output, Port B Control register (address = csiop+offset 03h)(1)(2)

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

27.4.40 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 68 on page 208* 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 72 on page 215* 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.” 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 72*). Then in the 8032 initialization firmware, a logic ’1’ is written to the csiop VM register, Bit 7 (PIO_EN) as shown in *Table 109 on page 162*. 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 72. Peripheral I/O mode

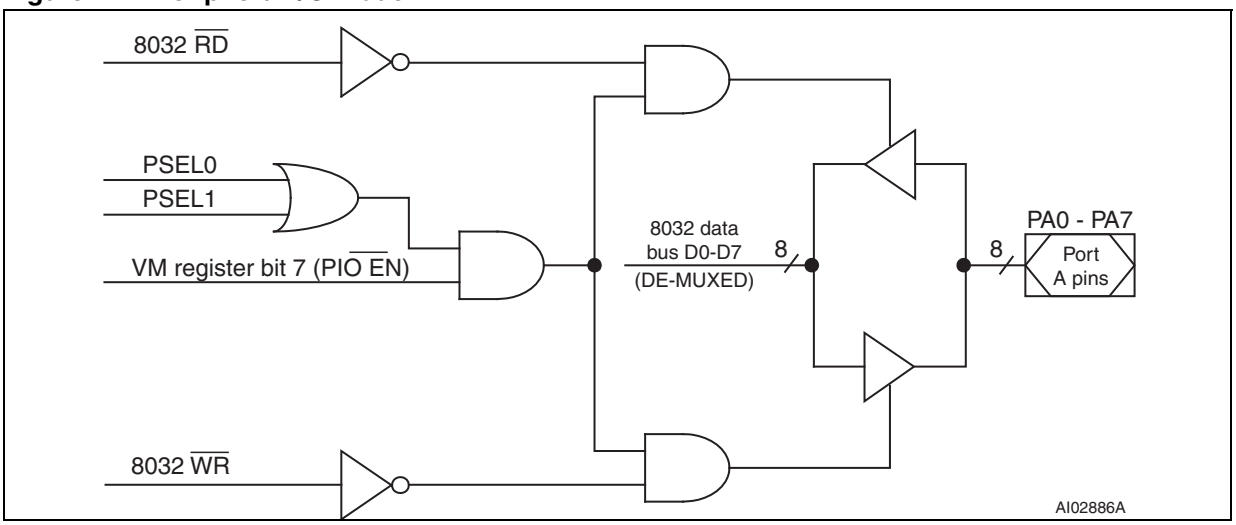


Figure 85. PLD I_{CC} /frequency consumption (3 V range)

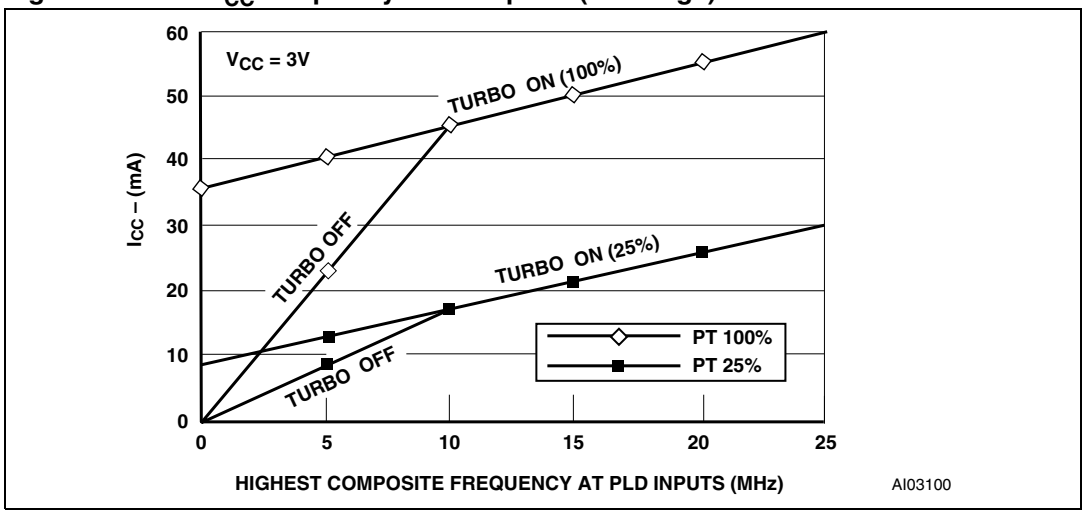


Table 159. PSD module example, typ. power calculation at V_{CC} = 5.0 V (Turbo mode Off)

