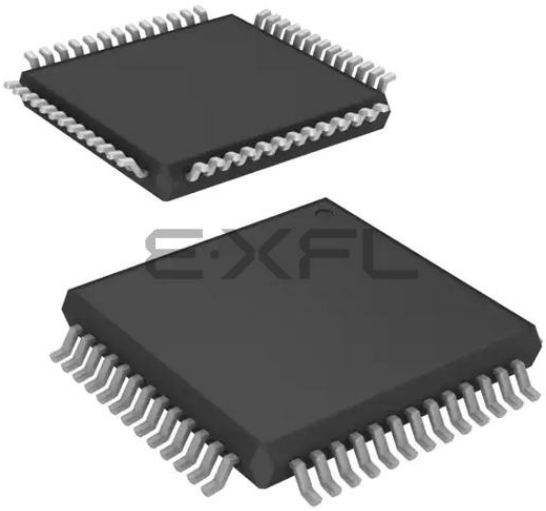


Welcome to E-XFL.COM

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, UART/USART, USB
Peripherals	LVR, POR, PWM, WDT
Number of I/O	37
Program Memory Size	288KB (288K 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 4x8b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	52-LQFP
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/upsd3234a-40t6

2.2.4 Program counter

The Program Counter is a 16-bit wide which consists of two 8-bit registers, PCH and PCL. This counter indicates the address of the next instruction to be executed. In $\overline{\text{RESET}}$ state, the program counter has reset routine address (PCH:00h, PCL:00h).

2.2.5 Program status word

The Program Status Word (PSW) contains several bits that reflect the current state of the CPU and select Internal RAM (00h to 1Fh: Bank0 to Bank3). The PSW is described in *Figure 8*. It contains the Carry flag, the Auxiliary Carry flag, the Half Carry (for BCD operation), the General Purpose flag, the Register Bank Select flags, the Overflow flag, and Parity flag.

[Carry flag, CY]. This flag stores any carry or not borrow from the ALU of CPU after an arithmetic operation and is also changed by the Shift Instruction or Rotate Instruction.

[Auxiliary Carry flag, AC]. After operation, this flag is set when there is a carry from Bit 3 of ALU or there is no borrow from Bit 4 of ALU.

[Register Bank Select flags, RS0, RS1]. These flags select one of four banks (00~07H:bank0, 08~0Fh:bank1, 10~17h:bank2, 17~1Fh:bank3) in Internal RAM.

[Overflow flag, OV]. This flag is set to '1' when an overflow occurs as the result of an arithmetic operation involving signs. An overflow occurs when the result of an addition or subtraction exceeds +127 (7Fh) or -128 (80h). The CLRV instruction clears the overflow flag. There is no set instruction. When the BIT instruction is executed, Bit 6 of memory is copied to this flag.

[Parity flag, P]. This flag reflects the number of Accumulator's 1. If the number of Accumulator's 1 is odd, P=0; otherwise, P=1. The sum when adding Accumulator's 1 to P is always even.

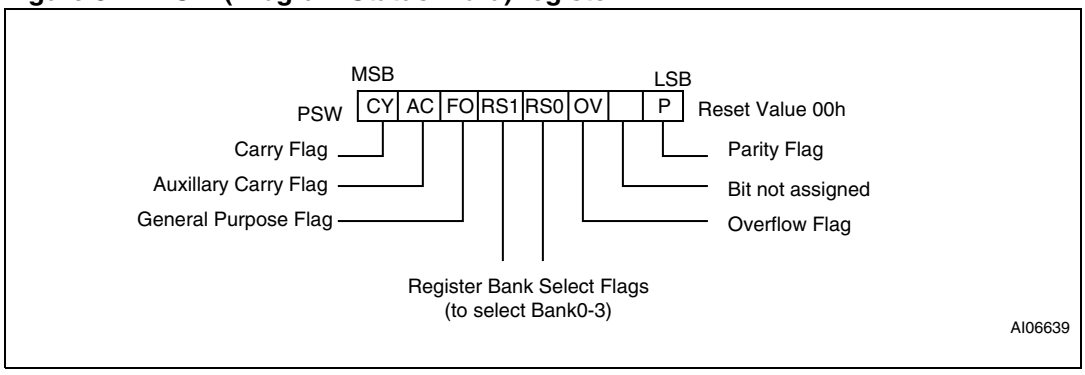
2.2.6 Registers R0~R7

General purpose 8-bit registers that are locked in the lower portion of internal data area.

2.2.7 Data pointer register

Data Pointer Register is 16-bit wide which consists of two-8bit registers, DPH and DPL. This register is used as a data pointer for the data transmission with external data memory in the PSD module.

Figure 8. PSW (Program Status Word) register



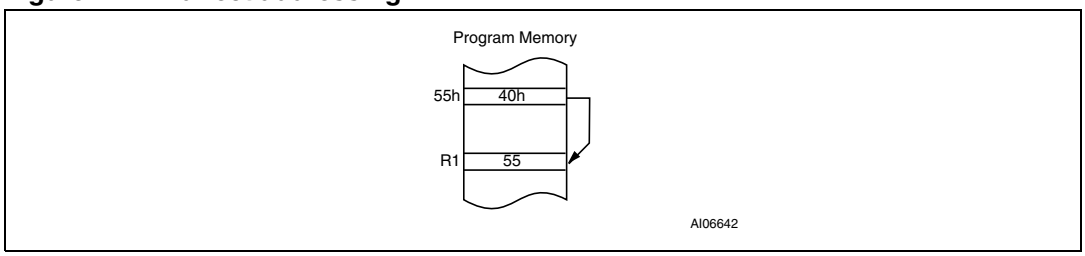
2.9.2 Indirect addressing

In indirect addressing the instruction specifies a register which contains the address of the operand. Both internal and external RAM can be indirectly addressed. The address register for 8-bit addresses can be R0 or R1 of the selected register bank, or the Stack Pointer. The address register for 16-bit addresses can only be the 16-bit “data pointer” register, DPTR.

