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

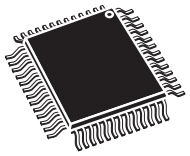
# DSM2180F3

## DSM (Digital Signal Processor System Memory) For Analog Devices ADSP-218X Family (5V Supply)

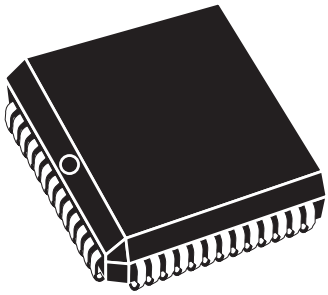
### FEATURES SUMMARY

- **Glueless Connection to DSP**
  - Easily add memory, logic, and I/O to DSP
- **128K Byte Flash Memory**
  - For Bootloading and/or Data Overlay Memory
  - Programmable Decoding and Paging Logic allows accessing Flash memory as Byte DMA (BDMA) and as External Data Overlay memory
  - Rapidly access Flash memory with BDMA for booting and loading internal DSP Overlay memory. Alternatively access the same Flash memory as External Data Overlay memory to efficiently write Flash memory with code updates and data, a byte at a time with no DMA setup overhead
  - Individual 16K Byte Flash memory sectors match size of DSP External Data Overlay window for efficient data management. Integrated page logic provides easy DSP access to all 128K Bytes.
  - DSM connects to lower byte of 16-bit DSP data bus. Byte-wide accesses to 8-bit BDMA space. Half-word accesses to 16-bit Data Memory Overlay and 16-bit I/O Mem space.
- **5V Devices (±10%)**
- **Up to 16 Multifunction I/O Pins**
  - Increase total DSP system I/O capability
  - I/O controlled by DSP software or PLD logic
  - 8mA I/O pin drive at 5 Vcc
- **General purpose PLD**
  - Over 3,000 Gates of PLD with 16 macro cells
  - Use for peripheral glue logic to keypads, control panel, displays, LCD, UART devices, etc.
  - Eliminate PLDs and external logic devices
  - Create state machines, chip selects, simple shifters and counters, clock dividers, delays
  - Simple PSDsoft Express™ software ...Free

**Figure 1. Packages**



PQFP52 (T)



PLCC52 (K)

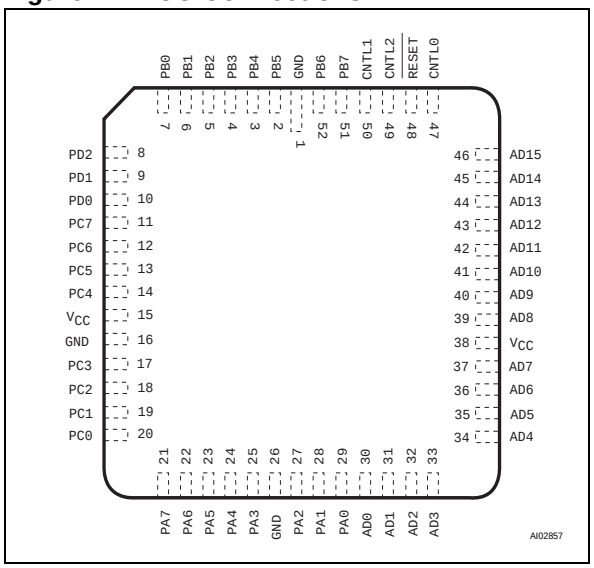
- **In-System Programming (ISP) with JTAG**
  - Program entire chip in 10-20 seconds with no involvement of the DSP
  - Eliminate sockets for pre-programmed memory and logic devices
  - Efficient manufacturing allows easy product testing and Just-In-Time inventory
  - Use low-cost FlashLINK™ cable with PC
- **Content Security**
  - Programmable Security Bit blocks access of device programmers and readers
- **Zero-Power Technology**
  - 75 µA standby at $V_{CC}$=5V
- **Small Packaging**
  - 52-pin PQFP or 52-pin PLCC
- **Memory Speed**
  - 90 ns

TABLE OF CONTENTS

## SUMMARY DESCRIPTION

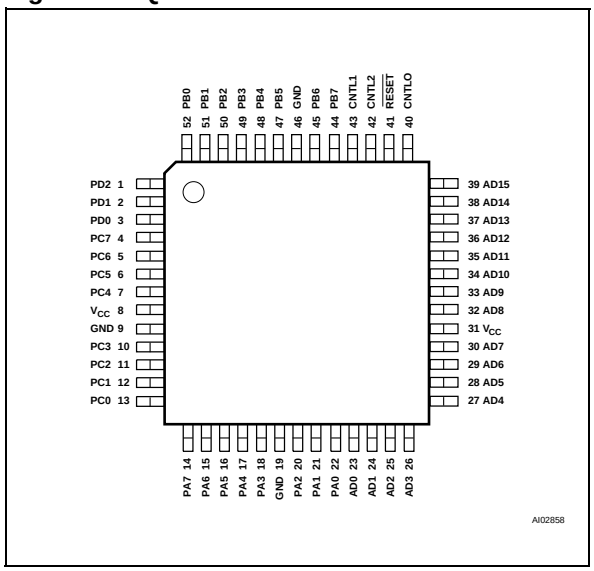
These are system memory devices for use with Digital Signal Processors from the popular Analog Devices ADSP-218X family. DSM means Digital signal processor System Memory. A DSM device brings in-system programmable Flash memory, programmable logic, and additional I/O to DSP systems. The result is a simple and flexible two-chip solution for DSP designs. DSM devices provide the flexibility of Flash memory and smart JTAG programming techniques for both manufacturing and the field. On-chip integrated memory decode logic and memory paging logic make it easy to add large amounts of external Flash memory to the ADSP-218X family for bootloading upon power-up and/or overlay memory. The DSP accesses this Flash memory using either its Byte DMA (BDMA) interface or as external data overlay memory (no DMA setup overhead).

