Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

PSD813F1A

Flash in-system programmable (ISP) peripherals for 8-bit MCUs, 5 V

NOT FOR NEW DESIGN

FEATURES SUMMARY

- DUAL BANK FLASH MEMORIES
  - 1 Mbit of Primary Flash Memory (8 Uniform Sectors)
  - 256 Kbit Secondary EEPROM (4 Uniform Sectors)
  - Concurrent operation: read from one memory while erasing and writing the other
- 16 Kbit SRAM
- PLD WITH MACROCELLS
  - Over 3,000 Gates Of PLD: DPLD and CPLD
  - DPLD - User-defined Internal chip-select decoding
  - CPLD with 16 Output Macrocells (OMCs) and 24 Input Macrocells (IMCs)
- 27 RECONFIGURABLE I/Os
  - 27 individually configurable I/O port pins that can be used for the following functions (16 I/O ports configurable as open-drain outputs):

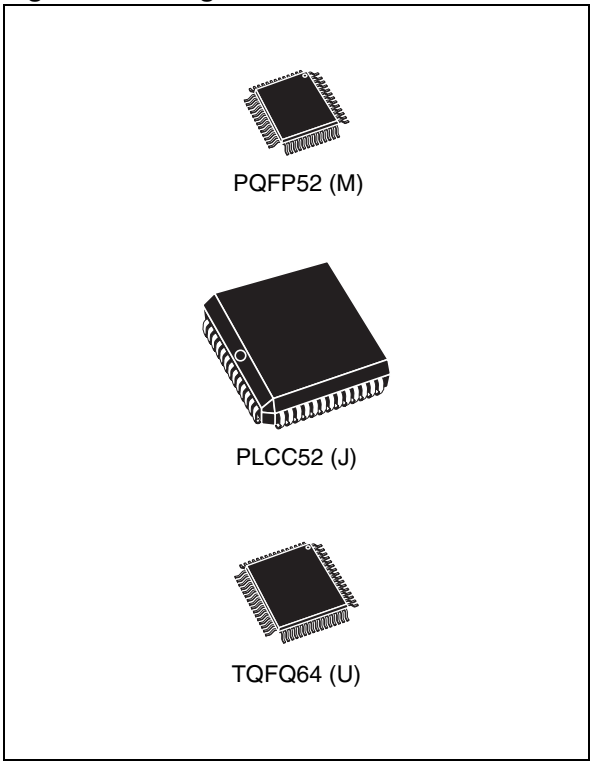
    MCU I/Os

    PLD I/Os

    Latched MCU address output; and

    Special function I/Os
- ENHANCED JTAG SERIAL PORT
  - Built-in JTAG-compliant serial port allows full-chip In-System Programmability (ISP)
  - Efficient manufacturing allows for easy product testing and programming
- PAGE REGISTER
  - Internal page register that can be used to expand the microcontroller address space by a factor of 256.
- PROGRAMMABLE POWER MANAGEMENT

Figure 1. Packages



- HIGH ENDURANCE:
  - 100,000 Erase/WRITE Cycles of Flash Memory
  - 10,000 Erase/WRITE Cycles of EEPROM
  - 1,000 Erase/WRITE Cycles of PLD
  - Data Retention: 15-year minimum at 90°C (for Main Flash, Boot, PLD and Configuration bits).
- SINGLE SUPPLY VOLTAGE:
  - 5V±10% for 5V
- STANDBY CURRENT AS LOW AS 50µA
- Packages are ECOPACK®

This is information on a product still in production but not recommended for new designs.

TABLE OF CONTENTS

SUMMARY DESCRIPTION

The PSD family of Programmable Microcontroller (MCU) Peripherals brings In-System Programmability (ISP) to Flash memory and programmable logic. The result is a simple and flexible solution for embedded designs. PSD devices combine many of the peripheral functions found in MCU based applications.

PSD devices integrate an optimized “microcontroller macrocell” logic architecture. The Macrocell was created to address the unique requirements of embedded system designs. It allows direct connection between the system address/data bus and the internal PSD registers to simplify communication between the MCU and other supporting devices.

The PSD family offers two methods to program PSD Flash memory while the PSD is soldered to a circuit board.

In-System Programming (ISP) via JTAG

An IEEE 1149.1 compliant JTAG interface is included on the PSD enabling the entire device (Flash memory, EEPROM, the PLD, and all configuration) to be rapidly programmed while soldered to the circuit board. This requires no MCU participation, which means the PSD can be programmed anytime, even while completely blank.

The innovative JTAG interface to Flash memories is an industry first, solving key problems faced by designers and manufacturing houses, such as:

First time programming. How do I get firmware into the Flash the very first time? JTAG is the answer, program the PSD while blank with no MCU involvement.

Inventory build-up of pre-programmed devices. How do I maintain an accurate count of pre-programmed Flash memory and PLD devices based on customer demand? How many and what version? JTAG is the answer, build your hardware with blank PSDs soldered directly to the board and then custom program just before they are shipped to customer. No more labels on chips and no more wasted inventory.

Expensive sockets. How do I eliminate the need for expensive and unreliable sockets? JTAG is the answer. Solder the PSD directly to the circuit board. Program first time and subsequent times with JTAG. No need to handle devices and bend the fragile leads.

In-Application Programming (IAP)

Two independent memory arrays (Flash and EEPROM) are included so the MCU can execute code from one memory while erasing and programming the other. Robust product firmware updates in the field are possible over any communication channel (CAN, Ethernet, UART, J1850, etc.) using this unique architecture. Designers are relieved of these problems:

Simultaneous read and write to Flash memory. How can the MCU program the same memory from which it is executing code? It cannot. The PSD allows the MCU to operate the two memories concurrently, reading code from one while erasing and programming the other during IAP.

Complex memory mapping. I have only a 64K-byte address space to start with. How can I map these two memories efficiently? A Programmable Decode PLD is the answer. The concurrent PSD memories can be mapped anywhere in MCU address space, segment by segment with extremely high address resolution. As an option, the secondary Flash memory can be swapped out of the system memory map when IAP is complete. A built-in page register breaks the 64K-byte address limit.

Separate program and data space. How can I write to Flash or EEPROM memory while it resides in “program” space during field firmware updates, my MCU won’t allow it! The Flash PSD provides means to “reclassify” Flash or EEPROM memory as “data” space during IAP, then back to “program” space when complete.

PSDsoft Express

PSDsoft Express, a software development tool from ST, guides you through the design process step-by-step making it possible to complete an embedded MCU design capable of ISP/IAP in just hours. Select your MCU and PSDsoft Express takes you through the remainder of the design with point and click entry, covering PSD selection, pin definitions, programmable logic inputs and outputs, MCU memory map definition, ANSI-C code generation for your MCU, and merging your MCU firmware with the PSD design. When complete, two different device programmers are supported directly from PSDsoft Express: FlashLINK (JTAG) and PSDpro.