Example:

```
mov @R1, #40 H ;[R1] <-----40H
```

Figure 11. Indirect addressing



2.9.3 Register addressing

The register banks, containing registers R0 through R7, can be accessed by certain instructions which carry a 3-bit register specification within the opcode of the instruction. Instructions that access the registers this way are code efficient, since this mode eliminates an address byte. When the instruction is executed, one of four banks is selected at execution time by the two bank select bits in the PSW.

Example:

```
mov PSW, #0001000B ; select Bank0
mov A, #30H
mov R1, A
```

2.9.4 Register-specific addressing

Some instructions are specific to a certain register. For example, some instructions always operate on the Accumulator, or Data Pointer, etc., so no address byte is needed to point it. The opcode itself does that.

2.9.5 Immediate constants addressing

The value of a constant can follow the opcode in Program memory.

Example:

```
mov A, #10H.
```

2.9.6 Indexed addressing

Only Program memory can be accessed with indexed addressing, and it can only be read. This addressing mode is intended for reading look-up tables in Program memory. A 16-bit base register (either DPTR or PC) points to the base of the table, and the Accumulator is set up with the table entry number. The address of the table entry in Program memory is formed by adding the Accumulator data to the base pointer.

Table 12. Boolean instructions

Mnemonic	Operation
ANL C,bit	C = A .AND. bit
ANL C,/bit	C = C .AND. .NOT. bit
ORL C,bit	C = A .OR. bit
ORL C,/bit	C = C .OR. .NOT. bit
MOV C,bit	C = bit
MOV bit,C	bit = C
CLR C	C = 0
CLR bit	bit = 0
SETB C	C = 1
SETB bit	bit = 1
CPL C	C = .NOT. C
CPL bit	bit = .NOT. bit
JC rel	Jump if C =1
JNC rel	Jump if C = 0
JB bit,rel	Jump if bit =1
JNB bit,rel	Jump if bit = 0
JBC bit,rel	Jump if bit = 1; CLR bit

2.14 Relative offset

The destination address for these jumps is specified to the assembler by a label or by an actual address in Program memory. However, the destination address assembles to a relative offset byte. This is a signed (two's complement) offset byte which is added to the PC in two's complement arithmetic if the jump is executed.

The range of the jump is therefore -128 to +127 Program Memory bytes relative to the first byte following the instruction.

2.15 Jump instructions

Table 13 shows the list of unconditional jump instructions. The table lists a single "JMP add" instruction, but in fact there are three SJMP, LJMP, and AJMP, which differ in the format of the destination address. JMP is a generic mnemonic which can be used if the programmer does not care which way the jump is en-coded.

The SJMP instruction encodes the destination address as a relative offset, as described above. The instruction is 2 bytes long, consisting of the opcode and the relative offset byte. The jump distance is limited to a range of -128 to +127 bytes relative to the instruction following the SJMP.

If there is no interrupt in progress at the time RETI is executed, then the RETI is functionally identical to RET.

Table 13. Unconditional Jump instructions

Mnemonic	Operation
JMP addr	Jump to addr
JMP @A+DPTR	Jump to A+DPTR
CALL addr	Call Subroutine at addr
RET	Return from subroutine
RETI	Return from interrupt
NOP	No operation

Table 14 shows the list of conditional jumps available to the UPSD323xx device user. All of these jumps specify the destination address by the relative offset method, and so are limited to a jump distance of -128 to +127 bytes from the instruction following the conditional jump instruction. Important to note, however, the user specifies to the assembler the actual destination address the same way as the other jumps: as a label or a 16-bit constant.

There is no Zero Bit in the PSW. The JZ and JNZ instructions test the Accumulator data for that condition.

The DJNZ instruction (Decrement and Jump if Not Zero) is for loop control. To execute a loop N times, load a counter byte with N and terminate the loop with a DJNZ to the beginning of the loop, as shown below for N = 10:

```
MOV COUNTER,#10
LOOP: (begin loop)
  •
  •
  •
(end loop)
DJNZ COUNTER, LOOP
(continue)
```

The CJNE instruction (Compare and Jump if Not Equal) can also be used for loop control as in *Table 9*. Two bytes are specified in the operand field of the instruction. The jump is executed only if the two bytes are not equal. In the example of *Table 9* Shifting a BCD Number One Digits to the Right, the two bytes were data in R1 and the constant 2Ah. The initial data in R1 was 2Eh.

Every time the loop was executed, R1 was decremented, and the looping was to continue until the R1 data reached 2Ah.

Another application of this instruction is in “greater than, less than” comparisons. The two bytes in the operand field are taken as unsigned integers. If the first is less than the second, then the Carry Bit is set (1). If the first is greater than or equal to the second, then the Carry Bit is cleared.