**Figure 2. PLCC Connections**



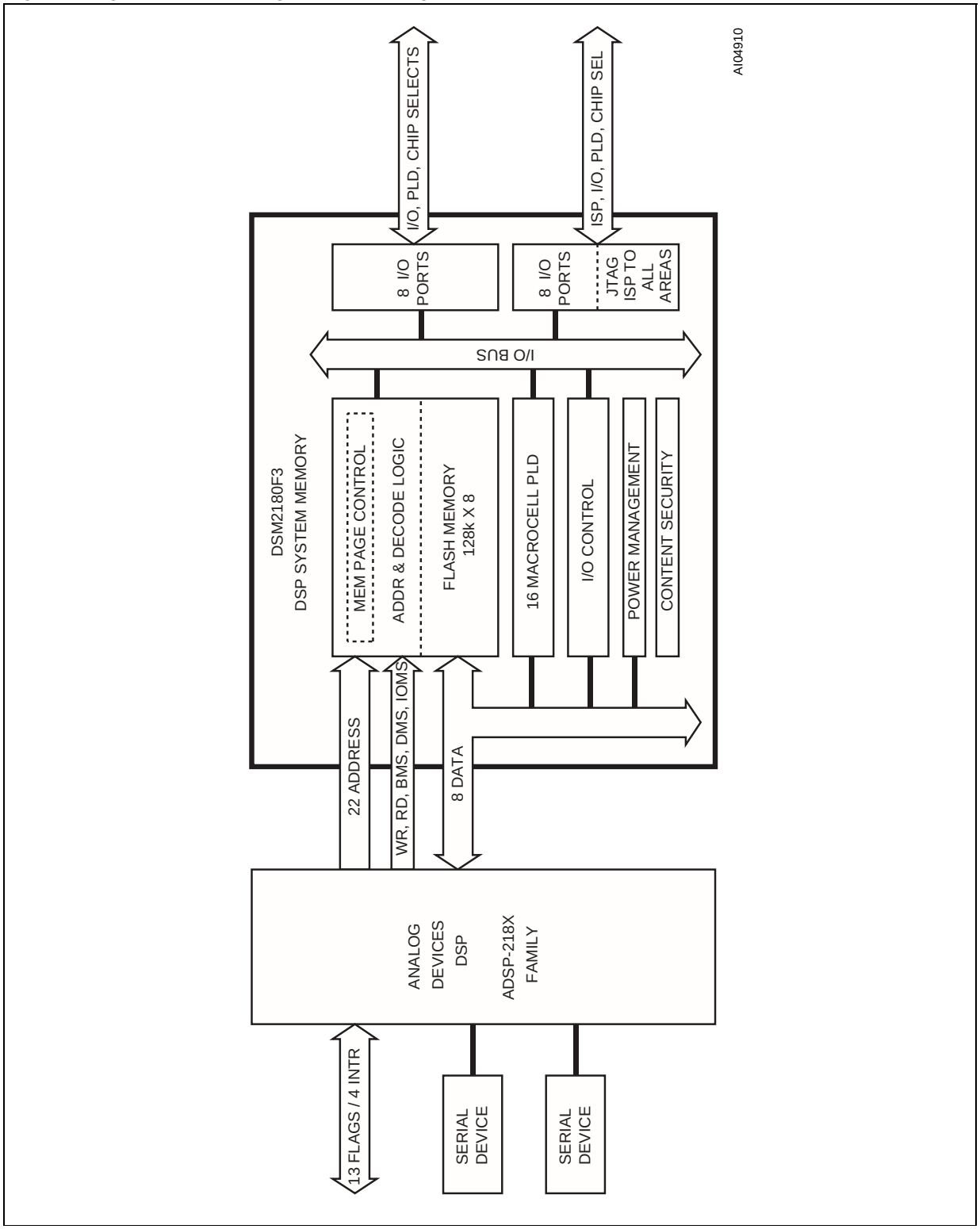

JTAG In-System Programming (ISP) reduces development time, simplifies manufacturing flow, and lowers the cost of field upgrades. The JTAG ISP interface eliminates the need for sockets and pre-programmed memory and logic devices. For manufacturing, end products may be assembled with a blank DSM device soldered to the circuit board and programmed at the end of the manufacturing line in 10 to 20 seconds with no involvement of the DSP. This allows efficient means to test product and manage inventory by rapidly programming test code, then application code as determined by inventory requirements (Just-In Time inventory). Additionally, JTAG ISP reduces development time by turning fast iterations of DSP code in the lab. Code updates in the field require no disassembly of product. The FlashLINK™ JTAG programming cable costs $59 USD and plugs into any PC or note-book parallel port.

**Figure 3. PQFP Connections**




In addition to ISP Flash memory, DSM devices add programmable logic (PLD) and up to 16 configurable I/O pins to the DSP system. The state of each I/O pin can be driven by DSP software or PLD logic. PLD and I/O configuration are programmable by JTAG ISP, just like the Flash memory. The PLD consists of more than 3000 gates and has 16 macro cell registers. Common uses for the PLD include chip selects for external devices (i.e. UART), state-machines, simple shifters and counters, keypad and control panel interfaces, clock dividers, handshake delay, muxes, etc. This eliminates the need for small external PLDs and logic devices. Configuration of PLD, I/O, and Flash memory mapping are easily entered in a point-and-click environment using the software development tool, PSDsoft Express™. This software is available at no charge from www.psdst.com.

**Figure 4. System Block Diagram, Two-Chip Solution**




The two-chip combination of a DSP and a DSM device is ideal for systems which have limitations on size, EMI levels, and power consumption. DSM memory and logic are “zero-power”, meaning they automatically go to standby between memory accesses or logic input changes, producing low active and standby current consumption, which is ideal for battery powered products.

A programmable security bit in the DSM protects its contents from unauthorized viewing and copying. When set, the security bit will block access of programming devices (JTAG or others) to the DSM Flash memory and PLD configuration. The only way to defeat the security bit is to erase the entire DSM device, after which the device is blank and may be used again. The DSP will always have access to Flash memory contents through the 8-bit data port even while the security bit is set.

**Table 1. DSM2180F3 DSP Memory System Devices**

| Part Number | ISP Flash Memory | Flash Partitioning | PLD | I/O Ports | $V_{CC}$ and I/O | Mem Speed |
|---|---|---|---|---|---|---|
| **DSM2180F3-90** | 128K Bytes | Eight 16K Byte Sectors | 16 macro cells | Up to 16 | 5V ±10% | 90 ns |

**Table 2. Compatible Analog Devices DSPs**

| DSP Part Numbers | Operating Voltage, $V_{CC}$ | I/O Capability |
|---|---|---|
| **ADSP- 2181, 2184, 2185, 2186** | 5.0 | 5.0V |

## ARCHITECTURAL OVERVIEW

Major functional blocks are shown in Figure 5.

### DSP Address/Data/Control Interface

These DSP signals attach directly to the DSM inputs for a glueless connection. An 8-bit data connection is formed and all 22 DSP address lines can be decoded while the DSP operates in full memory mode. DSP memory strobes; $\overline{BMS}$, $\overline{DMS}$, and $\overline{IOMS}$ are used for BDMA, data, & I/O access respectively (no program memory access, $\overline{PMS}$).

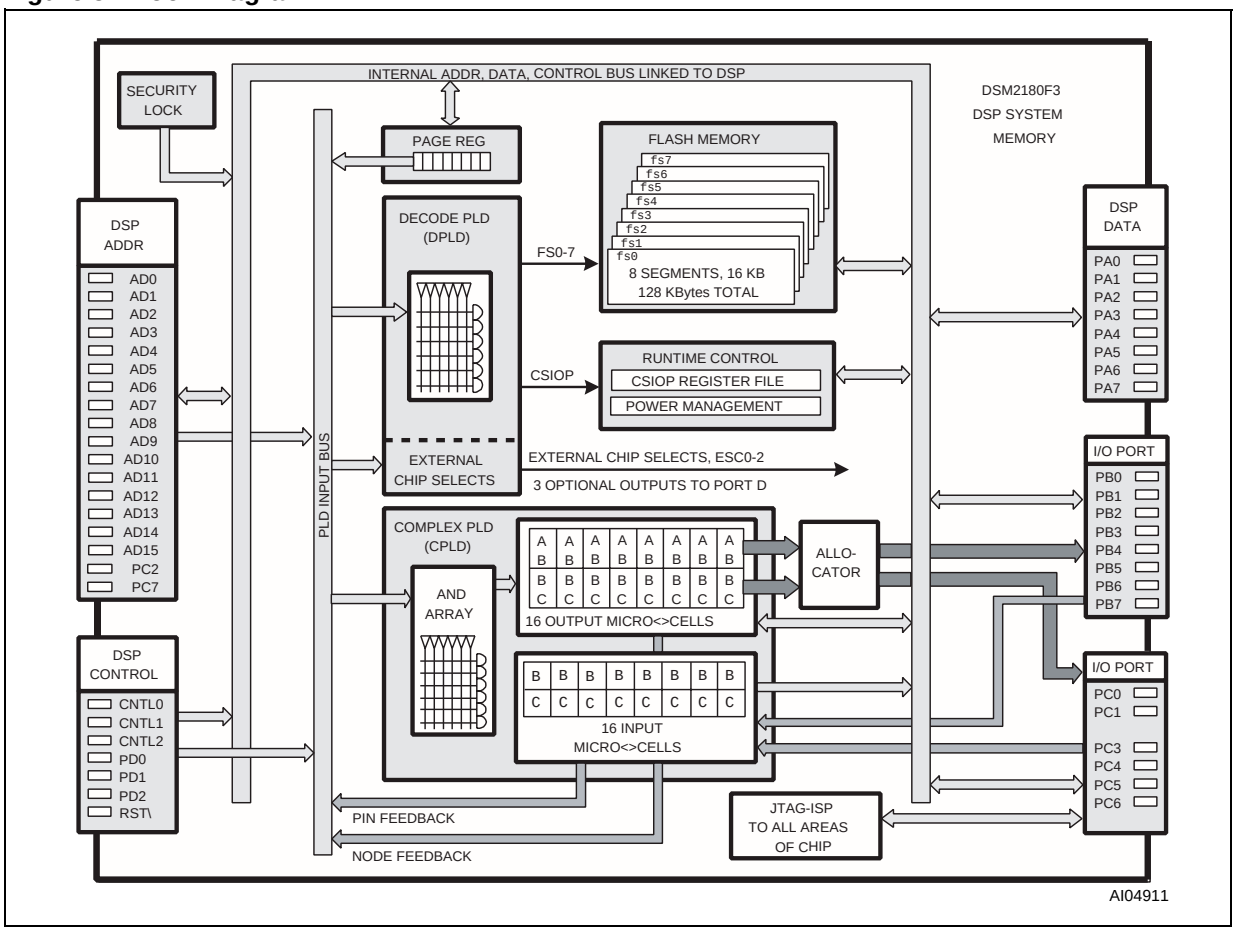
### Flash Memory

The 1 Mbit (128K x 8) Flash memory is divided into eight equally-sized 16K byte sectors that are individually selectable through the Decode PLD. Each Flash memory sector can be located at any address as defined by the user with PSDsoft Express. The flexibility of the Decode PLD and Page Register logic allow the DSP to access Flash memory as Byte DMA (BDMA) or as external data overlay memory across several memory pages. BDMA transfers are good for initial bootloading and for loading internal overlay memory at runtime, but BDMA is not efficient writing to Flash memory because Flash memory is unlocked, written, and status is checked one byte at a time, requiring an initialization of the BDMA channel for each and every byte transfer. The DSM device allows the DSP to alternatively access Flash memory as data overlay memory (using $\overline{DMS}$ instead of $\overline{BMS}$). Writing Flash memory this way is faster and requires simpler code. Note: During a DSP data access using the $\overline{DMS}$ strobe, only the upper byte of a 16-bit DSP data word is used.