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 165. Major parameters

Parameter	Test conditions/comments	5.0 V value	3.3 V value	Unit
Operating voltage	–	4.5 to 5.5 (PSD); 3.0 to 3.6 (MCU)	3.0 to 3.6 (PSD and MCU)	V
Operating temperature	–	–40 to 85	–40 to 85	°C
MCU frequency	8 MHz (min) for I^2C	1 Min, 40 Max	1 Min, 40 Max	MHz
Typical Active current (20% of PLD used; 25 °C operation)	40 MHz crystal, Turbo	50	40	mA
	40 MHz crystal, Non-Turbo	48	38	mA
	8 MHz crystal, Turbo	21	18	mA
	8 MHz crystal, Non-Turbo	10	8	mA
Typical Idle current (20% of PLD used; 25 °C operation)	40 MHz crystal divided by 2048 internally. All interfaces are disabled.	16	11	mA
Typical Standby current	Power-down mode needs reset to exit.	140	120	µA
I/O sink/source current, Ports A, B, C, and D	V_{OL} = 0.45 V (max); V_{OH} = 2.4 V (min)	I_{OL} = 8 (max); I_{OH} = –2 (min)	I_{OL} = 4 (max); I_{OH} = –1 (min)	mA
I/O sink/source current, Port 4	V_{OL} = 0.6 V (max); V_{OH} = 2.4 V (min)	I_{OL} = 10 (max); I_{OH} = –10 (min)	I_{OL} = 10 (max); I_{OH} = –10 (min)	mA
PLD macrocells	For registered or combinatorial logic	16	16	–
PLD inputs	Inputs from pins, feedback, or MCU addresses	69	69	–
PLD Outputs	Output to pins or internal feedback	18	18	–
Typical PLD propagation delay, Turbo mode	PLD input to output	15	22	ns

Table 166. Preliminary MCU module DC characteristics

Symbol	Parameter	Test conditions	Min.	Typ.	Max.	Unit
V_{CC}, AV_{CC}	Supply voltage(1)		3.0		3.6	V
V_{IH}	High level input voltage (Ports 1, 3, 4, MCUAD0-7, MCUA8-11, XTAL1, RESET) 5 V tolerant - max voltage 5.5 V	3.0 V < V_{CC} < 3.6 V	0.7V_{CC}		5.5(2)	V
V_{IL}	Low level input voltage (Ports 1, 3, 4, MCUAD0-7, MCUA8-11, XTAL1, RESET)	3.0 V < V_{CC} < 3.6 V	V_{SS} – 0.5		0.3V_{CC}	V
V_{OL1}	Output low voltage (Port 4)	I_{OL} = 10 mA			0.6	V
V_{OL2}	Output low voltage (Other Ports)	I_{OL} =5 mA			0.6	V
V_{OH1}	Output high voltage (Ports 4 push-pull)	I_{OH} = –10 mA	2.4			V
V_{OH2}	Output high voltage (Port 0 push-pull)	I_{OH} = –5 mA	2.4			V
V_{OH3}	Output high voltage (Other Ports Bi-directional mode)	I_{OH} = –20 µA	2.4			V
V_{OP}	XTAL open bias voltage (XTAL1, XTAL2)	I_{OL} = 3.2 mA	1.0		2.0	V
I_{RST}	RESET pin pull-up current (RESET)	V_{IN} = VSS	–10		–55	µA
I_{FR}	XTAL feedback resistor current (XTAL1)	XTAL1 = V_{CC}; XTAL2 = V_{SS}	–20		50	µA
I_{IHL1}	Input high leakage current (Port 0)	V_{SS} < V_{IN} < 5.5 V	–10		10	µA
I_{IHL2}	Input high leakage current (Port 1, 2, 3, 4)	V_{IH} = 2.3 V	–10		10	µA
I_{ILL}	Input low leakage current (Port 1, 2, 3, 4)	V_{IL} < 0.5 V	–10		10	µA
I_{PD}(3)	Power-down mode	V_{CC} = 3.6 V		65	95	µA
$I_{CC\text{-}CPU}$ (4)(5)(6)	Active - 12 MHz	V_{CC} = 3.6 V		14	20	mA
	Idle - 12 MHz			10	12	mA
	Active - 24 MHz	V_{CC} = 3.6 V		19	30	mA
	Idle - 24 MHz			13	17	mA
	Active - 40 MHz	V_{CC} = 3.6 V		26	40	mA
	Idle - 40 MHz			17	22	mA