Table 16. List of all SFRs (continued)

SFR Addr	Reg Name	Bit Register Name								Reset Value	Comments
		7	6	5	4	3	2	1	0		
C9	T2MOD								DCEN	00	Timer 2 mode
CA	RCAP2L									00	Timer 2 Reload low
CB	RCAP2H									00	Timer 2 Reload High
CC	TL2									00	Timer 2 Low byte
CD	TH2									00	Timer 2 High byte
D0	PSW	CY	AC	FO	RS1	RS0	OV		P	00	Program Status Word
D1	S1SETUP									00	DDC I^2C (S1) Setup
D2	S2SETUP									00	I^2C (S2) Setup
D4	RAMBUF									XX	DDC Ram Buffer
D5	DDCDAT									00	DDC Data xmit register
D6	DDCADR									00	Addr pointer register
D7	DDCCON	—	EX_DAT	SWENB	DDC_AX	DDCINT	DDC1EN	SWHINT	M0	00	DDC Control Register
D8	S1CON	CR2	ENI1	STA	STO	ADDR	AA	CR1	CR0	00	DDC I^2C Control Reg
D9	S1STA	GC	Stop	Intr	TX-Md	Bbusy	Blost	ACK_R	SLV	00	DDC I^2C Status
DA	S1DAT									00	Data Hold Register
DB	S1ADR									00	DDC I^2C address
DC	S2CON	CR2	EN1	STA	STO	ADDR	AA	CR1	CR0	00	I^2C Bus Control Reg
DD	S2STA	GC	Stop	Intr	TX-Md	Bbusy	Blost	ACK_R	SLV	00	I^2C Bus Status
DE	S2DAT									00	Data Hold Register
DF	S2ADR									00	I^2C address
E0	ACC									00	Accumulator

5.11 How interrupts are handled

The interrupt flags are sampled at S5P2 of every machine cycle. The samples are polled during following machine cycle. If one of the flags was in a set condition at S5P2 of the preceding cycle, the polling cycle will find it and the interrupt system will generate an LCALL to the appropriate service routine, provided this H/W generated LCALL is not blocked by any of the following conditions:

- An interrupt of equal priority or higher priority level is already in progress.
- The current machine cycle is not the final cycle in the execution of the instruction in progress.
- The instruction in progress is RETI or any access to the interrupt priority or interrupt enable registers.

The polling cycle is repeated with each machine cycle, and the values polled are the values that were present at S5P2 of the previous machine cycle.

Note: If an interrupt flag is active but being responded to for one of the above mentioned conditions, if the flag is still inactive when the blocking condition is removed, the denied interrupt will not be serviced. In other words, the fact that the interrupt flag was once active but not serviced is not remembered. Every polling cycle is new.

The processor acknowledges an interrupt request by executing a hardware generated LCALL to the appropriate service routine. The hardware generated LCALL pushes the contents of the Program Counter on to the stack (but it does not save the PSW) and reloads the PC with an address that depends on the source of the interrupt being vectored to as shown in *Table 24*.

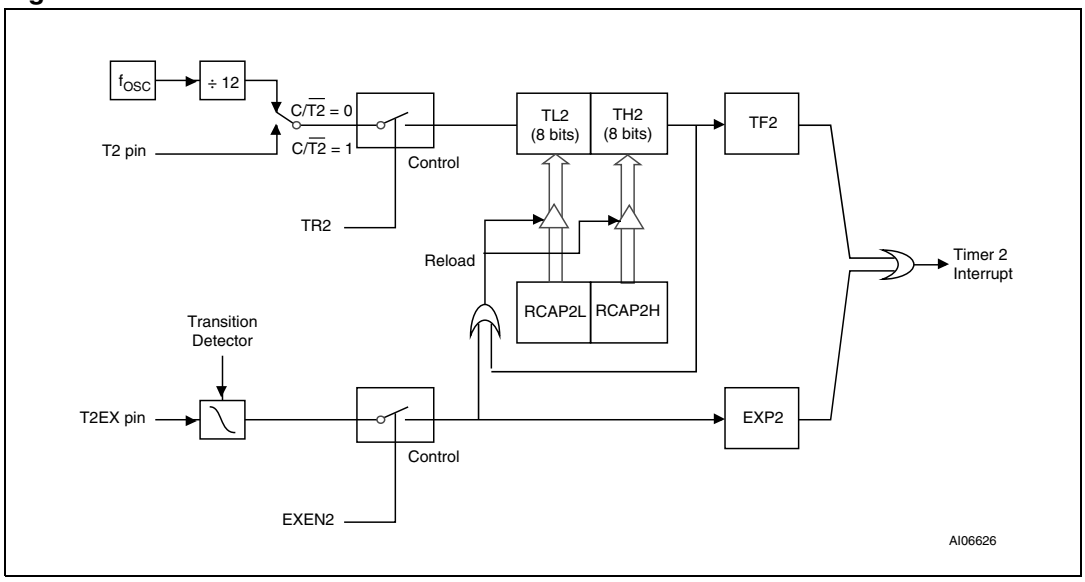
Execution proceeds from that location until the RETI instruction is encountered. The RETI instruction informs the processor that the interrupt routine is no longer in progress, then pops the top two bytes from the stack and reloads the Program Counter. Execution of the interrupted program continues from where it left off.

Note: A simple RET instruction would also return execution to the interrupted program, but it would have left the interrupt control system thinking an interrupt was still in progress, making future interrupts impossible.