Figure 2. PQFP52 Connections

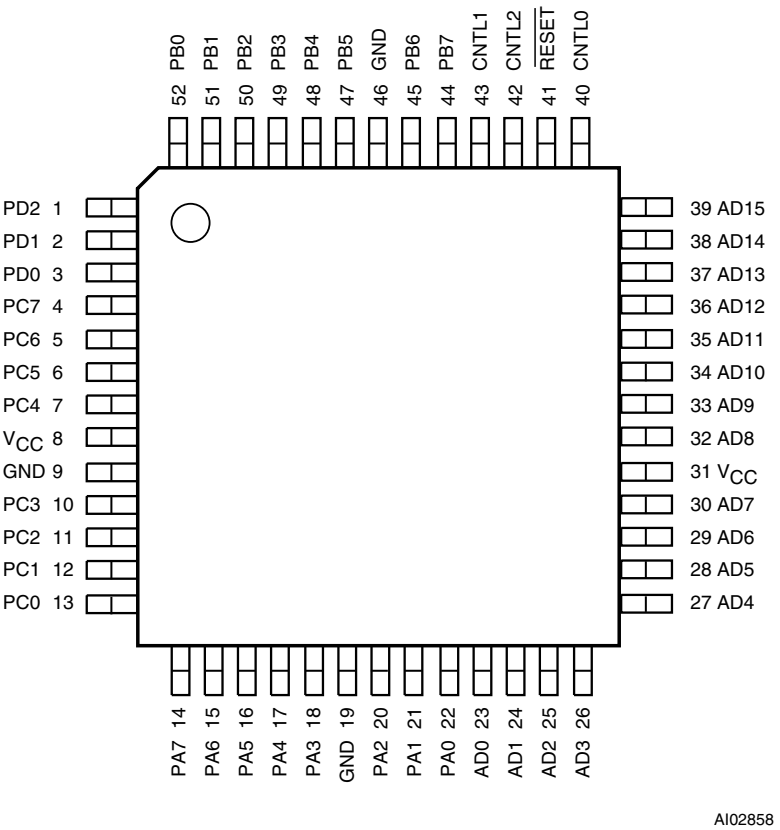


Figure 3. PLCC52 Connections

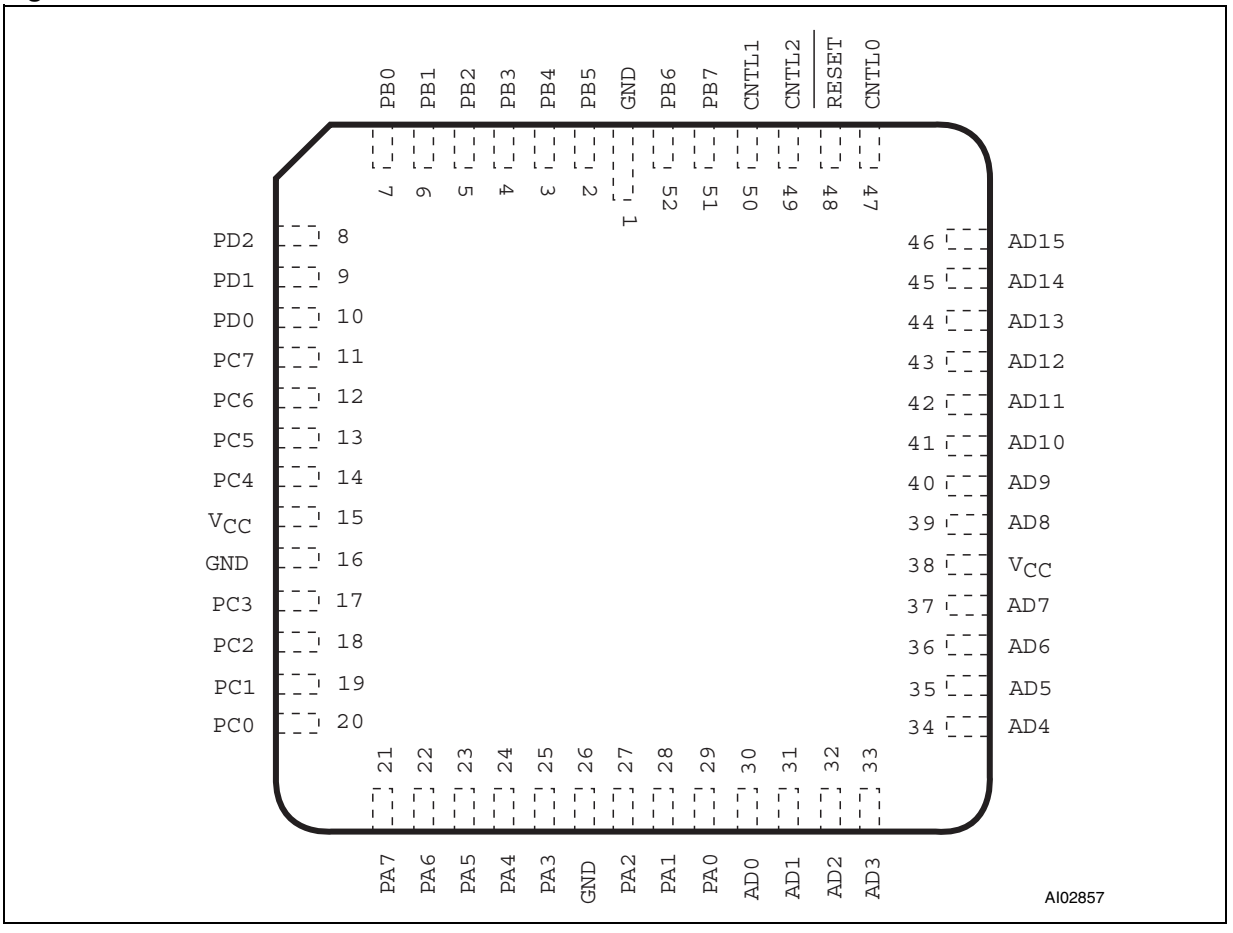
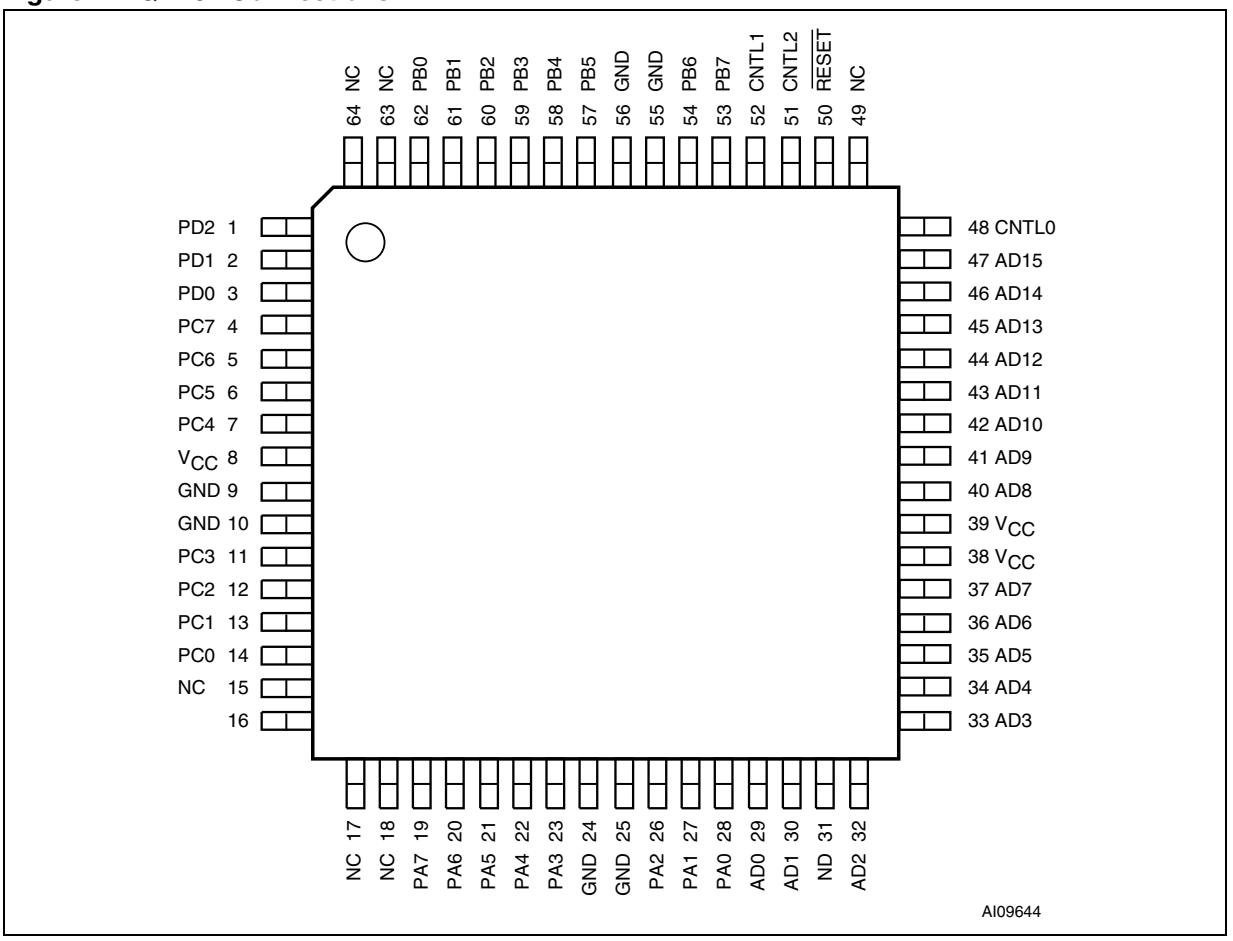