DSM Flash memory sector size of 16K bytes matches the DSP external Data Memory Overlay window size of 16K locations (two 8K windows when DMOVLAY register is used, see Analog Devices ADSP-218X data sheets). This alignment provides convenient data management. Also, each 16K byte sector can be loaded with contents from different firmware or data files specified in PSDsoft Express™.

Miscellaneous: The DSP can erase Flash memory by individual sectors or the entire Flash memory array may be erased at one time. The Flash memory automatically goes to standby between DSP read or write accesses to conserve power. Maximum access times include sector decoding time. Maximum erase cycles is 100K and data retention is 15 years minimum. Flash memory, as well as the entire DSM device may be programmed with the JTAG ISP interface with no DSP involvement.

**Figure 5. Block Diagram**




**Programmable Logic (PLDs)**

The DSM family contains two PLDS that may optionally run in Turbo or Non-Turbo mode. PLDs operate faster (less propagation delay) while in Turbo mode but consume more power than Non-Turbo mode. Non-Turbo mode allows the PLDs to automatically go to standby when no inputs are change to conserve power. The Turbo mode setting is controlled at runtime by DSP software.

**Decode PLD (DPLD).** This is programmable logic used to select one of the eight individual Flash memory segments or the group of control registers within the DSM device. The DPLD can also optionally drive external chip select signals on Port D pins. DPLD input signals include: DSP address and control signals, Page Register outputs, DSM Port Pins, CPLD logic feedback.

**Complex PLD (CPLD).** This programmable logic is used to create both combinatorial and sequential general purpose logic. The CPLD contains 16 Output Macrocells (OMCs) and 16 Input Macrocells (IMCs). PSD macrocell registers are unique in that that have direct connection to the DSP data bus allowing them to be loaded and read directly by the DSP at runtime. This direct access is good for making small peripheral devices (shifters, counters, state machines, etc.) that are accessed directly by the DSP with little overhead. DPLD inputs include DSP address and control signals, Page Register outputs, DSM Port Pins, and CPLD feedback.

OMCs: The general structure of the CPLD is similar in nature to a 22V10 PLD device with the familiar sum-of-products (AND-OR) construct. True and compliment versions of 64 input signals are available to a large AND array. AND array outputs feed into a multiple product-term OR gate within each OMC (up to 10 product-terms for each OMC). Logic output of the OR gate can be passed on as combinatorial logic or combined with a flip-flop within in each OMC to realize sequential logic. OMCs can be used as a buried nodes with feedback to the AND array or OMC output can be routed to pins on Port B or PortC.

IMCs: Inputs from pins on Port B or Port C are routed to IMCs for conditioning (clocking or latching) as they enter the chip, which is good for sampling and debouncing inputs. Alternatively, IMCs can pass Port input signals directly to PLD inputs

without clocking or latching. The DSP may read the IMCs at any time.

**Runtime Control Registers**

A block of 256 bytes is decoded inside the DSM device as DSM control and status registers. 27 registers are used in the block of 256 locations to control the output state of I/O pins, to read I/O pins, to control power management, to read/write macrocells, and other functions at runtime. See Table 4 for description. The base address of these 256 locations is referred to in this data sheet as *csiop* (Chip Select I/O Port). Individual registers within this block are accessed with an offset from the base address. The DSP accesses *csiop* registers using I/O memory with the $\overline{IOMS}$ strobe. *csiop* registers are accessed as bytes, so only the lower half of a DSP I/O word is used during access.

**Memory Page Register**

This 8-bit register can be loaded and read by the DSP at runtime as one of the *csiop* registers. Its outputs feed directly into the PLDs. The page register is a powerful feature that allows the DSP to access all 128K Bytes of DSM Flash memory in 16K byte pages. This size matches the 16K location data overlay window the ADSP-218X family. Page register outputs may also be used as CPLD inputs for general use.

**I/O Ports**

The DSM has 19 individually configurable I/O pins distributed over the three ports (Ports B, C, and D). Each I/O pin can be individually configured for different functions such as standard MCU I/O ports or PLD I/O on a pin by pin basis. (MCU I/O means that for each pin, its output state can be controlled or its input value can be read by the DSP at runtime using the *csiop* registers like an MCU would do.)

Port C hosts the JTAG ISP signals. Since JTAG-ISP does not occur frequently during the life of a product, those Port C pins are under-utilized. In applications that need every I/O pin, JTAG signals can be multiplexed with general I/O signals to use them for I/O when not performing ISP. See section titled “Programming In-Circuit using JTAG ISP” on page 41 for muxing JTAG pins on Port C, and Application Note *AN1153*.

The static configuration of all Port pins is defined with the PSDsoft Express™ software development tool. The dynamic action of the Ports pins is controlled by DSP runtime software.

**JTAG ISP Port**

In-System Programming (ISP) can be performed through the JTAG signals on Port C. This serial interface allows programming of the entire DSM device or subsections (that is, only Flash memory but not the PLDs) without the participation of the DSP. A blank DSM device soldered to a circuit board can be completely programmed in 10 to 20 seconds. The basic JTAG signals; TMS, TCK, TDI, and TDO form the IEEE-1149.1 interface. The DSM device does not implement the IEEE-1149.1 Boundary Scan functions. The DSM uses the JTAG interface for ISP only. However, the DSM device can reside in a standard JTAG chain with other JTAG devices and it will remain in BYPASS mode while other devices perform Boundary Scan.

ISP programming time can be reduced as much as 30% by using two more signals on Port C, TSTAT and $\overline{TERR}$ in addition to TMS, TCK, TDI and TDO. The FlashLINK™ JTAG programming cable is available from STMicroelectronics for $59USD and PSDsoft Express software is available at no charge from www.psdst.com. That is all that is needed to program a DSM device using the parallel port on any PC or note-book. See section titled “Programming In-Circuit using JTAG ISP” on page 41.

**Power Management**

The DSM has bits in *csiop* control registers that are configured at run-time by the DSP to reduce power consumption of the CPLD. The Turbo bit in the PMMR0 register can be set to logic 1 and the CPLD will go to Non-Turbo mode, meaning it will latch its outputs and go to sleep until the next transition on its inputs. There is a slight penalty in PLD performance (longer propagation delay), but significant power savings are realized.

Additionally, bits in two *csiop* registers can be set by the DSP to selectively block signals from entering the CPLD which reduces power consumption. See section titled “Power Management” on page 39.

**Security and NVM Sector Protection**

A programmable security bit in the DSM protects its contents from unauthorized viewing and copying. When set, the security bit will block access of programming devices (JTAG or others) to the DSM Flash memory and PLD configuration. The only way to defeat the security bit is to erase the entire DSM device, after which the device is blank and may be used again.

Additionally, the contents of each individual Flash memory sector can be write protected (sector protection) by configuration with PSDsoft Express™. This is typically used to protect DSP boot code from being corrupted by inadvertent writes to Flash memory from the DSP.

**Pin Assignments**

Pin assignment are shown for the 52-pin PLCC package in Figure 2, and the 52-pin PQFP package in Figure 3.