Table 24. Vector addresses

Source	Vector address
Int0	0003h
2nd USART	004Bh
Timer 0	000Bh
I²C	0043h
Int1	0013h
DDC	003Bh
Timer 1	001Bh
USB	0033h
1st USART	0023h
Timer 2+EXF2	002Bh

Figure 25. Timer 2 in Auto-Reload mode



13 Analog-to-digital convertor (ADC)

The analog to digital (A/D) converter allows conversion of an analog input to a corresponding 8-bit digital value. The A/D module has four analog inputs, which are multiplexed into one sample and hold. The output of the sample and hold is the input into the converter, which generates the result via successive approximation. The analog supply voltage is connected to AV_{REF} of ladder resistance of A/D module.

The A/D module has two registers which are the control register ACON and A/D result register ADAT. The register ACON, shown in *Table 47*, controls the operation of the A/D converter module. To use analog inputs, I/O is selected by P1SFS register. Also an 8-bit prescaler ASCL divides the main system clock input down to approximately 6MHz clock that is required for the ADC logic. Appropriate values need to be loaded into the prescaler based upon the main MCU clock frequency prior to use.

The processing of conversion starts when the Start bit ADST is set to '1.' After one cycle, it is cleared by hardware. The register ADAT contains the results of the A/D conversion. When conversion is completed, the result is loaded into the ADAT the A/D Conversion Status bit ADSF is set to '1.'

The block diagram of the A/D module is shown in Figure *Figure 34*. The A/D Status bit ADSF is set automatically when A/D conversion is completed, cleared when A/D conversion is in process.

The ASCL should be loaded with a value that results in a clock rate of approximately 6MHz for the ADC using the following formula:

ADC clock input = (f_{OSC} / 2) / (Prescaler register value +1)

Where f_{OSC} is the MCU clock input frequency.

The conversion time for the ADC can be calculated as follows:

ADC Conversion Time = 8 clock * 8bits * (ADC Clock) ~= 10.67usec (at 6MHz)

13.1 ADC interrupt

The ADSF Bit in the ACON register is set to '1' when the A/D conversion is complete. The status bit can be driven by the MCU, or it can be configured to generate a falling edge interrupt when the conversion is complete.

The ADSF Interrupt is enabled by setting the ADSFINT Bit in the PCON register. Once the bit is set, the external INT1 Interrupt is disabled and the ADSF Interrupt takes over as INT1. INT1 must be configured as if it is an edge interrupt input. The INP1 pin (p3.3) is available for general I/O functions, or Timer1 gate control.

16.1.1 DDCDAT register

DDC1 DATA register for transmission (DDCDAT: 0D5h)

- 8-bit READ and WRITE register
- Indicates DATA BYTE to be transmitted in DDC1 protocol

16.1.2 DDCADR register

Address pointer for DDC interface (DDCADR: 0D6h)

- 8-bit READ and WRITE register.
- Address pointer with the capability of the post increment. After each access to RAMBUF register (either by software or by hardware DDC1 interface), the content of this register will be increased by one. It's available both in DDC1, DDC2 (DDC2B, DDC2B+, and DDC2AB) and system operation.

Table 60. DDC SFR memory map

SFR Addr	Reg Name	Bit Register Name								Reset Value	Comments
		7	6	5	4	3	2	1	0		
D4	RAMBUF									XX	DDC Ram Buffer
D5	DDCDAT									00	DDC Data xmit register
D6	DDCADR									00	Addr pointer register
D7	DDCCON	—	EX_DAT	SWENB	DDC_AX	DDCINT	DDC1EN	SWHINT	M0	00	DDC Control Register

Table 61. Description of the DDCON register bits

Bit	Symbol	Function
7	—	Reserved
6	EX_DAT	0 = The SRAM has 128 bytes (Default) 1 = The SRAM has 256 bytes
5	SWENB	**Note:** This bit is valid for DDC1 & DDC2b modes 0 = Data is automatically read from SRAM at the current location of DDCADR and sent out via current DDC protocol. (Default) 1 = MCU is interrupted during the current data byte transmission period to load the next byte of data to send out.
4	DDC_AX	**Note:** This bit is valid for DDC1 & DDC2b modes 0 = Data is automatically read from SRAM at the current location of DDCADR and sent out via current DDC protocol. (Default) 1 = MCU is interrupted during the current data byte transmission period to load the next byte of data to send out. This bit only affects DDC2b mode Operation: 0 = DDC2b I2C Address is A0/A1 (default) 1 = DDC2b I2C Address is AX. Least 3 significant address bits are ignored.

Table 64. Description of the UADR Bits

Bit	Symbol	R/W	Function
7	USBEN	R/W	USB Function Enable Bit. When USBEN is clear, the USB module will not respond to any tokens from host. $\overline{\text{RESET}}$ clears this bit.
6 to 0	UADD6 to UADD0	R/W	Specify the USB address of the device. $\overline{\text{RESET}}$ clears these bits.