1. Power supply (V_{CC} , AV_{CC}) is always 3.0 to 3.6V for the MCU module. V_{DD} for the PSD module may be 3V or 5 V.
2. Port 1 is not 5 V tolerant; maximum V_{IH} = V_{CC} + 0.5
3. I_{PD} (Power-down mode) is measured with: XTAL1 = V_{SS}; XTAL2 = NC; RESET = V_{CC}; Port 0 = V_{CC}; all other pins are disconnected.
4. $I_{CC\text{-}CPU}$ (Active mode) is measured with: XTAL1 driven with t_{CLCH}, t_{CHCL} = 5ns, V_{IL} = V_{SS} + 0.5 V, V_{IH} = V_{CC} – 0.5 V, XTAL2 = NC; RESET = V_{SS}; Port 0 = V_{CC}; all other pins are disconnected. I_{CC} would be slightly higher if a crystal oscillator is used (approximately 1 mA).
5. $I_{CC\text{-}CPU}$ (Idle mode) is measured with: XTAL1 driven with t_{CLCH}, t_{CHCL} = 5ns, V_{IL} = V_{SS} + 0.5 V, V_{IH} = V_{CC} – 0.5 V, XTAL2 = NC; RESET = V_{CC}; Port 0 = V_{CC}; all other pins are disconnected. I_{CC} would be slightly higher if a crystal oscillator is used (approximately 1 mA). All IP clocks are disabled.
6. I/O current = 0 mA, all I/O pins are disconnected.

Figure 94. Peripheral I/O READ timing

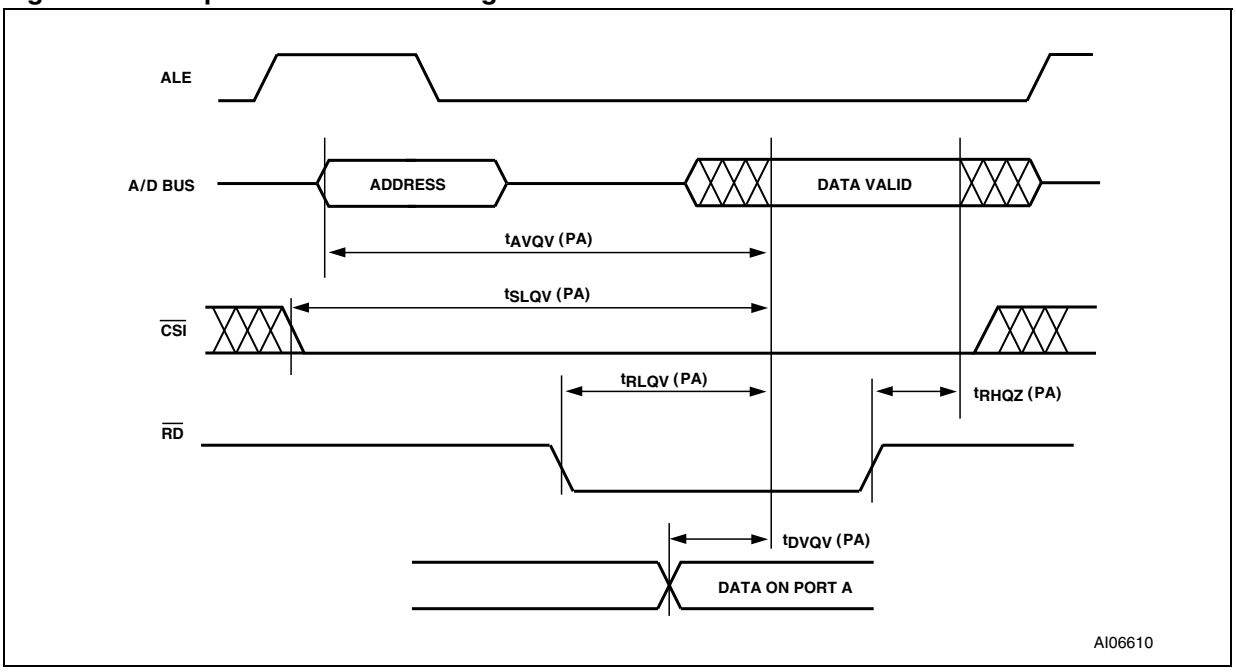


Table 183. Port A peripheral data mode READ timing (5 V PSD module)