**Table 3. Pin Description**

| Pin Name | Type | Description |
|---|---|---|
| ADIO0-15 | In | Sixteen address inputs from the DSP. |
| CNTL0 | In | Active low write strobe input ($\overline{WR}$) from the DSP |
| CNTL1 | In | Active low read strobe input ($\overline{RD}$) from the DSP. |
| CNTL2 | In | Active low Byte Memory Select ($\overline{BMS}$) signal from the DSP. |
| $\overline{Reset}$ | In | Active low reset input from system. Resets DSM I/O Ports, Page Register contents, and other DSM configuration registers. Must be logic Low at Power-up. |
| PA0-7 | I/O | Eight data bus signals connected to DSP pins D8 - D15. |
| PB0-7 | I/O | Eight configurable Port B signals with the following functions:<br>1. MCU I/O – DSP may write or read pins directly at runtime with csiop registers.<br>2. CPLD Output Macrocell (McellAB0-7 or McellBC0-7) outputs.<br>3. Inputs to the PLDs (Input Macrocells).<br>Note: Each of the four Port B signals PB0-PB3 may be configured at run-time as either standard CMOS or for high slew rate. Each of the four Port B signals PB3-PB7 may be configured at run-time as either standard CMOS or Open Drain Outputs. |
| PC0-7 | I/O | Eight configurable Port C signals with the following functions:<br>1. MCU I/O – DSP may write or read pins directly at runtime with csiop registers.<br>2. CPLD Output Macrocell (McellBC0-7) output.<br>3. Input to the PLDs (Input Macrocells).<br>4. Pins PC0, PC1, PC5, and PC6 can optionally form the JTAG IEEE-1149.1 ISP serial interface as signals TMS, TCK, TDI, and TDO respectively.<br>5. Pins PC3 and PC4 can optionally form the enhanced JTAG signals TSTAT and $\overline{TERR}$ respectively. Reduces ISP programming time by up to 30% when used in addition to the standard four JTAG signals: TDI, TDO, TMS, TCK.<br>6. Pin PC3 can optionally be configured as the Ready/Busy output to indicate Flash memory programming status during parallel programming. May be polled by DSP or used as DSP interrupt to indicate when Flash memory byte programming or erase operations are complete.<br>Note 1: Port C pin PC2 input (or any PLD input pin) can be connected to DSP D18 output which functions as DSP address A16 in DSP Full Memory Mode. See Figure 6.<br>Note 2: Port C pin PC7 input (or any PLD input pin) can be connected to DSP D19 output which functions as DSP address A17 in DSP Full Memory Mode. See Figure 6.<br>Note 3: When used as general I/O, each of the eight Port C signals may be configured at run-time as either standard CMOS or Open Drain Outputs.<br>Note 4: The JTAG ISP pins may be multiplexed with other I/O functions. |
| PD0-2 | I/O | Three configurable Port D signals with the following functions:<br>1. MCU I/O – DSP may write or read pins directly at runtime with csiop registers.<br>2. Input to the PLDs (no associated Input Macrocells, routes directly into PLDs).<br>3. CPLD output (External Chip Select). Does not consume Output Macrocells.<br>4. Pin PD1 can optionally be configured as CLKIN, a common clock input to PLD.<br>5. Pin PD2 can optionally be configured as $\overline{CSI}$, an active low Chip Select Input to select Flash memory. Flash memory is disabled to conserve more power when $\overline{CSI}$ is logic high. Can connect $\overline{CSI}$ to ADSP-218X PWDACK output signal.<br>Note 1: It is recommended to connect Port D pin PD0 input to DSP $\overline{IOMS}$ output which is the active low I/O Memory Select strobe. See Figure 6.<br>Note 2: It is recommended to connect Port D pin PD1 input to DSP $\overline{DMS}$ output which is the active low Data Memory Select strobe. See Figure 6.<br>Note 3: It is recommended to connect Port D pin PD2 input to DSP PWDACK output if the DSP Power Down mode is used. See Figure 6. |
| $V_{CC}$ | | Supply Voltage |
| GND | | Ground pins |

## TYPICAL CONNECTIONS

Figure 6 shows a typical connection scheme. Many connection possibilities exist since most DSM pins are multipurpose. The scheme illustrated is ideal for a design that needs fast JTAG ISP, Eight additional general I/O with PLD capability, access to Flash memory as Byte DMA or as Data Overlay memory, and the DSP uses Power Down mode. If your design needs more I/O, or Byte DMA access to Flash memory is all that is needed (no Data Overlay), or lowest power consumption is not an issue, then consider the following options.

**Port C JTAG:** Figure 6 shows all six JTAG signals in use full time (not multiplexed with I/0). Using six-pin JTAG can reduce ISP time by as much as 30% compared to four-pin JTAG. Alternatively, four-pin JTAG (TMS, TCK, TDI, TDO) can be used if more general I/O pins are needed and the few extra seconds of programming time is not crucial, freeing up pins PC3 and PC4. Other JTAG options include mutiplexing JTAG pins with general I/O (see “Programming In-Circuit using JTAG ISP” on page 41 and Application Note *AN1153*) or not using JTAG at all. If no JTAG is used, the DSM device has to be programmed on a conventional programmer before it is installed on the circuit board. Using no JTAG makes more I/O available.

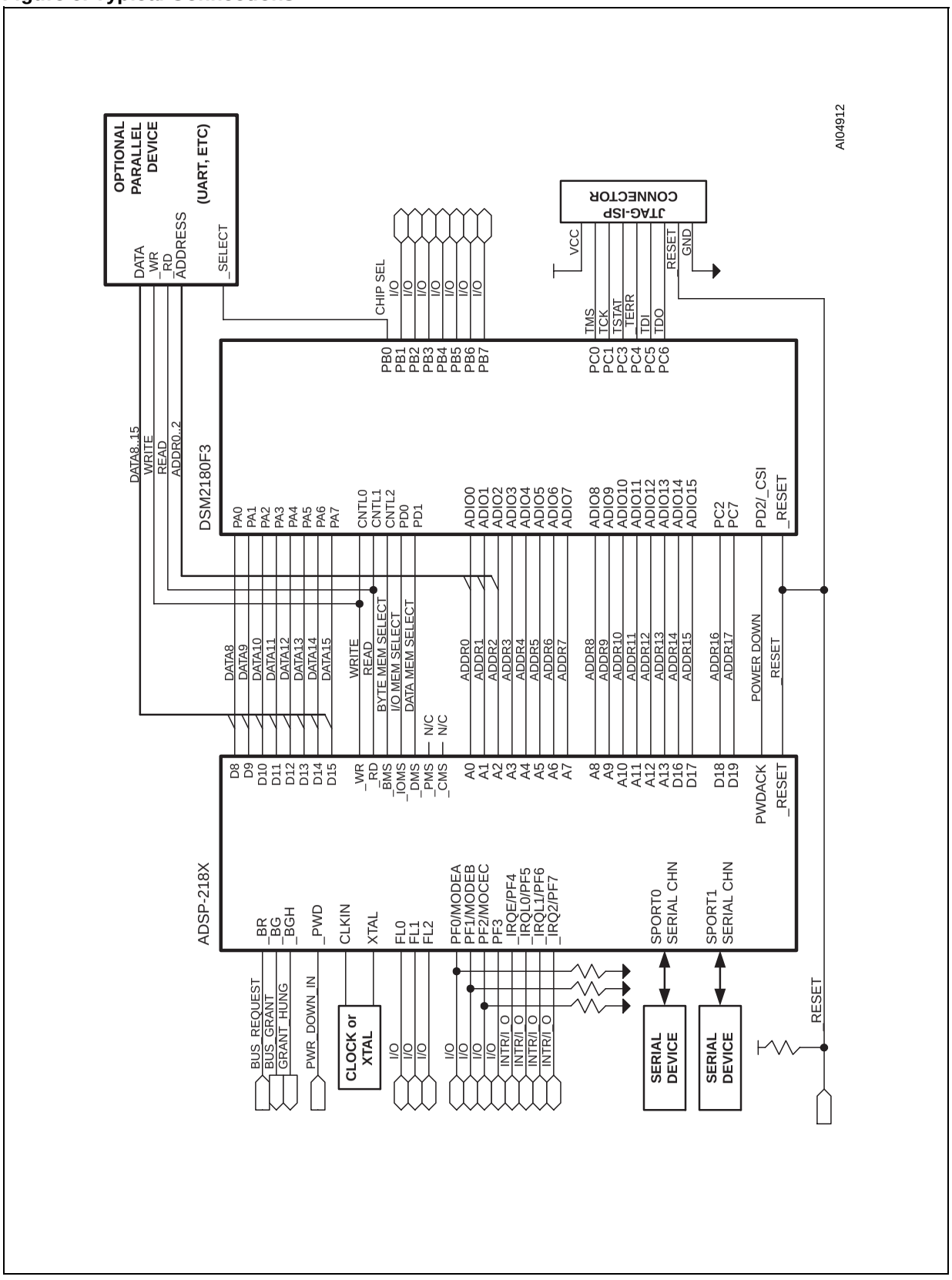
**Pin PD1.** If Flash memory will be accessed only using Byte DMA mode in your design, and no external Data Overlay memory accesses are used, then pin PD1 can be used for other purposes (MCUI/O, common CPLD clock input, external chip select, or PLD input)

**Pin PD2.** If the DSP will not use Power Down mode, then PD2 can be used for other purposes (MCUI/O, external chip select, PLD input)

**Pins PC2 and PC7.** In Figure 6, these two pins are used as dedicated address inputs connected to DSP address outputs. This will route DSP address signals A16 and A17 directly into the DPLD. Be aware that any free pin on Port B, Port C, or Port D may be used for DSP address inputs, it does not have to be pins PC2 and PC7.

**Pin PB0.** This pin is shown as a chip select for an external peripheral device such as a 16450 or 16550 UART. Equivalently, any free pin on Ports B, C, or D may be used for this.

**Figure 6. Typical Connections**



## MEMORY MAP

Figure 7 shows a typical system memory map. The nomenclature *fs0..fs7* are individual 16K Byte Flash memory segment designators. *csiop* designates the DSM control register block. The DSP runs in Full Memory Mode. Memory contents of the DSM device may lie in one or more of three different DSP address spaces; I/O space, Byte DMA space, and/or External Data Overlay Memory space. Since the DSM device is a byte-wide memory, it typically is not used in DSP Program Memory space ($\overline{\text{PMS}}$ active).