Figure 4. TQFP64 Connections



PIN DESCRIPTION

Table 1. Pin Description (for the PLCC52 package)

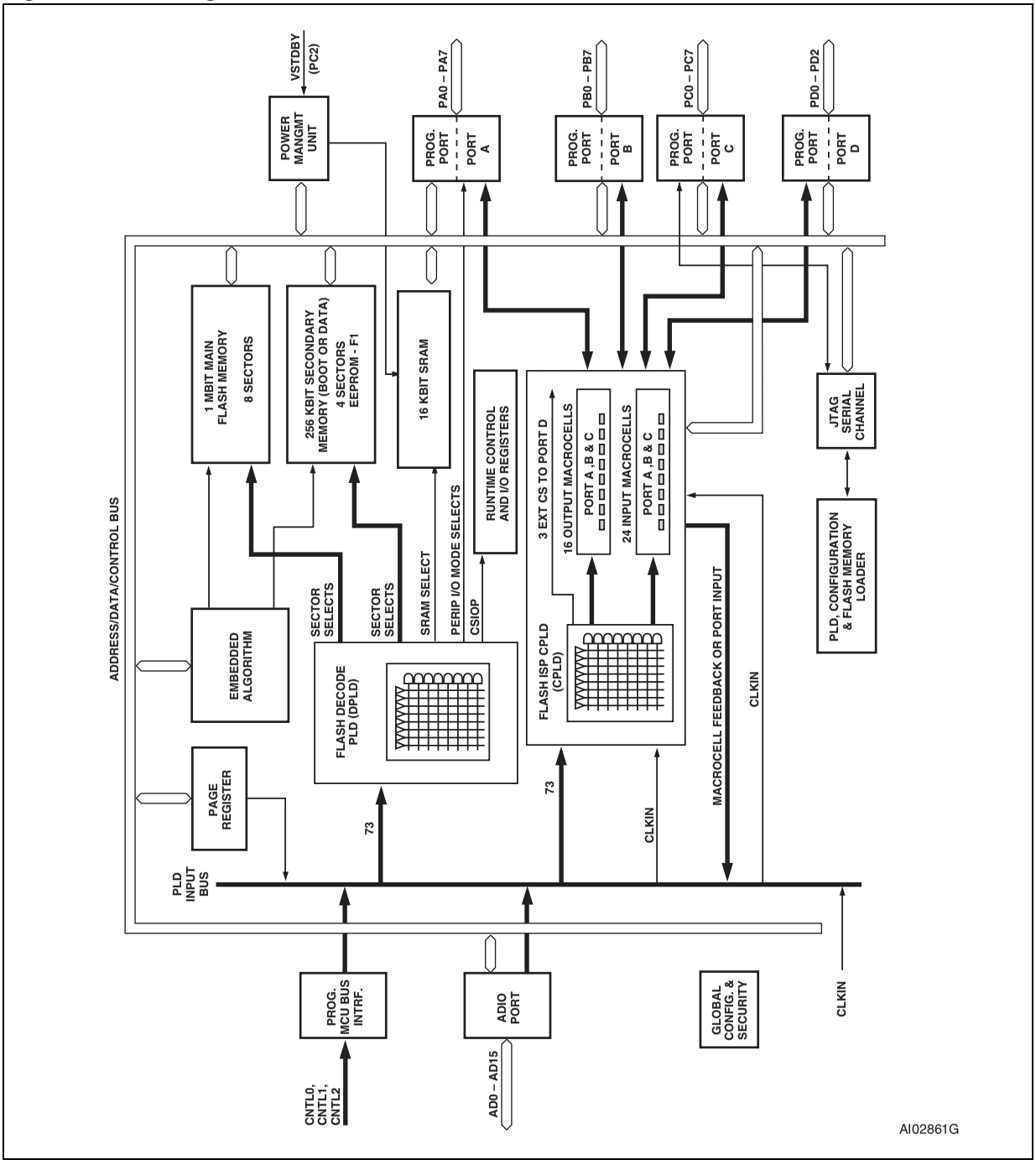
Pin Name	Pin	Type	Description(1)
ADIO0-7	30-37	I/O	This is the lower Address/Data port. Connect your MCU address or address/data bus according to the following rules: 1. If your MCU has a multiplexed address/data bus where the data is multiplexed with the lower address bits, connect AD0-AD7 to this port. 2. If your MCU does not have a multiplexed address/data bus, or you are using an 80C251 in page mode, connect A0-A7 to this port. 3. If you are using an 80C51XA in burst mode, connect A4/D0 through A11/D7 to this port. ALE or AS latches the address. The PSD drives data out only if the READ signal is active and one of the PSD functional blocks was selected. The addresses on this port are passed to the PLDs.
ADIO8-15	39-46	I/O	This is the upper Address/Data port. Connect your MCU address or address/data bus according to the following rules: 1. If your MCU has a multiplexed address/data bus where the data is multiplexed with the lower address bits, connect A8-A15 to this port. 2. If your MCU does not have a multiplexed address/data bus, connect A8-A15 to this port. 3. If you are using an 80C251 in page mode, connect AD8-AD15 to this port. 4. If you are using an 80C51XA in burst mode, connect A12/D8 through A19/D15 to this port. ALE or AS latches the address. The PSD drives data out only if the READ signal is active and one of the PSD functional blocks was selected. The addresses on this port are passed to the PLDs.
CNTL0	47	I	The following control signals can be connected to this port, based on your MCU: 1. $\overline{WR}$ – active Low Write Strobe input. 2. R_$\overline{W}$ – active High READ/active Low write input. This port is connected to the PLDs. Therefore, these signals can be used in decode and other logic equations.
CNTL1	50	I	The following control signals can be connected to this port, based on your MCU: 1. $\overline{RD}$ – active Low Read Strobe input. 2. E – E clock input. 3. $\overline{DS}$ – active Low Data Strobe input. 4. $\overline{PSEN}$ – connect $\overline{PSEN}$ to this port when it is being used as an active Low READ signal. For example, when the 80C251 outputs more than 16 address bits, $\overline{PSEN}$ is actually the READ signal. This port is connected to the PLDs. Therefore, these signals can be used in decode and other logic equations.
CNTL2	49	I	This port can be used to input the $\overline{PSEN}$ (Program Select Enable) signal from any MCU that uses this signal for code exclusively. If your MCU does not output a Program Select Enable signal, this port can be used as a generic input. This port is connected to the PLDs.
Reset	48	I	Active Low Reset input. Resets I/O Ports, PLD macrocells and some of the Configuration Registers. Must be Low at Power-up.