Table 65. USB interrupt enable register (UIEN: 0E9h)

7	6	5	4	3	2	1	0
SUSPNDI	RSTE	RSTFIE	TXD0IE	RXD0IE	TXD1IE	EOPIE	RESUMI

Table 66. Description of the UIEN bits

Bit	Symbol	R/W	Function
7	SUSPNDI	R/W	Enable SUSPND Interrupt
6	RSTE	R/W	Enable USB Reset; also resets the CPU and PSD modules when bit is set to '1.'
5	RSTFIE	R/W	Enable RSTF (USB Bus Reset Flag) Interrupt
4	TXD0IE	R/W	Enable TXD0 Interrupt
3	RXD0IE	R/W	Enable RXD0 Interrupt
2	TXD1IE	R/W	Enable TXD1 Interrupt
1	EOPIE	R/W	Enable EOP Interrupt
0	RESUMI	R/W	Enable USB Resume Interrupt when it is the Suspend mode

Table 67. USB interrupt status register (UISTA: 0E8h)

7	6	5	4	3	2	1	0
SUSPND	—	RSTF	TXD0F	RXD0F	TXD1F	EOPF	RESUMF

Table 68. Description of the UISTA bits

Bit	Symbol	R/W	Function
7	SUSPND	R/W	USB Suspend Mode Flag. To save power, this bit should be set if a 3ms constant idle state is detected on USB bus. Setting this bit stops the clock to the USB and causes the USB module to enter Suspend mode. Software must clear this bit after the Resume flag (RESUMF) is set while this Resume Interrupt Flag is serviced
6	—	—	Reserved
5	RSTF	R	USB Reset Flag. This bit is set when a valid $\overline{\text{RESET}}$ signal state is detected on the D+ and D- lines. When the RSTE bit in the UIEN Register is set, this reset detection will also generate an internal reset signal to reset the CPU and other peripherals including the USB module.

21 PSD module detailed operation

As shown in *Figure 14*, the PSD module consists of five major types of functional blocks:

- Memory blocks
- PLD blocks
- I/O Ports
- Power Management Unit (PMU)
- JTAG Interface

The functions of each block are described in the following sections. Many of the blocks perform multiple functions, and are user configurable.

23 PLDs

PLDs bring programmable logic functionality to the UPSD. After specifying the logic for the PLDs using PSDsoft Express, the logic is programmed into the device and available upon Power-up.

Table 90. DPLD and CPLD Inputs

Input Source	Input Name	Number of Signals
MCU Address Bus	A15-A0	16
MCU Control Signals	$\overline{PSEN}$, $\overline{RD}$, $\overline{WR}$, ALE	4
RESET	$\overline{RST}$	1
Power-down	PDN	1
Port A Input Macrocells(1)	PA7-PA0	8
Port B Input Macrocells	PB7-PB0	8
Port C Input Macrocells	PC2-4, PC7	4
Port D Inputs	PD2-PD1	2
Page Register	PGR7-PGR0	8
Macrocell AB Feedback	MCELLAB.FB7-FB0	8
Macrocell BC Feedback	MCELLBC.FB7-FB0	8
Flash memory Program Status bit	Ready/$\overline{Busy}$	1

Note: 1. These inputs are not available in the 52-pin package.

The PSD module contains two PLDs: the Decode PLD (DPLD), and the Complex PLD (CPLD). The PLDs are briefly discussed in the next few paragraphs, and in more detail in *Section 23.2: Decode PLD (DPLD)*, and *Section 23.3: Complex PLD (CPLD)*. *Figure 57* shows the configuration of the PLDs.

The DPLD performs address decoding for Select signals for PSD module components, such as memory, registers, and I/O ports.

The CPLD can be used for logic functions, such as loadable counters and shift registers, state machines, and encoding and decoding logic. These logic functions can be constructed using the Output Macrocells (OMC), Input Macrocells (IMC), and the AND Array. The CPLD can also be used to generate External Chip Select (ECS1-ECS2) signals.

The AND Array is used to form product terms. These product terms are specified using PSDsoft. The PLD input signals consist of internal MCU signals and external inputs from the I/O ports. The input signals are shown in *Table 90*.