The designer may easily specify memory mapping in a point-and-click software environment using PSDsoft Express™. Since the memory mapping is implemented with the DPLD and the Page Register, many possibilities exist. Figure 7 shows a typical memory map with the following attributes:

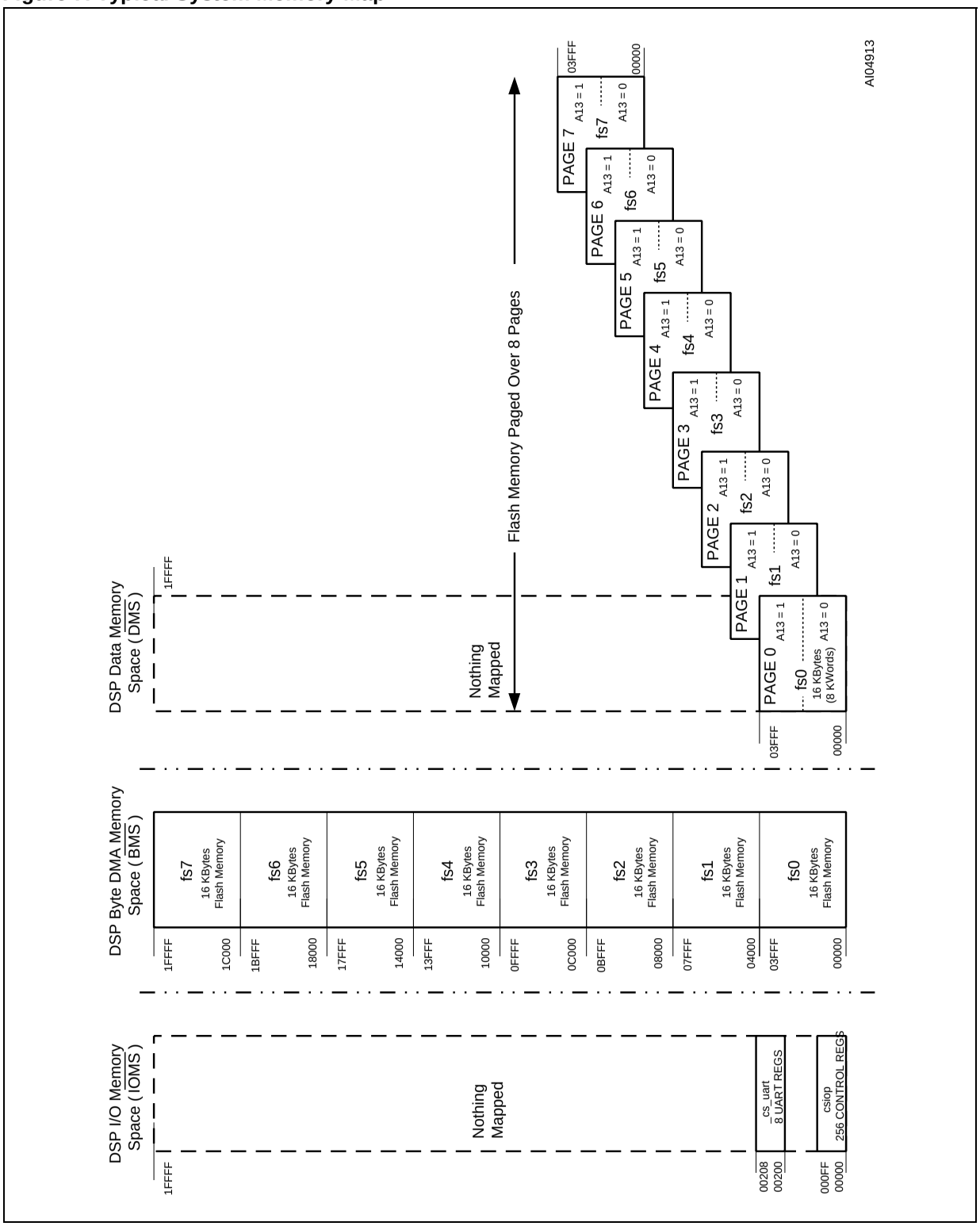
**I/O Address Space.** The 256 byte locations for DSM control registers (*csiop*) reside in DSP I/O address space, selected by the DSP $\overline{\text{IOMS}}$ signal. Since DSP I/O accesses are by 16 bits, not 8 bits, the upper byte of a 16-bit DSP I/O access must be ignored.

**Byte DMA Address Space.** The DSP may bootload or fetch overlay bytes from 128K Bytes of Flash memory using the DSP BDMA channel. The DSP may also write to Flash memory using the Byte DMA channel. DSM Flash memory is accessed in 128K continuous byte address locations through the BDMA channel and is selected whenever the DSP $\overline{\text{BMS}}$ signal is active.

Flash memory in the DSM device must be unlocked and written by the DSP one byte at a time, checking status after each write (typical Flash memory programming algorithm). A DMA channel is not optimum for this scenario since the channel must be initialized on each byte access. That is why the 128K Bytes of Flash memory also lie in DSP Data Overlay Memory space as described next.

**Data Overlay Memory Address Space.** All 128K Bytes of Flash memory also reside in DSP External Data Overlay Memory space, selected by $\overline{\text{DMS}}$, allowing more efficient byte writes to Flash memory. The DSP uses its external data overlay window of 8K locations to access external memory as data. The DSP doubles the size of this window to 16K locations by manipulating its A13 address line using its DMOVLAY register (See ADSP-218X data sheets for details). Since all 128K Bytes of Flash memory must be accessed through a window of only 16K locations, the DSP uses the Page Register inside the DSM device to page through 8 pages of 16K Bytes as shown in Figure 7. Since DSP Data accesses are by 16 bits, not 8 bits, the upper byte of a 16-bit DSP Data access must be ignored.

**Figure 7. Typical System Memory Map**



### SPECIFYING MEM MAP WITH PSDSOFT EXPRESS™

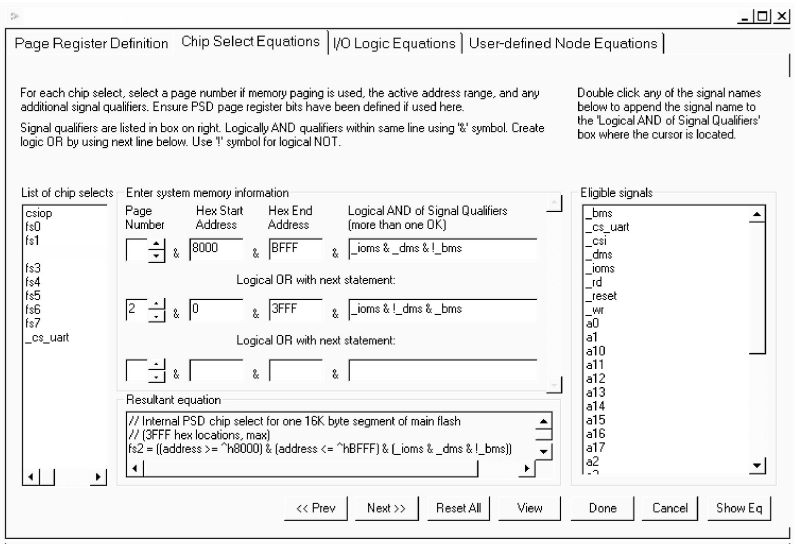
The memory map shown in Figure 7 can be easily specified with PSDsoft Express™ in a point-and-click environment. PSDsoft Express™ will generate Hardware Definition Language (HDL) statements of the ABEL language. Figure 8 shows the resulting equations generated by PSDsoft Express™.

**Figure 8. HDL Statements Generated from PSDsoft Express to Implement Memory Map**

```
csiop = ((address >= ^h0000) & (address <= ^h00FF) & (!_ioms & _dms & _bms));
fs0 = ((address >= ^h0000) & (address <= ^h3FFF) & (_ioms & _dms & !_bms))
# ((page == 0) & (address >= ^h0000) & (address <= ^h3FFF) & (_ioms & !_dms &
_bms));
fs1 = ((address >= ^h4000) & (address <= ^h7FFF) & (_ioms & _dms & !_bms))
# ((page == 1) & (address >= ^h0000) & (address <= ^h3FFF) & (_ioms & !_dms &
_bms));
fs2 = ((address >= ^h8000) & (address <= ^hBFFF) & (_ioms & _dms & !_bms))
# ((page == 2) & (address >= ^h0000) & (address <= ^h3FFF) & (_ioms & !_dms &
_bms));
fs3 = ((address >= ^hC000) & (address <= ^hFFFF) & (_ioms & _dms & !_bms))
# ((page == 3) & (address >= ^h0000) & (address <= ^h3FFF) & (_ioms & !_dms &
_bms));
fs4 = ((address >= ^h10000) & (address <= ^h13FFF) & (_ioms & _dms & !_bms))
# ((page == 4) & (address >= ^h0000) & (address <= ^h3FFF) & (_ioms & !_dms &
_bms));
fs5 = ((address >= ^h14000) & (address <= ^h17FFF) & (_ioms & _dms & !_bms))
# ((page == 5) & (address >= ^h0000) & (address <= ^h3FFF) & (_ioms & !_dms &
_bms));
fs6 = ((address >= ^h18000) & (address <= ^h1BFFF) & (_ioms & _dms & !_bms))
# ((page == 6) & (address >= ^h0000) & (address <= ^h3FFF) & (_ioms & !_dms &
_bms));
fs7 = ((address >= ^h1C000) & (address <= ^h1FFFF) & (_ioms & _dms & !_bms))
# ((page == 7) & (address >= ^h0000) & (address <= ^h3FFF) & (_ioms & !_dms &
_bms));
! _cs_uart = ((address >= ^h0200) & (address <= ^h0207) & (!_ioms & _dms & _bms));
```

Specifying these equations using PSDsoft Express™ is very simple. Figure 9 shows how to specify the equation for the 16K Byte Flash memory segment, *fs2*. Notice how *fs2* can reside in two different address spaces depending on the state of the control signals from the DSP ($\overline{IOMS}$, $\overline{DMS}$, or $\overline{BMS}$) and the memory page number coming from the DSM Page Register outputs. This specification process is repeated for all other Flash memory segments, the *csiop* register block, and any external chip select signals (UART, etc.).