Pin Name	Pin	Type	Description(1)
PA0 PA1 PA2 PA3 PA4 PA5 PA6 PA7	29 28 27 25 24 23 22 21	I/O	These pins make up Port A. These port pins are configurable and can have the following functions: 1. MCU I/O – write to or read from a standard output or input port. 2. CPLD macrocell (McellAB0-7) outputs. 3. Inputs to the PLDs. 4. Latched address outputs (see Table 5). 5. Address inputs. For example, PA0-3 could be used for A0-A3 when using an 80C51XA in burst mode. 6. As the data bus inputs D0-D7 for non-multiplexed address/data bus MCUs. 7. D0/A16-D3/A19 in M37702M2 mode. 8. Peripheral I/O mode. Note: PA0-PA3 can only output CMOS signals with an option for high slew rate. However, PA4-PA7 can be configured as CMOS or Open Drain Outputs.
PB0 PB1 PB2 PB3 PB4 PB5 PB6 PB7	7 6 5 4 3 2 52 51	I/O	These pins make up Port B. These port pins are configurable and can have the following functions: 1. MCU I/O – write to or read from a standard output or input port. 2. CPLD macrocell (McellAB0-7 or McellBC0-7) outputs. 3. Inputs to the PLDs. 4. Latched address outputs (see Table 5). Note: PB0-PB3 can only output CMOS signals with an option for high slew rate. However, PB4-PB7 can be configured as CMOS or Open Drain Outputs.
PC0	20	I/O	PC0 pin of Port C. This port pin can be configured to have the following functions: 1. MCU I/O – write to or read from a standard output or input port. 2. CPLD macrocell (McellBC0) output. 3. Input to the PLDs. 4. TMS Input[2] for the JTAG Interface. This pin can be configured as a CMOS or Open Drain output.
PC1	19	I/O	PC1 pin of Port C. This port pin can be configured to have the following functions: 1. MCU I/O – write to or read from a standard output or input port. 2. CPLD macrocell (McellBC1) output. 3. Input to the PLDs. 4. TCK Input[2] for the JTAG Interface. This pin can be configured as a CMOS or Open Drain output.
PC2	18	I/O	PC2 pin of Port C. This port pin can be configured to have the following functions: 1. MCU I/O – write to or read from a standard output or input port. 2. CPLD macrocell (McellBC2) output. 3. Input to the PLDs. This pin can be configured as a CMOS or Open Drain output.
PC3	17	I/O	PC3 pin of Port C. This port pin can be configured to have the following functions: 1. MCU I/O – write to or read from a standard output or input port. 2. CPLD macrocell (McellBC3) output. 3. Input to the PLDs. 4. $\overline{\text{TSTAT}}$ output[2] for the JTAG Serial Interface. 5. Ready/$\overline{\text{Busy}}$ output for In-System parallel programming. This pin can be configured as a CMOS or Open Drain output.
PC4	14	I/O	PC4 pin of Port C. This port pin can be configured to have the following functions: 1. MCU I/O – write to or read from a standard output or input port. 2. CPLD macrocell (McellBC4) output. 3. Input to the PLDs. 4. $\overline{\text{TERR}}$ output[2] for the JTAG Interface. This pin can be configured as a CMOS or Open Drain output.

Pin Name	Pin	Type	Description(1)
PC5	13	I/O	PC5 pin of Port C. This port pin can be configured to have the following functions: 1. MCU I/O – write to or read from a standard output or input port. 2. CPLD macrocell (McellBC5) output. 3. Input to the PLDs. 4. TDI input[2] for the JTAG Interface. This pin can be configured as a CMOS or Open Drain output.
PC6	12	I/O	PC6 pin of Port C. This port pin can be configured to have the following functions: 1. MCU I/O – write to or read from a standard output or input port. 2. CPLD macrocell (McellBC6) output. 3. Input to the PLDs. 4. TDO output[2] for the JTAG Interface. This pin can be configured as a CMOS or Open Drain output.
PC7	11	I/O	PC7 pin of Port C. This port pin can be configured to have the following functions: 1. MCU I/O – write to or read from a standard output or input port. 2. CPLD macrocell (McellBC7) output. 3. Input to the PLDs. 4. DBE – active Low Data Byte Enable input from 68HC912 type MCUs. This pin can be configured as a CMOS or Open Drain output.
PD0	10	I/O	PD0 pin of Port D. This port pin can be configured to have the following functions: 1. ALE/AS input latches address output from the MCU. 2. MCU I/O – write or read from a standard output or input port. 3. Input to the PLDs. 4. CPLD output (External Chip Select).
PD1	9	I/O	PD1 pin of Port D. This port pin can be configured to have the following functions: 1. MCU I/O – write to or read from a standard output or input port. 2. Input to the PLDs. 3. CPLD output (External Chip Select). 4. CLKIN – clock input to the CPLD macrocells, the APD Unit's Power-down counter, and the CPLD AND Array.
PD2	8	I/O	PD2 pin of Port D. This port pin can be configured to have the following functions: 1. MCU I/O – write to or read from a standard output or input port. 2. Input to the PLDs. 3. CPLD output (External Chip Select). 4. PSD Chip Select Input ($\overline{CSI}$). When Low, the MCU can access the PSD memory and I/O. When High, the PSD memory blocks are disabled to conserve power.
V_{CC}	15, 38		Supply Voltage
GND	1, 16, 26		Ground pins

Note: 1. The pin numbers in this table are for the PLCC package only. See the Figure 2., page 7, for pin numbers on other package type.
2. These functions can be multiplexed with other functions.

Figure 5. Block Diagram



PSD ARCHITECTURAL OVERVIEW

PSD devices contain several major functional blocks. Figure 5 shows the architecture of the PSD device. The functions of each block are described briefly in the following sections. Many of the blocks perform multiple functions and are user configurable.

Memory

The PSD contains the following memories:

- a 1 Mbit Flash memory
- a secondary 256 Kbit EEPROM memory
- a 16 Kbit SRAM

Each of the memory blocks is briefly discussed in the following paragraphs. A more detailed discussion can be found in the section entitled MEMORY BLOCKS, page 18.

The 1 Mbit Flash memory is the main memory of the PSD. It is divided into 8 equally-sized sectors that are individually selectable.

The 256 Kbit EEPROM or Flash memory is divided into 4 equally-sized sectors. Each sector is individually selectable.

The 16 Kbit SRAM is intended for use as a scratchpad memory or as an extension to the microcontroller SRAM.

Each sector of memory can be located in a different address space as defined by the user. The access times for all memory types includes the address latching and DPLD decoding time.

PLDs

The device contains two PLD blocks, each optimized for a different function, as shown in Table 2. The functional partitioning of the PLDs reduces power consumption, optimizes cost/performance, and eases design entry.

The Decode PLD (DPLD) is used to decode addresses and generate chip selects for the PSD internal memory and registers. The CPLD can implement user-defined logic functions. The DPLD has combinatorial outputs. The CPLD has 16 Output macrocells and 3 combinatorial outputs. The PSD also has 24 Input macrocells that can be configured as inputs to the PLDs. The PLDs receive their inputs from the PLD Input Bus and are differentiated by their output destinations, number of Product Terms, and macrocells.

The PLDs consume minimal power by using Zero-Power design techniques. The speed and power consumption of the PLD is controlled by the Turbo Bit (ZPSD only) in the PMMR0 register and other bits in the PMMR2 registers. These registers are set by the microcontroller at runtime. There is a slight penalty to PLD propagation time when invoking the ZPSD features.

I/O Ports

The PSD has 27 I/O pins divided among four ports (Port A, B, C, and D). Each I/O pin can be individually configured for different functions. Ports A, B, C and D can be configured as standard MCU I/O ports, PLD I/O, or latched address outputs for microcontrollers using multiplexed address/data busses.