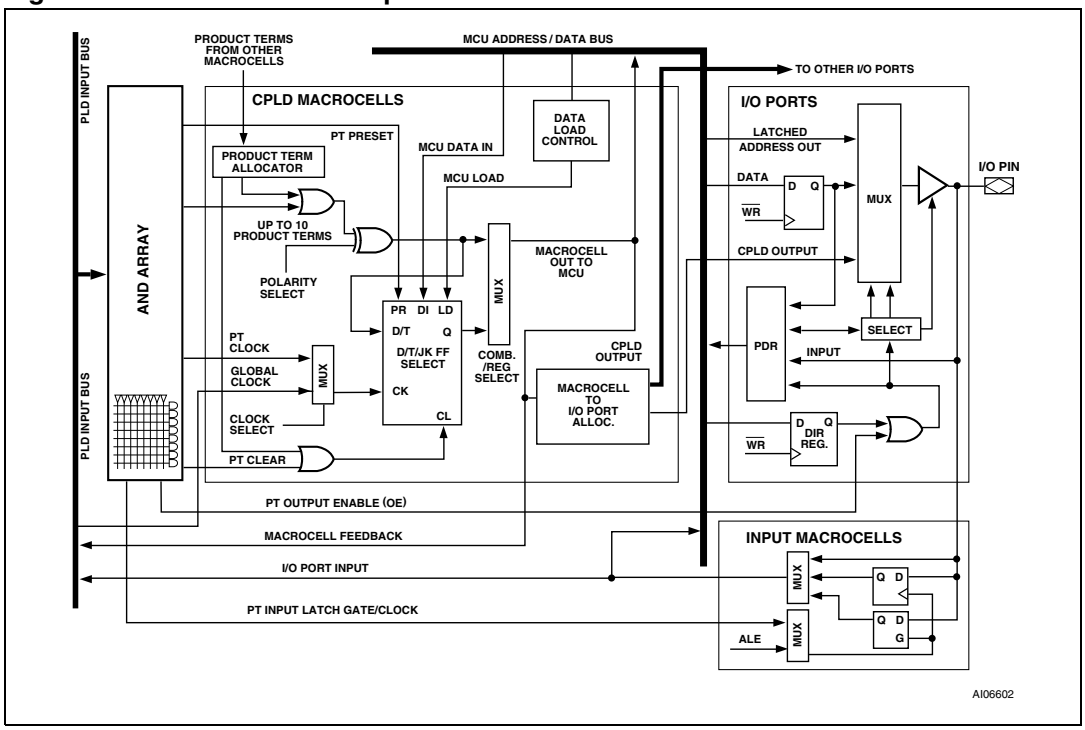
23.1 Turbo bit in PSD module

The PLDs can minimize power consumption by switching off when inputs remain unchanged for an extended time of about 70ns. Resetting the Turbo bit to '0' (Bit 3 of PMMR0) automatically places the PLDs into standby if no inputs are changing. Turning the Turbo mode off increases propagation delays while reducing power consumption.

The Input Macrocells (IMC) and Output Macrocells (OMC) are connected to the PSD module internal data bus and can be directly accessed by the MCU. This enables the MCU software to load data into the Output Macrocells (OMC) or read data from both the Input and Output Macrocells (IMC and OMC).

This feature allows efficient implementation of system logic and eliminates the need to connect the data bus to the AND Array as required in most standard PLD macrocell architectures.

Figure 58. Macrocell and I/O ports



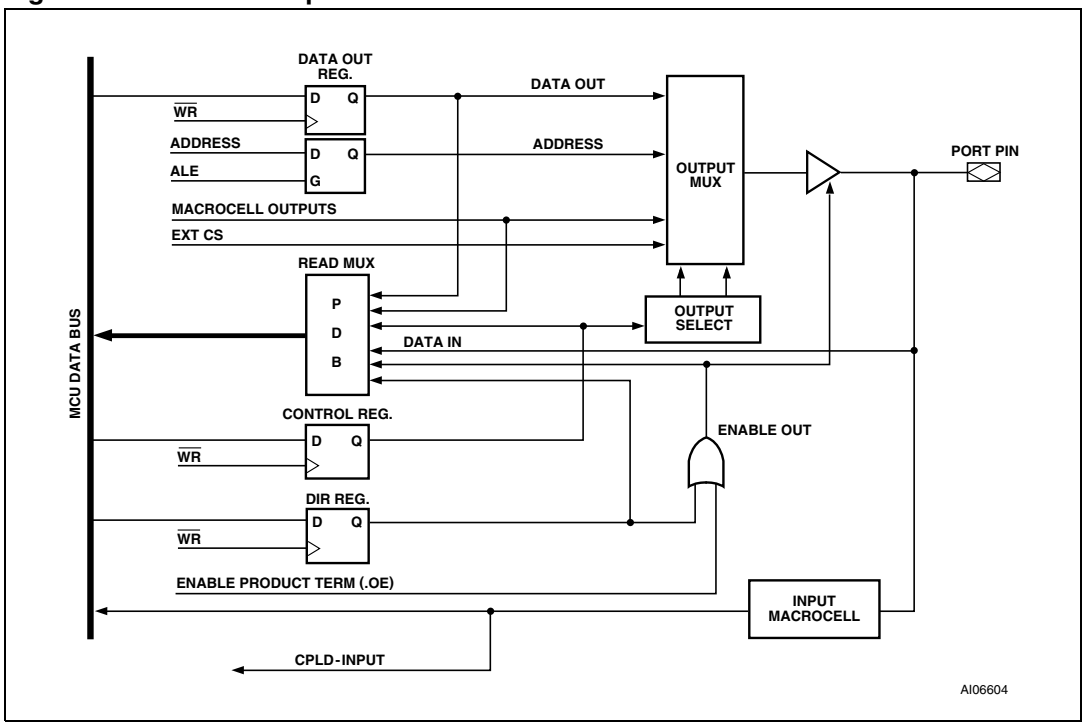
23.4 Output macrocell (OMC)

Eight of the Output Macrocells (OMC) are connected to Ports A and B pins and are named as McellAB0-McellAB7. The other eight macrocells are connected to Ports B and C pins and are named as McellBC0-McellBC7. If an McellAB output is not assigned to a specific pin in PSDsoft, the Macrocell Allocator block assigns it to either Port A or B. The same is true for a McellBC output on Port B or C. *Table 92* shows the macrocells and port assignment.

The Output Macrocell (OMC) architecture is shown in *Figure 59*. As shown in the figure, there are native product terms available from the AND Array, and borrowed product terms available (if unused) from other Output Macrocells (OMC). The polarity of the product term is controlled by the XOR gate. The Output Macrocell (OMC) can implement either sequential logic, using the flip-flop element, or combinatorial logic. The multiplexer selects between the sequential or combinatorial logic outputs. The multiplexer output can drive a port pin and has a feedback path to the AND Array inputs.

The flip-flop in the Output Macrocell (OMC) block can be configured as a D, T, JK, or SR type in PSDsoft. The flip-flop's clock, preset, and clear inputs may be driven from a product term of the AND Array. Alternatively, CLKIN (PD1) can be used for the clock input to the flip-

Figure 61. General I/O port architecture



The Port pin's tri-state output driver enable is controlled by a two input OR gate whose inputs come from the CPLD AND Array enable product term and the Direction Register. If the enable product term of any of the Array outputs are not defined and that port pin is not defined as a CPLD output in the PSDsoft, then the Direction Register has sole control of the buffer that drives the port pin.