Symbol	Parameter	Conditions	Min	Max	Turbo off	Unit
$t_{AVQV-PA}$	Address valid to data valid	(1)		37	+ 10	ns
$t_{SLQV-PA}$	$\overline{CSI}$ valid to data valid			27	+ 10	ns
$t_{RLQV-PA}$	$\overline{RD}$ to data valid	(2)		32		ns
$t_{DVQV-PA}$	Data In to data out valid			22		ns
$t_{RHQZ-PA}$	$\overline{RD}$ to data high-Z			23		ns

1. Any input used to select Port A Data Peripheral mode.
2. Data is already stable on Port A.

Table 184. Port A peripheral data mode READ Timing (3 V PSD module)

Symbol	Parameter	Conditions	Min	Max	Turbo off	Unit
$t_{AVQV-PA}$	Address valid to data valid	(1)		50	+ 15	ns
$t_{SLQV-PA}$	$\overline{CSI}$ valid to data valid			37	+ 15	ns
$t_{RLQV-PA}$	$\overline{RD}$ to data valid	(2)		45		ns
$t_{DVQV-PA}$	Data In to data out valid			38		ns
$t_{RHQZ-PA}$	$\overline{RD}$ to data high-Z			36		ns

1. Any input used to select Port A Data Peripheral mode.
2. Data is already stable on Port A.

Figure 95. Peripheral I/O WRITE timing

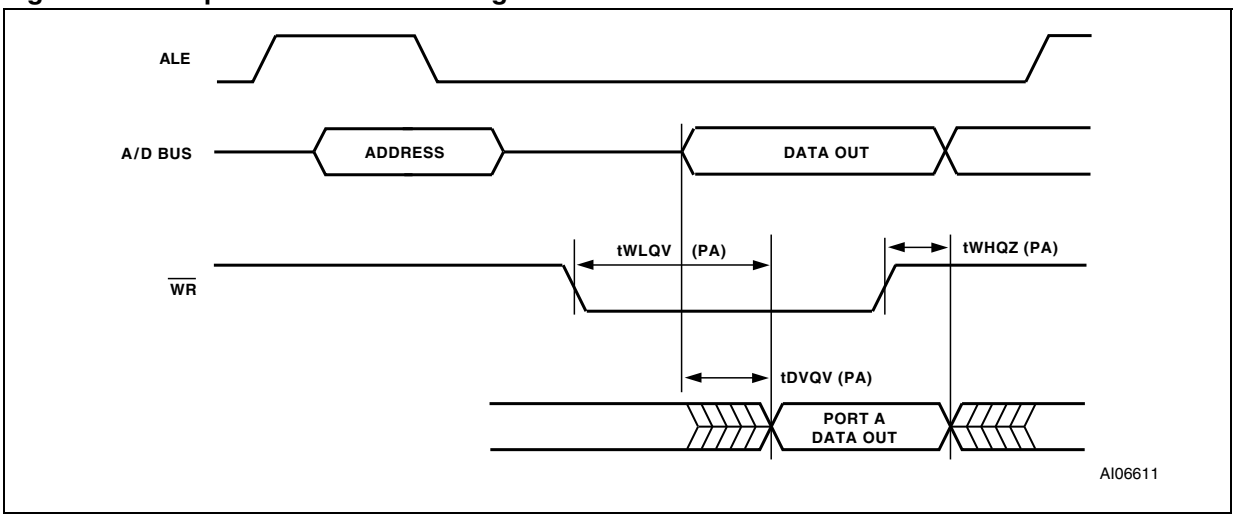


Table 185. Port A peripheral data mode WRITE Timing (5 V PSD module)

Symbol	Parameter	Conditions	Min	Max	Unit
$t_{WLQV-PA}$	$\overline{WR}$ to data propagation delay			25	ns
$t_{DVQV-PA}$	Data to Port A data propagation delay	(1)		22	ns
$t_{WHQZ-PA}$	$\overline{WR}$ Invalid to Port A Tri-state			20	ns

1. Data stable on Port 0 pins to data on Port A.

Table 186. Port A peripheral data mode WRITE Timing (3 V PSD module)