The JTAG pins can be enabled on Port C for In-System Programming (ISP).

Ports A and B can also be configured as a data port for a n on-multiplexed bus or multiplexed Address/Data buses for certain types of 16-bit microcontrollers.

Microcontroller Bus Interface

The PSD easily interfaces with most 8-bit microcontrollers that have either multiplexed or non-multiplexed address/data busses. The device is configured to respond to the microcontroller's control signals, which are also used as inputs to the PLDs. Where there is a requirement to use a 16-bit data bus to interface to a 16-bit microcontroller, two PSDs must be used. For examples, please see the section entitled MCU Bus Interface Examples, page 47.

Table 2. PLD I/O

Name	Inputs	Outputs	Product Terms
Decode PLD (DPLD)	73	17	42
Complex PLD (CPLD)	73	19	140

JTAG Port

In-System Programming can be performed through the JTAG pins on Port C. This serial interface allows complete programming of the entire PSD device. A blank device can be completely programmed. The JTAG signals (TMS, TCK, TSTAT, TERR, TDI, TDO) can be multiplexed with other functions on Port C. Table 3 indicates the JTAG signals pin assignments.

In-System Programming (ISP)

Using the JTAG signals on Port C, the entire PSD device can be programmed or erased without the use of the microcontroller. The main Flash memory can also be programmed in-system by the microcontroller executing the programming algorithms out of the EEPROM or SRAM. The EEPROM can be programmed the same way by executing out of the main Flash memory. The PLD logic or other PSD configuration can be programmed through the JTAG port or a device programmer. Table 4 indicates which programming methods can program different functional blocks of the PSD.

Page Register

The 8-bit Page Register expands the address range of the microcontroller by up to 256 times. The paged address can be used as part of the address space to access external memory and peripherals, or internal memory and I/O. The Page Register can also be used to change the address mapping of blocks of Flash memory into different memory spaces for in-circuit programming.

Power Management Unit (PMU)

The Power Management Unit (PMU) in the PSD gives the user control of the power consumption on selected functional blocks based on system requirements. The PMU includes an Automatic Power Down unit (APD) that will turn off device functions due to microcontroller inactivity. The APD unit has a Power Down Mode that helps reduce power consumption.

The PSD also has some bits that are configured at run-time by the MCU to reduce power consumption of the CPLD. The turbo bit in the PMMR0 register can be turned off and the CPLD will latch its outputs and go to sleep until the next transition on its inputs.

Additionally, bits in the PMMR2 register can be set by the MCU to block signals from entering the CPLD to reduce power consumption. Please see the section entitled POWER MANAGEMENT, page 64 for more details.

Table 3. JTAG SIgnals on Port C

Port C Pins	JTAG Signal
PC0	TMS
PC1	TCK
PC3	$\overline{\text{TSTAT}}$
PC4	$\overline{\text{TERR}}$
PC5	TDI
PC6	TDO

Table 4. Methods of Programming Different Functional Blocks of the PSD

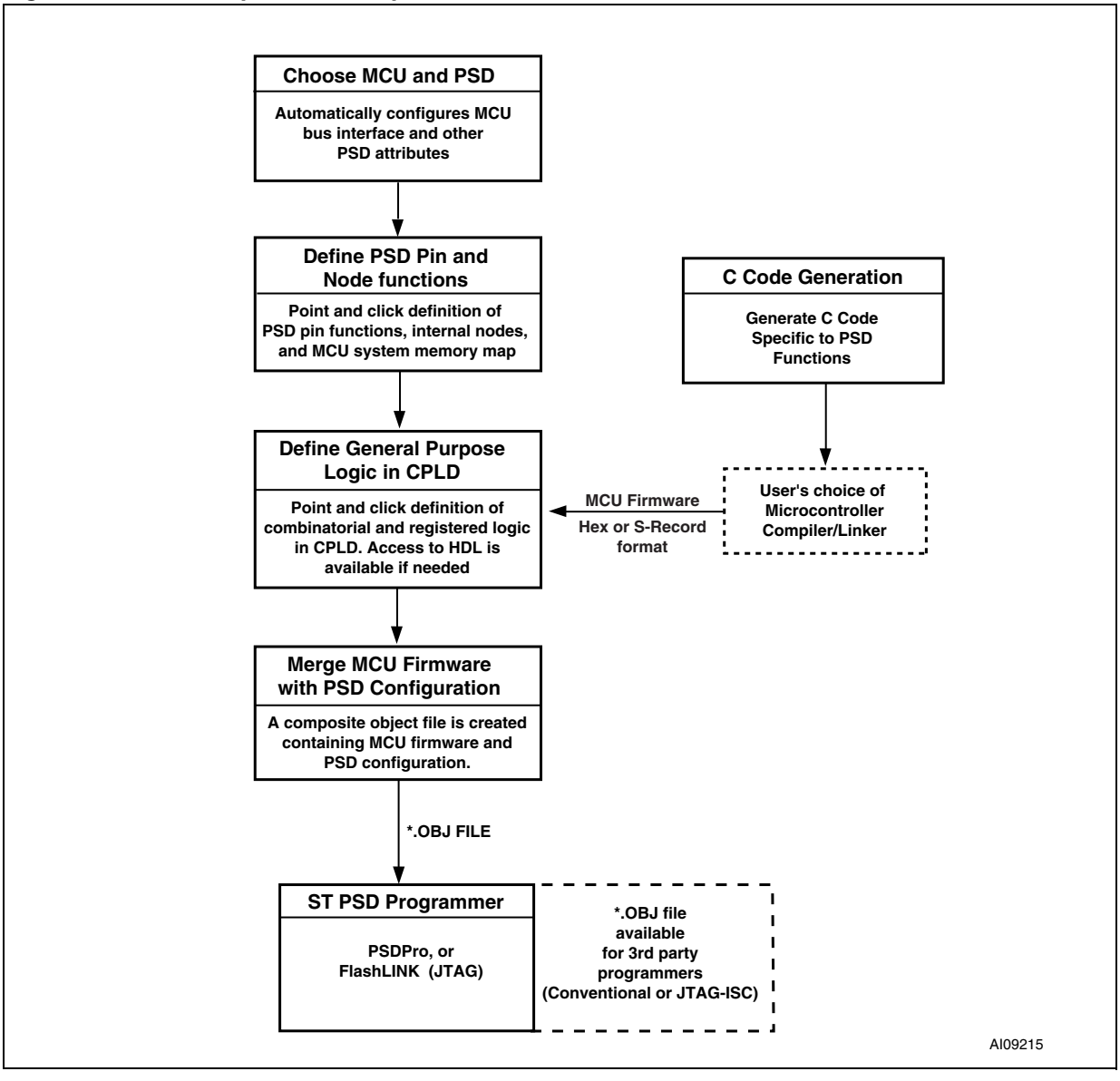
Functional Block	JTAG Programming	Device Programmer	In-System Parallel Programming
Main Flash Memory	Yes	Yes	Yes
EEPROM Memory	Yes	Yes	Yes
PLD Array (DPLD and CPLD)	Yes	Yes	No
PSD Configuration	Yes	Yes	No
Optional OTP Row	No	Yes	Yes

DEVELOPMENT SYSTEM

The PSD is supported by PSDsoft Express a Windows-based (95, 98, NT) software development tool. A PSD design is quickly and easily produced in a point and click environment. The designer does not need to enter Hardware Definition Language (HDL) equations (unless desired) to define PSD pin functions and memory map information. The general design flow is shown in Figure 6 below. PSDsoft Express is available from our web site (www.st.com/psm) or other distribution channels.

PSDsoft Express directly supports two low cost device programmers from ST, PSDpro and FlashLINK (JTAG). Both of these programmers may be purchased through your local distributor/representative, or directly from our web site using a credit card. The PSD is also supported by third party device programmers, see web site for current list.