The contents of these registers can be altered by the MCU. The Port Data Buffer (PDB) feedback path allows the MCU to check the contents of the registers.

Ports A, B, and C have embedded Input Macrocells (IMC). The Input Macrocells (IMC) can be configured as latches, registers, or direct inputs to the PLDs. The latches and registers are clocked by Address Strobe (ALE) or a product term from the PLD AND Array. The outputs from the Input Macrocells (IMC) drive the PLD input bus and can be read by the MCU. See *Figure 60*.

24.2 Port operating modes

The I/O Ports have several modes of operation. Some modes can be defined using PSDsoft, some by the MCU writing to the Control Registers in CSIOP space, and some by both. The modes that can only be defined using PSDsoft must be programmed into the device and cannot be changed unless the device is reprogrammed. The modes that can be changed by the MCU can be done so dynamically at run-time. The PLD I/O, Data Port, Address Input, and Peripheral I/O modes are the only modes that must be defined before programming the device. All other modes can be changed by the MCU at run-time. See Application Note *AN1171* for more detail.

Table 93 summarizes which modes are available on each port. *Table 96* shows how and where the different modes are configured. Each of the port operating modes are described in the following sections.

Figure 66. Port D external chip select signals

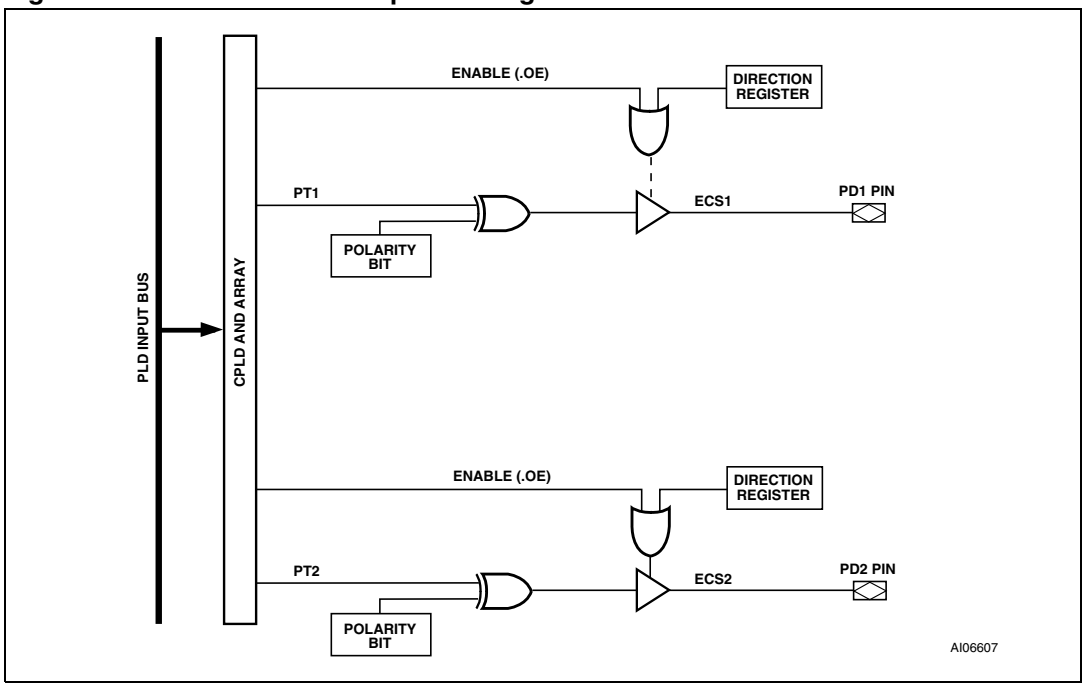


Table 118. DC characteristics (5 V devices)

Symbol	Parameter	Test conditions (in addition to those in *Table 113*)	Min.	Typ.	Max.	Unit
V_{IH}	Input high voltage (Ports 1, 2, 3, 4[Bits 7,6,5,4,3,1,0], XTAL1, $\overline{RESET}$)	4.5 V < V_{CC} < 5.5 V	0.7 V_{CC}		V_{CC} + 0.5	V
V_{IH1}	Input high voltage (Ports A, B, C, D, 4[Bit 2], USB+, USB–)	4.5 V < V_{CC} < 5.5 V	2.0		V_{CC} + 0.5	V
V_{IL}	Input low voltage (Ports 1, 2, 3, 4[Bits 7,6,5,4,3,1,0], XTAL1, $\overline{RESET}$)	4.5 V < V_{CC} < 5.5 V	V_{SS}– 0.5		0.3 V_{CC}	V
V_{IL1}	Input low voltage (Ports A, B, C, D, 4[Bit 2])	4.5 V < V_{CC} < 5.5 V	–0.5		0.8	V
	Input low voltage (USB+, USB–)	4.5 V < V_{CC} < 5.5 V	V_{SS}– 0.5		0.8	V
V_{OL}	Output low voltage (Ports A,B,C,D)	I_{OL} = 20 µA V_{CC} = 4.5 V		0.01	0.1	V
		I_{OL} = 8 mA V_{CC} = 4.5 V		0.25	0.45	V
V_{OL1}	Output low voltage (Ports 1,2,3,4, $\overline{WR}$, $\overline{RD}$)	I_{OL} = 1.6 mA			0.45	V
V_{OL2}	Output low voltage (Port 0, ALE, PSEN)	I_{OL} = 3.2 mA			0.45	V
V_{OH}	Output high voltage (Ports A,B,C,D)	I_{OH} = –20 µA V_{CC} = 4.5 V	4.4	4.49		V
		I_{OH} = –2 mA V_{CC} = 4.5 V	2.4	3.9		V
V_{OH2}	Output high voltage (Port 0 in ext. Bus mode, ALE, PSEN)	I_{OH} = –800 µA	2.4			V
		I_{OH} = –80 µA	4.05			V
V_{LVR}	Low Voltage $\overline{RESET}$	0.1 V hysteresis	3.75	4.0	4.25	V
V_{OP}	XTAL open bias voltage (XTAL1, XTAL2)	I_{OL} = 3.2 mA	2.0		3.0	V
V_{LKO}	V_{CC}(min) for Flash Erase and Program		2.5		4.2	V
I_{IL}	Logic '0' input current (Ports 1,2,3,4)	V_{IN} = 0.45 V (0 V for Port 4[pin 2])	–10		–50	µA
I_{TL}	Logic 1-to-0 transition current (Ports 1,2,3,4)	V_{IN} = 3.5 V (2.5 V for Port 4[pin 2])	–65		–650	µA
I_{RST}	Reset pin pull-up current ($\overline{RESET}$)	V_{IN} = V_{SS}	–10		–55	µA