**Figure 9. PSDsoft Express™ Memory Mapping**



## RUNTIME CONTROL REGISTER DEFINITION

There are up to 256 addresses decoded inside the DSM device for control and status information. 27 of these locations contain registers that the DSP can access at runtime. The base address of this block of 256 locations is referred to in this manual as *csiop* (Chip Select I/O Port). Table 4 lists the 27 registers and their offsets (in hex) from the *csiop* base address needed to access individual DSM control and status registers. The DSP will access these registers in I/O memory space using its $\overline{\text{IOMS}}$ strobe. These registers are accesses in bytes, so the DSP should ignore the upper byte of its 16-bit I/O access.

Note1: All *csiop* registers are cleared to logic 0 at reset.

Note2: Do not write to unused locations within the *csiop* block of 256 registers. They should remain logic zero.

**Table 4. *CSIOP* Registers and their Offsets (in hex)**

| Register Name | Port B | Port C | Port D | Other | Description |
|---|---|---|---|---|---|
| Data In | 01 | 10 | 11 | | MCUI/O input mode. Read to obtain current logic level of Port pins. No writes. |
| Data Out | 05 | 12 | 13 | | MCU I/O output mode. Write to set logic level on Port pins. Read to check status. |
| Direction | 07 | 14 | 15 | | MCU I/O mode. Configures Port pin as input or output. Write to set direction of Port pins. Logic 1 = out, Logic 0 = in. Read to check status. |
| Drive Select | 09 | 16 | 17 | | Write to configure Port pins as either standard CMOS or Open Drain on some pins, while selecting high slew rate on other pins. Read to check status. |
| Input Macrocells | 0B | 18 | | | Read to obtain state of IMCs. No writes. |
| Enable Out | 0D | 1A | 1B | | Read to obtain the status of the output enable logic on each I/O Port driver. No writes. |
| Output Macrocells AB | | | | 20 | Read to get logic state of output of OMC bank AB. Write to load registers of OMC bank AB. |
| Output Macrocells BC | | | | 21 | Read to get logic state of output of OMC bank BC. Write to load registers of OMC bank BC. |
| Mask Macrocells AB | | | | 22 | Write to set mask for loading OMCs in bank AB. A logic 1 in a bit position will block reads/writes of the corresponding OMC. A logic 0 will pass OMC value. Read to check status. |
| Mask Macrocells BC | | | | 23 | Write to set mask for loading OMCs in bank BC. A logic 1 in a bit position will block reads/writes of the corresponding OMC. A logic 0 will pass OMC value. Read to check status. |
| Flash Sector Protect | | | | C0 | Read to determine Flash Sector Protection Setting. No writes. |
| Security Bit | | | | C2 | Read to determine if DSM devices Security Bit is active. Logic 1 = device secured. No writes. |
| JTAG Enable | | | | C7 | Write to enable JTAG Pins (optional feature). Read to check status. |
| PMMR0 | | | | B0 | Power Management Register 0. Write and read. |
| PMMR2 | | | | B4 | Power Management Register 2. Write and read. |
| Page | | | | E0 | Memory Page Register. Write and read. |

## DETAILED OPERATION

Figure 5 shows major functional areas of the device:

- Flash Memory
- PLDs (DPLD, CPLD, Page Register)
- DSP Bus Interface (Address, Data, Control)
- I/O Ports
- Runtime Control Registers
- JTAG ISP Interface

The following describes these functions in more detail.

### Flash Memory

The Flash memory array is divided evenly into eight equal 16K byte sectors. Each sector is selected by the DPLD can be separately protected from program and erase cycles. This configuration is specified by using PSDsoft Express™.

**Memory Sector Select Signals.** The DPLD generates the Select signals for all the internal memory blocks (see Figure 14). Each of the eight sectors of the Flash memory has a Select signal (*FS0-FS7*) which contains up to three product terms. Having three product terms for each Select signal allows a given sector to be mapped into multiple areas of system memory.

**Ready/$\overline{\text{Busy}}$ (PC3).** This signal can be used to output the Ready/$\overline{\text{Busy}}$ status of the device. The output on Ready/$\overline{\text{Busy}}$ (PC3) is a 0 (Busy) when Flash memory is being written, *or* when Flash memory is being erased. The output is a 1 (Ready) when no Write or Erase cycle is in progress. This signal may be polled by the DSP or used as a DSP interrupt to indicate when an erase or program cycle is complete.

**Memory Operation.** The Flash memory is accessed through the DSP Address, Data, and Control Bus Interface. The DSP can access Flash memory as BDMA mode or as External Data Memory Overlay. But from the DSM perspective, it sees either type of access as a series of byte operations (reads and writes). If the DSP accesses the DSM in BDMA mode, then the DSP BDMA channel must be initialized and run for each byte (or block of bytes) read from Flash memory or it must initialize the DMA channel for each byte written to Flash memory. Alternatively, if the DSP accesses the DSM in External Data Memory Overlay mode, then the DSP must only ensure the PSD Page Register and the DSP DMOVLAY register contains the correct value, then it performs a normal data read or data write operation without the burden of initializing the BDMA channel for each operation (upper byte of 16-bit word is ignored).

DSPs and MCUs cannot write to Flash memory as it would an SRAM device. Flash memory must first be “unlocked” with a special sequence of byte write operations to invoke an internal algorithm, then a single data byte is written to the Flash memory array, then programming status is checked by a byte read operation or by checking the Ready/$\overline{\text{Busy}}$ pin (PC3). Table 5 lists all of the special instruction sequences to program (write) data to the Flash memory array, erase the array, and check for different types of status from the array. These instruction sequences are different combinations of individual byte write and byte read operations.

Once the Flash memory array is programmed (written) and then it is in “Read Array” mode, the DSP will read from Flash memory just as if would from any 8-bit ROM or SRAM device.

**Table 5. Instruction Sequences[1,2,3,4]**

| Instruction Sequence | Cycle 1 | Cycle 2 | Cycle 3 | Cycle 4 | Cycle 5 | Cycle 6 | Cycle 7 |
|---|---|---|---|---|---|---|---|
| Read Memory Contents[5] | Read byte from any valid Flash memory addr | | | | | | |
| Read Flash Identifier[6,7] | Write AAh to XX555h | Write 55h to XXAAAh | Write 90h to XX555h | Read identifier with addr lines A6,A1,A0 = 0,0,1 | | | |
| Read Memory Sector Protection Status[6,7,8] | Write AAh to XX555h | Write 55h to XXAAAh | Write 90h to XX555h | Read identifier with addr lines A6,A1,A0 = 0,1,0 | | | |
| Program a Flash Byte | Write AAh to XX555h | Write 55h to XXAAAh | Write A0h to XX555h | Write (program) data to addr | | | |
| Flash Bulk Erase[9] | Write AAh to XX555h | Write 55h to XXAAAh | Write 80h to XX555h | Write AAh to XX555h | Write 55h to XXAAAh | Write 10h to XX555h | |
| Flash Sector Erase[10] | Write AAh to XX555h | Write 55h to XXAAAh | Write 80h to XX555h | Write AAh to XX555h | Write 55h to XXAAAh | Write 30h to another Sector | Write 30h to another Sector |
| Suspend Sector Erase[11] | Write B0h to address that activates any of FS0 - FS7 | | | | | | |
| Resume Sector Erase[12] | Write 30h to addr that activates any of FS0 - FS7 | | | | | | |
| Reset Flash[6] | Write F0h to address that activates any of FS0 - FS7 | | | | | | |