Figure 6. PSDsoft Express Development Tool



PSD REGISTER DESCRIPTION AND ADDRESS OFFSET

Table 5 shows the offset addresses to the PSD registers relative to the CSIOP base address. The CSIOP space is the 256 bytes of address that is allocated by the user to the internal PSD registers. Table 6 provides brief descriptions of the registers in CSIOP space. The following section gives a more detailed description.

Table 5. I/O Port Latched Address Output Assignments

MCU(1)	Port A(2)		Port B(2)	
	Port A (3:0)	Port A (7:4)	Port B (3:0)	Port B (7:4)
8051XA (8-bit)	N/A	Address a7-a4	Address a11-a8	N/A
80C251 (page mode)	N/A	N/A	Address a11-a8	Address a15-a12
All other 8-bit multiplexed	Address a3-a0	Address a7-a4	Address a3-a0	Address a7-a4
8-bit non-multiplexed bus	N/A	N/A	Address a3-a0	Address a7-a4

Note: 1. See the section entitled I/O PORTS, page 52, on how to enable the Latched Address Output function.
2. N/A = Not Applicable

Table 6. Register Address Offset

Register Name	Port A	Port B	Port C	Port D	Other(1)	Description
Data In	00	01	10	11		Reads Port pin as input, MCU I/O input mode
Control	02	03				Selects mode between MCU I/O or Address Out
Data Out	04	05	12	13		Stores data for output to Port pins, MCU I/O output mode
Direction	06	07	14	15		Configures Port pin as input or output
Drive Select	08	09	16	17		Configures Port pins as either CMOS or Open Drain on some pins, while selecting high slew rate on other pins.
Input Macrocell	0A	0B	18			Reads Input Macrocells
Enable Out	0C	0D	1A	1B		Reads the status of the output enable to the I/O Port driver
Output Macrocells AB	20	20				READ – reads output of macrocells AB WRITE – loads macrocell flip-flops
Output Macrocells BC		21	21			READ – reads output of macrocells BC WRITE – loads macrocell flip-flops
Mask Macrocells AB	22	22				Blocks writing to the Output Macrocells AB
Mask Macrocells BC		23	23			Blocks writing to the Output Macrocells BC
Primary Flash Protection					C0	Read only – Flash Sector Protection
Secondary Flash memory Protection					C2	Read only – PSD Security and EEPROM Sector Protection
JTAG Enable					C7	Enables JTAG Port
PMMR0					B0	Power Management Register 0
PMMR2					B4	Power Management Register 2
Page					E0	Page Register
VM					E2	Places PSD memory areas in Program and/or Data space on an individual basis.

Note: 1. Other registers that are not part of the I/O ports.

DETAILED OPERATION

As shown in Figure 5., page 13, the PSD consists of six major types of functional blocks:

- Memory Blocks
- PLD Blocks
- MCU Bus Interface
- 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.

MEMORY BLOCKS

The PSD has the following memory blocks (see Table 7):

– The Main Flash memory
– Secondary EEPROM memory
– SRAM

The Memory Select signals for these blocks originate from the Decode PLD (DPLD) and are user-defined in PSDsoft Express.

Primary Flash Memory and Secondary EEPROM Description

The 1Mb primary Flash memory is divided evenly into eight 16-KByte sectors. The EEPROM memory is divided into four sectors of eight KBytes each. Each sector of either memory can be separately protected from Program and Erase operations.

Flash memory may be erased on a sector-by-sector basis and programmed byte-by-byte. Flash sector erasure may be suspended while data is read from other sectors of memory and then resumed after reading.

EEPROM may be programmed byte-by-byte or sector-by-sector, and erasing is automatic and transparent. The integrity of the data can be secured with the help of Software Data Protection (SDP). Any write operation to the EEPROM is inhibited during the first five milliseconds following power-up.

During a program or erase of Flash, or during a write of the EEPROM, the status can be output on the Ready/Busy (PC3) pin of Port C3. This pin is set up using PSDsoft Express Configuration.

Memory Block Select Signals. The decode PLD in the PSD generates the chip selects for all the internal memory blocks (refer to the section entitled PLD'S, page 34). Each of the eight Flash memory sectors have a Flash Select signal (FS0-FS7) which can contain up to three product terms. Each of the four EEPROM memory sectors have a Select signal (EES0-3 or CSBOOT0-3) which can contain up to three product terms. Having three product terms for each sector select signal allows a given sector to be mapped in different areas of system memory. When using a microcontroller with separate Program and Data space, these flexible select signals allow dynamic re-mapping of sectors from one space to the other.

Ready/Busy Pin (PC3). Pin PC3 can be used to output the Ready/Busy status of the PSD. The output on the pin will be a '0' (Busy) when Flash or EEPROM memory blocks are being written to, or when the Flash memory block is being erased. The output will be a '1' (Ready) when no write or erase operation is in progress.

Table 7. Memory Blocks

Device	Main Flash	EEPROM	SRAM
PSD813F1A	128KB	32KB	2KB

Memory Operation

The main Flash and EEPROM memory are addressed through the microcontroller interface on the PSD device. The microcontroller can access these memories in one of two ways:

– The microcontroller can execute a typical bus WRITE or READ operation just as it would if accessing a RAM or ROM device using standard bus cycles.

– The microcontroller can execute a specific instruction that consists of several WRITE and READ operations. This involves writing specific data patterns to special addresses within the Flash or EEPROM to invoke an embedded algorithm. These instructions are summarized in Table 8., page 20.

Typically, Flash memory can be read by the microcontroller using READ operations, just as it would read a ROM device. However, Flash memory can only be erased and programmed with specific instructions. For example, the microcontroller cannot write a single byte directly to Flash memory as one would write a byte to RAM. To program a byte into Flash memory, the microcontroller must execute a program instruction sequence, then test the status of the programming event. This status test is achieved by a READ operation or polling the Ready/Busy pin (PC3).

The Flash memory can also be read by using special instructions to retrieve particular Flash device information (sector protect status and ID).

The EEPROM is a bit different. Data can be written to EEPROM memory using write operations, like writing to a RAM device, but the status of each WRITE event must be checked by the microcontroller. A WRITE event can be one to 64 contiguous bytes. The status test is very similar to that used for Flash memory (READ operation or Ready/Busy). Optionally, the EEPROM memory may be put into a Software Data Protect (SDP) mode where it requires instructions, rather than operations, to alter its contents. SDP mode makes writing to EEPROM much like writing to Flash memory.

Table 8. Instructions

Instruction	EEPROM Sector Select (EESi)	Flash Sector Select (FSi)(2)	Cycle 1	Cycle 2	Cycle 3	Cycle 4	Cycle 5	Cycle 6	Cycle 7
Read Flash Identifier[3,5]	0	1	AAh@ X555h	55h@ XAAAh	90h@ X555h	Read Identifier with (A6,A1,A0 at 0,0,1)			
Read OTP row[4]	1	0	AAh@ X555h	55h@ XAAAh	90h@ X555h	Read byte 1	Read byte 2		Read byte N
Read Sector Protection Status[3,5]	0	1	AAh@ X555h	55h@ XAAAh	90h@ X555h	Read identifier with (A6, A1; A0 = 0,1,0)			
Program a Flash Byte[5]	0	1	AAh@ X555h	55h@ XAAAh	A0h@ X555h	Data@ address			
Erase one Flash Sector[5]	0	1	AAh@ X555h	55h@ XAAAh	80h@ X555h	AAh@ X555h	55h@ XAAAh	30h@ Sector address	30h@ Sector address[1]
Erase the Whole Flash[5]	0	1	AAh@ X555h	55h@ XAAAh	80h@ X555h	AAh@ X555h	55h@ XAAAh	10h@ X555h	
Suspend Sector Erase[5]	0	1	B0h@ XXXXh						
Resume Sector Erase[5]	0	1	30h@ XXXXh						
EEPROM Power Down[4]	1	0	AAh@ X555h	55h@ XAAAh	30h@ X555h				
SDP Enable/ EEPROM Write[4]	1	0	AAh@ X555h	55h@ XAAAh	A0h@ X555h	Write byte 1	Write byte 2		Write byte N
SDP Disable[4]	1	0	AAh@ X555h	55h@ XAAAh	80h@ X555h	AAh@ X555h	55h@ XAAAh	20h@ X555h	
Write in OTP Row[4,6]	1	0	AAh@ X555h	55h@ XAAAh	B0h@ X555h	Write byte 1	Write byte 2		Write byte N
Return (from OTP Read or EEPROM Power-Down)[4]	1	0	F0h@ XXXX						
Reset[3.5]	0	1	AAh@ X555h	55h@ XAAAh	F0h@ XXXX				
Reset (short instruction)[5]	0	1	F0h@ XXXX						