Figure 80. Input macrocell timing (product term clock)

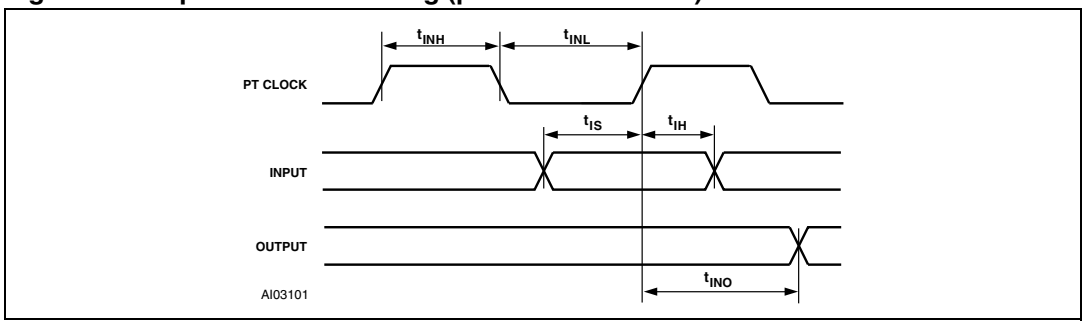


Table 133. Input macrocell timing (5 V devices)

Symbol	Parameter	Conditions	Min.	Max.	PT aloc	Turbo Off	Unit
t_{IS}	Input setup time	(Note 1)	0				ns
t_{IH}	Input hold time	(Note 1)	15			+ 10	ns
t_{INH}	NIB input high time	(Note 1)	9				ns
t_{INL}	NIB input low time	(Note 1)	9				ns
t_{INO}	NIB input to combinatorial delay	(Note 1)		34	+ 2	+ 10	ns

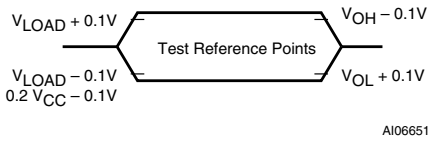
1. Inputs from Port A, B, and C relative to register/ latch clock from the PLD. ALE/AS latch timings refer to t_{AVLX} and t_{LXAX}.

Table 134. Input macrocell timing (3 V devices)

Symbol	Parameter	Conditions	Min.	Max.	PT aloc	Turbo Off	Unit
t_{IS}	Input setup time	(Note 1)	0				ns
t_{IH}	Input hold time	(Note 1)	25			+ 20	ns
t_{INH}	NIB input high time	(Note 1)	12				ns
t_{INL}	NIB input low time	(Note 1)	12				ns
t_{INO}	NIB input to combinatorial delay	(Note 1)		46	+ 4	+ 20	ns

1. Inputs from Port A, B, and C relative to register/latch clock from the PLD. ALE latch timings refer to t_{AVLX} and t_{LXAX}.

Figure 86. PSD module AC float I/O waveform



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

Figure 87. External clock cycle

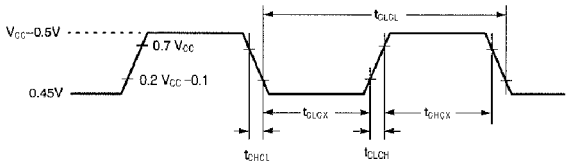
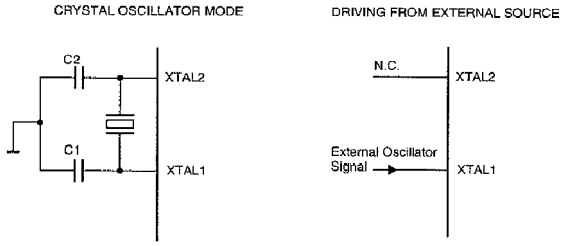


Figure 88. Recommended oscillator circuits



1. C1, C2 = 30 pF ± 10 pF for crystals
2. For ceramic resonators, contact resonator manufacturer
3. Oscillation circuit is designed to be used either with a ceramic resonator or crystal oscillator. Since each crystal and ceramic resonator
4. have their own characteristics, the user should consult the crystal manufacturer for appropriate values of external components.

Figure 89. PSD module AC measurement I/O waveform