Symbol	Parameter	Conditions	Min	Max	Unit
$t_{WLQV-PA}$	$\overline{WR}$ to data propagation delay			42	ns
$t_{DVQV-PA}$	Data to Port A data propagation delay	(1)		38	ns
$t_{WHQZ-PA}$	$\overline{WR}$ Invalid to Port A Tri-state			33	ns

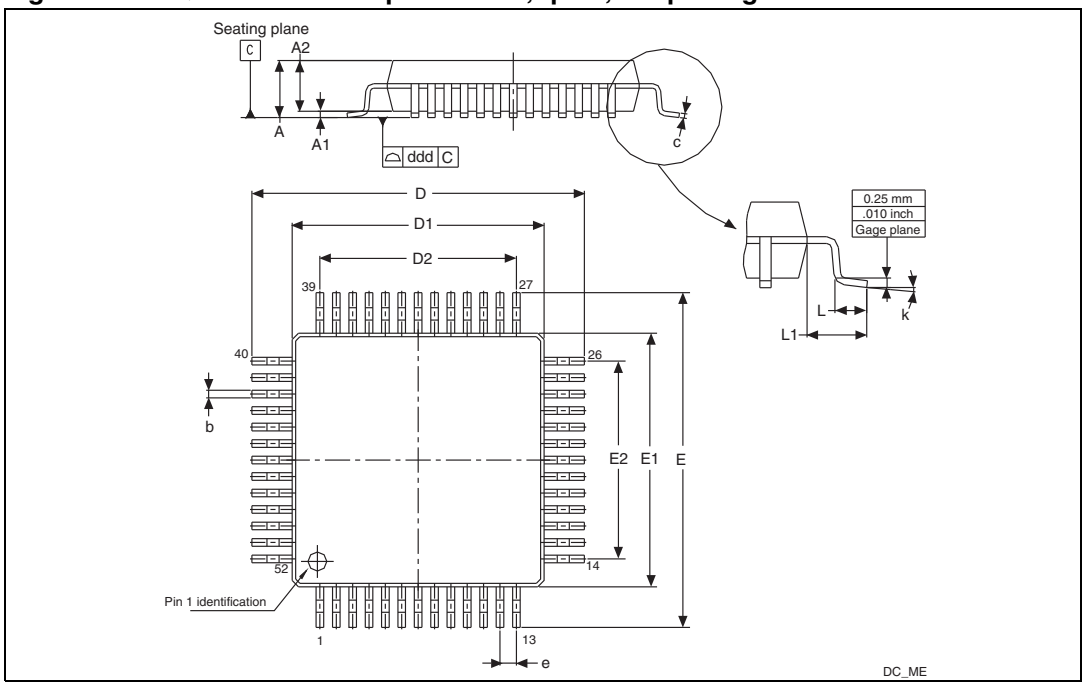
1. Data stable on Port 0 pins to data on Port A.

Table 187. Supervisor Reset and LVD.

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$t_{RST\_LO\_IN}$	Reset input duration		1(1)			µs
t_{RST_ACTV}	Generated Reset duration	f_{OSC} = 40 MHz	10(2)			ms
t_{RST_FIL}	Reset input spike filter			1		µs
V_{RST_HYS}	Reset input hysteresis	V_{CC} = 3.3 V		0.1		V
V_{RST_THRESH}	LVD trip threshold	V_{CC} = 3.3 V	2.4	2.6	2.8	V

1. 25µs minimum to abort a Flash memory program or erase cycle in progress.
2. As F_{OSC} decreases, t_{RST_ACTV} increases. Example: t_{RST_ACTV} = 50ms when F_{OSC} = 8 MHz

Figure 102. LQFP52 – 52-lead plastic thin, quad, flat package outline



1. Drawing not to scale.

Table 191. LQFP52 – 52-lead plastic thin, quad, flat package mechanical data

Symbol	millimeters			inches[1]		
	Typ	Min	Max	Typ	Min	Max
A			1.60			0.063
A1		0.05	0.15		0.002	0.0059
A2		1.35	1.45		0.0531	0.0571
b		0.22	0.38		0.0087	0.015
C		0.09	0.2		0.0035	0.0079
D	12			0.4724		
D1	10			0.3937		
D2	7.8			0.3071		
E	12			0.4724		
E1	10			0.3937		
E2	7.8			0.3071		
e	0.65			0.0256		
L		0.45	0.75		0.0177	0.0295
L1	1			0.0394		
k		0°	7°		0°	7°
ddd	0.100			0.0039		

1. Values in inches are converted from mm and rounded to 4 decimal digits.