Note: 1. Additional sectors to be erased must be entered within 80 µs. A Sector Address is any address within the Sector.
2. Flash and EEPROM Sector Selects are active high. Addresses A15-A12 are don't cares in Instruction Bus Cycles.
3. The Reset instruction is required to return to the normal READ mode if DQ5 goes high or after reading the Flash Identifier or Protection status.
4. The MCU cannot invoke these instructions while executing code from EEPROM. The MCU must be operating from some other memory when these instructions are performed.
5. The MCU cannot invoke these instructions while executing code from the same Flash memory for which the instruction is intended. The MCU must operate from some other memory when these instructions are executed.
6. Writing to OTP Row is allowed only when SDP mode is disabled.

INSTRUCTIONS

An instruction is defined as a sequence of specific operations. Each received byte is sequentially decoded by the PSD and not executed as a standard write operation. The instruction is executed when the correct number of bytes are properly received and the time between two consecutive bytes is shorter than the time-out value. Some instructions are structured to include READ operations after the initial WRITE operations.

The sequencing of any instruction must be followed exactly. Any invalid combination of instruction bytes or time-out between two consecutive bytes while addressing Flash memory will reset the device logic into READ mode (Flash memory reads like a ROM device). An invalid combination or time-out while addressing the EEPROM block will cause the offending byte to be interpreted as a single operation.

The PSD supports these instructions (see Table 8., page 20):

Flash memory:

- Erase memory by chip or sector
- Suspend or resume sector erase
- Program a Byte
- Reset to READ mode
- Read Flash Identifier value
- Read Sector Protection Status

EEPROM:

- Write data to OTP Row
- Read data from OTP Row
- Power down memory
- Enable Software Data Protect (SDP)
- Disable SDP
- Return from read OTP Row read mode or power down mode.

These instructions are detailed in Table 8., page 20. For efficient decoding of the instructions, the first two bytes of an instruction are the coded cycles and are followed by a command byte or confirmation byte. The coded cycles consist of writing the data AAh to address X555h during the first cycle and data 55h to address XAAAh during the second cycle. Address lines A15-A12 are don't cares during the instruction WRITE cycles. However, the appropriate sector select signal (FSi or EESi) must be selected.

Power-down Instruction and Power-up Mode

EEPROM Power Down Instruction. The EEPROM can enter power down mode with the help of the EEPROM power down instruction (see Table 8., page 20). Once the EEPROM power down instruction is decoded, the EEPROM memory cannot be accessed unless a Return instruction (also in Table 8., page 20) is decoded. Alternately, this power down mode will automatically occur when the APD circuit is triggered (see section entitled Automatic Power-down (APD) Unit and Power-down Mode, page 65). Therefore, this instruction is not required if the APD circuit is used.

Power-up Mode. The PSD internal logic is reset upon power-up to the READ mode. Any write operation to the EEPROM is inhibited during the first 5ms following power-up. The FSi and EESi select signals, along with the write strobe signal, must be in the false state during power-up for maximum security of the data contents and to remove the possibility of a byte being written on the first edge of a write strobe signal. Any write cycle initiation is locked when V_{CC} is below V_{LKO}.

READ

Under typical conditions, the microcontroller may read the Flash or EEPROM memory using READ operations just as it would a ROM or RAM device. Alternately, the microcontroller may use READ operations to obtain status information about a Program or Erase operation in progress. Lastly, the microcontroller may use instructions to read special data from these memories. The following sections describe these READ functions.

Read Memory Contents. Main Flash is placed in the READ mode after power-up, chip reset, or a Reset Flash instruction (see Table 8., page 20). The microcontroller can read the memory contents of main Flash or EEPROM by using READ operations any time the READ operation is not part of an instruction sequence.

Read Main Flash Memory Identifier. The main Flash memory identifier is read with an instruction composed of 4 operations:

3 specific write operations and a READ operation (see Table 8). During the READ operation, address bits A6, A1, and A0 must be 0,0,1, respectively, and the appropriate sector select signal (FSi) must be active. The Flash ID is E3h for the PSD. The MCU can read the ID only when it is executing from the EEPROM.

Read Main Flash Memory Sector Protection Status. The main Flash memory sector protection status is read with an instruction composed of 4 operations: 3 specific WRITE operations and a READ operation (see Table 8., page 20). During the READ operation, address bits A6, A1, and A0 must be 0,1,0, respectively, while the chip select FSi designates the Flash sector whose protection has to be verified. The READ operation will produce 01h if the Flash sector is protected, or 00h if the sector is not protected.

The sector protection status for all NVM blocks (main Flash or EEPROM) can be read by the microcontroller accessing the Flash Protection and PSD/EE Protection registers in PSD I/O space. See Flash Memory and EEPROM Sector Protect, page 30 for register definitions.

Reading the OTP Row. There are 64 bytes of One-Time-Programmable (OTP) memory that reside in EEPROM. These 64 bytes are in addition to the 32 Kbytes of EEPROM memory. A READ of the OTP row is done with an instruction composed of at least 4 operations: 3 specific WRITE operations and one to 64 READ operations (see Table 8., page 20). During the READ operation(s), address bit A6 must be zero, while address bits A5-A0 define the OTP Row byte to be read while any EEPROM sector select signal (EESi) is active. After reading the last byte, an EEPROM Return instruction must be executed (see Table 8., page 20).

Reading the Erase/Program Status Bits. The PSD provides several status bits to be used by the microcontroller to confirm the completion of an erase or programming instruction of Flash memory. Bits are also available to show the status of WRITES to EEPROM. These status bits minimize the time that the microcontroller spends performing these tasks and are defined in Table 9. The status bits can be read as many times as needed.

For Flash memory, the microcontroller can perform a READ operation to obtain these status bits while an Erase or Program instruction is being executed by the embedded algorithm. See the section entitled PROGRAMMING FLASH MEMORY, page 27 for details.

For EEPROM not in SDP mode, the microcontroller can perform a READ operation to obtain these status bits just after a data WRITE operation. The microcontroller may write one to 64 bytes before reading the status bits. See the section entitled Writing to the EEPROM, page 24 for details.

For EEPROM in SDP mode, the microcontroller will perform a READ operation to obtain these status bits while an SDP write instruction is being executed by the embedded algorithm. See section entitled EEPROM Software Data Protect (SDP), page 24 for details.

Table 9. Status Bit

Device	FSi/ CSBOOTi	EESi	DQ7	DQ6	DQ5	DQ4	DQ3	DQ2	DQ1	DQ0
Flash	V_{IH}	V_{IL}	Data Polling	Toggle Flag	Error Flag	X	Erase Timeout	X	X	X
EEPROM	V_{IL}	V_{IH}	Data Polling	Toggle Flag	X	X	X	X	X	X

Note: 1. X = not guaranteed value, can be read either 1 or 0.
2. DQ7-DQ0 represent the Data Bus Bits, D7-D0.
3. FSi and EESi are active High.