Note: 1. All values are in hexadecimal, X = Don't Care
2. A desired internal Flash memory sector select signal (FS0 - FS7) must be active for each write or read cycle. Only one of FS0 - FS7 will be active at any given time depending on the address presented by the DSP and the memory mapping defined in PSDsoft Express. FS0 - FS7 are active high logic internally.
3. DSP addresses A17 through A12 are Don't Care during the instruction sequence decoding. Only address bits A11-A0 are used during Flash memory instruction sequence decoding bus cycles. The individual sector select signal (FS0 - FS7) which is active during the instruction sequences determines the complete address.
4. For write operations, addresses are latched on the falling edge of Write Strobe ($\overline{WR}$, CNTL0), Data is latched on the rising edge of Write Strobe ($\overline{WR}$, CNTL0)
5. No Unlock or Instruction cycles are required when the device is in the Read Array mode. Operation is like reading a ROM device.
6. The Reset Flash instruction is required to return to the normal Read Array mode if the Error Flag (DQ5) bit goes High, or after reading the Flash Identifier or after reading the Sector Protection Status.
7. The DSP cannot invoke this instruction sequence while executing code from the same Flash memory as that for which the instruction sequence is intended. The DSP must fetch, for example, the code from the DSP SRAM when reading the Flash memory Identifier or Sector Protection Status.
8. The data is 00h for an unprotected sector, and 01h for a protected sector. In the fourth cycle, the Sector Select is active, and (A1,A0)=(1,0)
9. Directing this command to any individual active Flash memory segment (FS0 - FS7) will invoke the bulk erase of all eight Flash memory sectors.
10. DSP writes command sequece to initial segment to be erased, then writes the byte 30h to additional sectors to be erased. The byte 30h must be addressed to one of the other Flash memory segments (FS0 - FS7) for each additional segment (write 30h to any address within a desired sector). No more than 80uS can elapse between subsequent additional sector erase commands.
11. The system may perform Read and Program cycles in non-erasing sectors, read the Flash ID or read the Sector Protect Status, when in the Suspend Sector Erase mode. The Suspend Sector Erase instruction sequence is valid only during a Sector Erase cycle.
12. The Resume Sector Erase instruction sequence is valid only during the Suspend Sector Erase mode.

### Instruction Sequences

An instruction sequence consists of a sequence of specific write or read operations. Each byte written to the device is received and sequentially decoded and not executed as a standard write operation to the memory array. The instruction sequence is executed when the correct number of bytes are properly received and the time between two consecutive bytes is shorter than the time-out period. Some instruction sequences are structured to include read operations after the initial write operations.

The instruction sequence must be followed exactly. Any invalid combination of instruction bytes or time-out between two consecutive bytes while addressing Flash memory resets the device logic into Read Array mode (Flash memory is read like a ROM device). The device supports the instruction sequences summarized in Table 5:

Flash memory:

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

These instruction sequences are detailed in Table 5. For efficient decoding of the instruction sequences, the first two bytes of an instruction sequence are the coded cycles and are followed by an instruction byte or confirmation byte. The coded cycles consist of writing the data AAh to address XX555h during the first cycle and data 55h to address XXAAAh during the second cycle. Address signals A17-A12 are Don't Care during the instruction sequence Write cycles. However, the appropriate internal Sector Select (*FS0-FS7*) must be selected internally (active, which is logic 1).

### Reading Flash Memory

Under typical conditions, the DSP may read the Flash memory using read operations just as it would a ROM or RAM device. Alternately, the DSP may use read operations to obtain status information about a Program or Erase cycle that is currently in progress. Lastly, the DSP may use instruction sequences to read special data from these memory blocks. The following sections describe these read instruction sequences.

**Read Memory Contents.** Flash memory is placed in the Read Array mode after Power-up, chip reset, or a Reset Flash memory instruction sequence (see Table 5). The DSP can read the memory contents of the Flash memory by using read operations any time the read operation is not part of an instruction sequence.

**Read Flash Identifier.** The Flash memory identifier is read with an instruction sequence composed of 4 operations: 3 specific write operations and a read operation (see Table 5). During the read operation, address bits A6, A1, and A0 must be 0,0,1, respectively, and the appropriate internal Sector Select (*FS0-FS7*) must be active. The identifier 0xE3.

**Read Memory Sector Protection Status.** The Flash memory Sector Protection Status is read with an instruction sequence composed of 4 operations: 3 specific write operations and a read operation (see Table 5). During the read operation, address bits A6, A1, and A0 must be 0,1,0, respectively, while internal Sector Select (FS0-FS7) designates the Flash memory sector whose protection has to be verified. The read operation produces 01h if the Flash memory sector is protected, or 00h if the sector is not protected.

The sector protection status can also be read by the DSP accessing the Flash memory Protection register in *csiop* space. See the section entitled "Flash Memory Sector Protect" for register definitions.

**Table 6. Status Bit Definition**

| Functional Block | FS0-FS7 | DQ7 | DQ6 | DQ5 | DQ4 | DQ3 | DQ2 | DQ1 | DQ0 |
|---|---|---|---|---|---|---|---|---|---|
| Flash Memory | Active (the desired segment is selected) | Data Polling | Toggle Flag | Error Flag | X | Erase Time-out | 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.

**Reading the Erase/Program Status Bits.** The device provides several status bits to be used by the DSP to confirm the completion of an Erase or Program cycle of Flash memory. These status bits minimize the time that the DSP spends performing these tasks and are defined in Table 6. The status bits can be read as many times as needed.

For Flash memory, the DSP can perform a read operation to obtain these status bits while an Erase or Program instruction sequence is being executed by the embedded algorithm. See the

section entitled “Programming Flash Memory”, on page 21, for details.

**Data Polling Flag (DQ7).** When erasing or programming in Flash memory, the Data Polling Flag (DQ7) bit outputs the complement of the bit being entered for programming/writing on the Data Polling Flag (DQ7) bit. Once the Program instruction sequence or the write operation is completed, the true logic value is read on the Data Polling Flag (DQ7) bit (in a read operation).

**Flash memory instruction features:**

- Data Polling is effective after the fourth Write pulse (for a Program instruction sequence) or after the sixth Write pulse (for an Erase instruction sequence). It must be performed at the address being programmed or at an address within the Flash memory sector being erased.
- During an Erase cycle, the Data Polling Flag (DQ7) bit outputs a 0. After completion of the cycle, the Data Polling Flag (DQ7) bit outputs the last bit programmed (it is a 1 after erasing).
- If the byte to be programmed is in a protected Flash memory sector, the instruction sequence is ignored.
- If all the Flash memory sectors to be erased are protected, the Data Polling Flag (DQ7) bit is reset to 0 for about 100 µs, and then returns to the previous addressed byte. No erasure is performed.

**Toggle Flag (DQ6).** The device offers another way for determining when the Flash memory Program cycle is completed. During the internal write operation and when the Sector Select FS0-FS7 is true, the Toggle Flag (DQ6) bit toggles 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 stops 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 cycle is finished when two successive reads yield the same output data. Flash memory specific features:

- The Toggle Flag (DQ6) bit is effective after the fourth write operation (for a Program instruction sequence) or after the sixth write operation (for an Erase instruction sequence).
- If the byte to be programmed belongs to a protected Flash memory sector, the instruction sequence is ignored.
- If all the Flash memory sectors selected for erasure are protected, the Toggle Flag (DQ6) bit toggles to 0 for about 100 µs and then returns to the previous addressed byte.

**Error Flag (DQ5).** During a normal Program or Erase cycle, the Error Flag (DQ5) bit is to 0. This bit is set to 1 when there is a failure during Flash memory Byte Program, Sector Erase, or Bulk Erase cycle.

In the case of Flash memory programming, the Error Flag (DQ5) bit indicates the attempt to program a Flash memory bit from the programmed state, 0, to the erased state, 1, which is not valid. The Error Flag (DQ5) bit may also indicate a Time-out condition while attempting to program a byte.

In case of an error in a Flash memory Sector Erase or Byte Program cycle, the Flash memory sector in which the error occurred or to which the programmed byte belongs must no longer be used. Other Flash memory sectors may still be used. The Error Flag (DQ5) bit is reset after a Reset Flash instruction sequence.

**Erase Time-out Flag (DQ3).** The Erase Time-out Flag (DQ3) bit reflects the time-out period allowed between two consecutive Sector Erase instruction sequence bytes. The Erase Time-out Flag (DQ3) bit is reset to 0 after a Sector Erase cycle for a time period of 100 µs + 20% unless an additional Sector Erase instruction sequence is decoded. After this time period, or when the additional Sector Erase instruction sequence is decoded, the Erase Time-out Flag (DQ3) bit is set to 1.

**Programming Flash Memory**

When a byte of Flash memory is programmed, individual bits are programmed to logic 0. You cannot program a bit in Flash memory to a logic 1 once it has been programmed to a logic 0. A bit must be erased to logic 1, and programmed to logic 0. That means Flash memory must be erased prior to being programmed. A byte of Flash memory is erased to all 1s (FFh). The DSP may erase the entire Flash memory array all at once or individual sector-by-sector, but not byte-by-byte. However, the DSP may program Flash memory byte-by-byte.

The Flash memory requires the DSP to send an instruction sequence to program a byte or to erase sectors (see Table 5).