Data Polling Flag (DQ7)

When Erasing or Programming the Flash memory (or when Writing into the EEPROM memory), bit DQ7 outputs the complement of the bit being entered for Programming/Writing on DQ7. Once the Program instruction or the WRITE operation is completed, the true logic value is read on DQ7 (in a Read operation). Flash memory specific features:

– Data Polling is effective after the fourth WRITE pulse (for programming) or after the sixth WRITE pulse (for Erase). It must be performed at the address being programmed or at an address within the Flash sector being erased.
– During an Erase instruction, DQ7 outputs a '0.' After completion of the instruction, DQ7 will output the last bit programmed (it is a '1' after erasing).
– If the byte to be programmed is in a protected Flash sector, the instruction is ignored.
– If all the Flash sectors to be erased are protected, DQ7 will be set to '0' for about 100µs, and then return to the previous addressed byte. No erasure will be performed.

Toggle Flag (DQ6)

The PSD offers another way for determining when the EEPROM write or the Flash memory Program instruction is completed. During the internal WRITE operation and when either the FSi or EESi is true, the DQ6 will toggle from '0' to '1' and '1' to '0' on subsequent attempts to read any byte of the memory.

When the internal cycle is complete, the toggling will stop and the data read on the Data Bus D0-7 is the addressed memory byte. The device is now accessible for a new READ or WRITE operation.

The operation is finished when two successive reads yield the same output data. Flash memory specific features:

- The Toggle bit is effective after the fourth WRITE pulse (for programming) or after the sixth WRITE pulse (for Erase).
- If the byte to be programmed belongs to a protected Flash sector, the instruction is ignored.
- If all the Flash sectors selected for erasure are protected, DQ6 will toggle to '0' for about 100 µs and then return to the previous addressed byte.

Error Flag (DQ5)

During a correct Program or Erase, the Error bit will set to '0.' This bit is set to '1' when there is a failure during Flash byte programming, Sector erase, or Bulk Erase.

In the case of Flash programming, the Error Bit indicates the attempt to program a Flash bit(s) from the programmed state ('0') to the erased state ('1'), which is not a valid operation. The Error bit may also indicate a timeout condition while attempting to program a byte.

In case of an error in Flash sector erase or byte program, the Flash sector in which the error occurred or to which the programmed byte belongs must no longer be used. Other Flash sectors may still be used. The Error bit resets after the Reset instruction.

Erase Time-out Flag DQ3 (Flash Memory only)

The Erase Timer bit reflects the time-out period allowed between two consecutive Sector Erase instructions. The Erase timer bit is set to '0' after a Sector Erase instruction for a time period of 100µs + 20% unless an additional Sector Erase instruction is decoded. After this time period or when the additional Sector Erase instruction is decoded, DQ3 is set to '1.'

Writing to the EEPROM

Data may be written a byte at a time to the EEPROM using simple write operations, much like writing to an SRAM. Unlike SRAM though, the completion of each byte write must be checked before the next byte is written. To speed up this process, the PSD offers a Page write feature to allow writing of several bytes before checking status.

To prevent inadvertent writes to EEPROM, the PSD offers a Software Data Protect (SDP) mode. Once enabled, SDP forces the MCU to "unlock" the EEPROM before altering its contents, much like Flash memory programming.

Writing a Byte to EEPROM. A write operation is initiated when an EEPROM select signal (EESi) is true and the write strobe signal ($\overline{WR}$) into the PSD is true. If the PSD detects no additional writes within 120µsec, an internal storage operation is initiated. Internal storage to EEPROM memory technology typically takes a few milliseconds to complete.

The status of the write operation is obtained by the MCU reading the Data Polling or Toggle bits (as detailed in section entitled READ, page 22), or the Ready/$\overline{Busy}$ output pin (section Ready/Busy Pin (PC3), page 18).

Keep in mind that the MCU does not need to erase a location in EEPROM before writing it. Erasure is performed automatically as an internal process.

Writing a Page to EEPROM. Writing data to EEPROM using page mode is more efficient than writing one byte at a time. The PSD EEPROM has a 64 byte volatile buffer that the MCU may fill before an internal EEPROM storage operation is initiated. Page mode timing approaches a 64:1 advantage over the time it takes to write individual bytes.

To invoke page mode, the MCU must write to EEPROM locations within a single page, with no more than 120µs between individual byte writes. A single page means that address lines A14 to A6 must remain constant. The MCU may write to the 64 locations on a page in any order, which is determined by address lines A5 to A0. As soon as 120µs have expired after the last page write, the internal EEPROM storage process begins and the MCU checks programming status. Status is checked the same way it is for byte writes, described above.

Note: Be aware that if the upper address bits (A14 to A6) change during page write operations, loss of data may occur. Ensure that all bytes for a given page have been successfully stored in the EEPROM before proceeding to the next page. Correct management of MCU interrupts during EEPROM page write operations is essential.

EEPROM Software Data Protect (SDP). The SDP feature is useful for protecting the contents of EEPROM from inadvertent write cycles that may occur during uncontrolled MCU bus conditions. These may happen if the application software gets lost or when VCC is not within normal operating range.

Instructions from the MCU are used to enable and disable SDP mode (see Table 8., page 20). Once enabled, the MCU must write an instruction sequence to EEPROM before writing data (much like writing to Flash memory). SDP mode can be used for both byte and page writes to EEPROM. The device will remain in SDP mode until the MCU issues a valid SDP disable instruction.

PSD devices are shipped with SDP mode disabled. However, within PSDsoft Express, SDP mode may be enabled as part of programming the device with a device programmer (PSDpro).

To enable SDP mode at run time, the MCU must write three specific data bytes at three specific memory locations, as shown in Figure 7., page 25. Any further writes to EEPROM when SDP is set will require this same sequence, followed by the byte(s) to write. The first SDP enable sequence can be followed directly by the byte(s) to be written.

To disable SDP mode, the MCU must write specific bytes to six specific locations, as shown in Figure 8., page 26.

The MCU must not be executing code from EEPROM when these instructions are invoked. The MCU must be operating from some other memory when enabling or disabling SDP mode.

The state of SDP mode is not changed by power on/off sequences (nonvolatile). When either the SDP enable or SDP disable instructions are issued from the MCU, the MCU must use the Toggle bit (status bit DQ6) or the Ready/$\overline{Busy}$ output pin to check programming status. The Ready/$\overline{Busy}$ output is driven low from the first write of AAh @ 555h until the completion of the internal storage sequence. Data Polling (status bit DQ7) is not supported when issuing the SDP enable or SDP disable commands.

Note: Using the SDP sequence (enabling, disabling, or writing data) is initiated when specific bytes are written to addresses on specific "pages" of EEPROM memory, with no more than 120µs between WRITES. The addresses 555h and AAAh are located on different pages of EEPROM. This is how the PSD distinguishes these instruction sequences from ordinary writes to EEPROM, which are expected to be within a single EEPROM page.

Writing the OTP Row

Writing to the OTP row (64 bytes) can only be done once per byte, and is enabled by an instruction. This instruction is composed of three specific WRITE operations of data bytes at three specific memory locations followed by the data to be stored in the OTP row (refer to Table 8., page 20).

During the WRITE operations, address bit A6 must be zero, while address bits A5-A0 define the OTP Row byte to be written while any EEPROM Sector Select signal (EESi) is active. Writing the OTP Row is allowed only when SDP mode is not enabled.

Figure 7. EEPROM SDP Enable Flowcharts