Once the DSP issues a Flash memory Program or Erase instruction sequence, it must check for the status bits for completion. The embedded algorithms that are invoked inside the device provide several ways give status to the DSP. Status may be checked using any of three methods: Data Polling, Data Toggle, or Ready/Busy (pin PC3).

**Data Polling.** Polling on the Data Polling Flag (DQ7) bit is a method of checking whether a Pro-

gram or Erase cycle is in progress or has completed. Figure 10 shows the Data Polling algorithm.

When the DSP issues a Program instruction sequence, the embedded algorithm within the device begins. The DSP then reads the location of the byte to be programmed in Flash memory to check status. The Data Polling Flag (DQ7) bit of this location becomes the compliment of bit 7 of the original data byte to be programmed. The DSP continues to poll this location, comparing the Data Polling Flag (DQ7) bit and monitoring the Error Flag (DQ5) bit. When the Data Polling Flag (DQ7) bit matches bit7 of the original data, and the Error Flag (DQ5) bit remains 0, then the embedded algorithm is complete. If the Error Flag (DQ5) bit is 1, the DSP should test the Data Polling Flag (DQ7) bit again since the Data Polling Flag (DQ7) bit may have changed simultaneously with the Error Flag (DQ5) bit (see Figure 10).

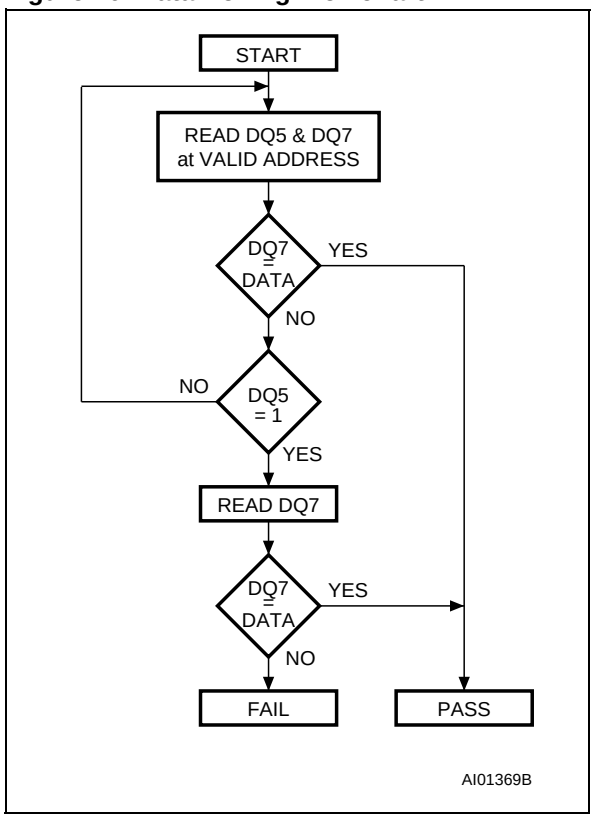
The Error Flag (DQ5) bit is set if either an internal time-out occurred while the embedded algorithm attempted to program the byte or if the DSP attempted to program a 1 to a bit that was not erased (not erased is logic 0).

It is suggested (as with all Flash memories) to read the location again after the embedded programming algorithm has completed, to compare the byte that was written to the Flash memory with the byte that was intended to be written.

When using the Data Polling method during an Erase cycle, Figure 10 still applies. However, the Data Polling Flag (DQ7) bit is 0 until the Erase cycle is complete. A 1 on the Error Flag (DQ5) bit indicates a time-out condition on the Erase cycle, a 0 indicates no error. The DSP can read any location within the sector being erased to get the Data Polling Flag (DQ7) bit and the Error Flag (DQ5) bit.

PSDsoft Express generates ANSI C code functions which implement these Data Polling algorithms.

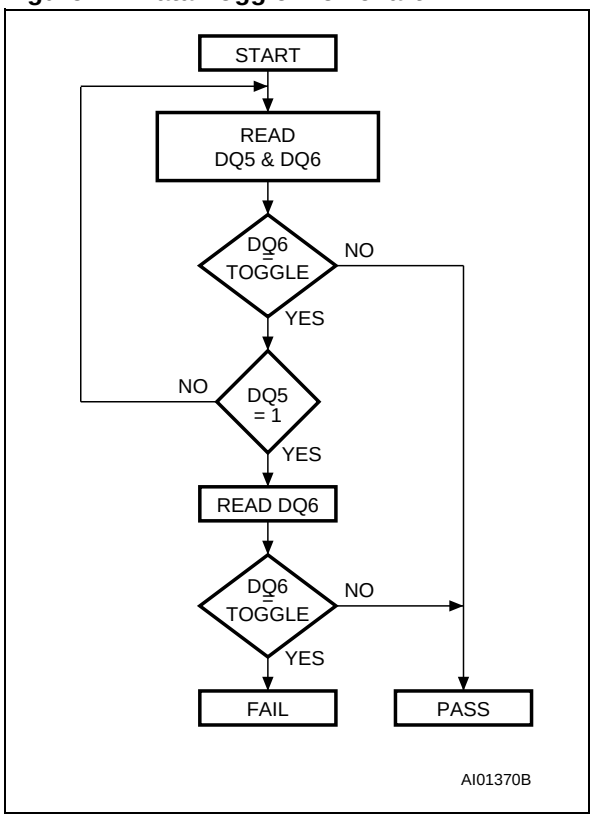
**Figure 10. Data Polling Flowchart**




**Data Toggle.** Checking the Toggle Flag (DQ6) bit is a method of determining whether a Program or Erase cycle is in progress or has completed. Figure 11 shows the Data Toggle algorithm.

When the DSP issues a Program instruction sequence, the embedded algorithm within the device begins. The DSP then reads the location of the byte to be programmed in Flash memory to check status. The Toggle Flag (DQ6) bit of this location toggles each time the DSP reads this location until the embedded algorithm is complete. The DSP continues to read this location, checking the Toggle Flag (DQ6) bit and monitoring the Error Flag (DQ5) bit. When the Toggle Flag (DQ6) bit stops toggling (two consecutive reads yield the same value), and the Error Flag (DQ5) bit remains 0, then the embedded algorithm is complete. If the Error Flag (DQ5) bit is 1, the DSP should test the Toggle Flag (DQ6) bit again, since the Toggle Flag (DQ6) bit may have changed simultaneously with the Error Flag (DQ5) bit (see Figure 11).

**Figure 11. Data Toggle Flowchart**




The Error Flag (DQ5) bit is set if either an internal time-out occurred while the embedded algorithm attempted to program the byte, or if the DSP attempted to program a 1 to a bit that was not erased (not erased is logic 0).

It is suggested (as with all Flash memories) to read the location again after the embedded programming algorithm has completed, to compare the byte that was written to Flash memory with the byte that was intended to be written.

When using the Data Toggle method after an Erase cycle, Figure 11 still applies. the Toggle Flag (DQ6) bit toggles until the Erase cycle is complete. A 1 on the Error Flag (DQ5) bit indicates a time-out condition on the Erase cycle, a 0 indicates no error. The DSP can read any location within the sector being erased to get the Toggle Flag (DQ6) bit and the Error Flag (DQ5) bit.

PSDsoft Express generates ANSI C code functions which implement these Data Toggling algorithms.

### Erasing Flash Memory

**Flash Bulk Erase.** The Flash Bulk Erase instruction sequence uses six write operations followed by a read operation of the status register, as described in Table 5. If any byte of the Bulk Erase instruction sequence is wrong, the Bulk Erase instruction sequence aborts and the device is reset to the Read Flash memory status. The Bulk Erase command may be addresses to any one individual valid Flash memory segment (*FS0-FS7*) and the entire array (all segments) will be erased.

During a Bulk Erase, the memory status may be checked by reading the Error Flag (DQ5) bit, the Toggle Flag (DQ6) bit, and the Data Polling Flag (DQ7) bit, as detailed in the section entitled “Programming Flash Memory”, on page 21. The Error Flag (DQ5) bit returns a 1 if there has been an Erase Failure (maximum number of Erase cycles have been executed).

It is not necessary to program the memory with 00h because the device automatically does this before erasing to 0FFh.

During execution of the Bulk Erase instruction sequence, the Flash memory does not accept any instruction sequences.

The address provided with the Flash Bulk Erase command sequence (Table 5) may select any one of the eight internal Flash memory Sector Select signals (FS0 - FS7). An erase of the entire Flash memory array will occur even though the command was sent to just one Flash memory sector.

**Flash Sector Erase.** The Sector Erase instruction sequence uses six write operations, as described in Table 5. Additional Flash Sector Erase codes and Flash memory sector addresses can be written subsequently to erase other Flash memory sectors in parallel, without further coded cycles, if the additional bytes are transmitted in a shorter time than the time-out period of about 100 µs. The input of a new Sector Erase code restarts the time-out period.

The status of the internal timer can be monitored through the level of the Erase Time-out Flag (DQ3) bit. If the Erase Time-out Flag (DQ3) bit is 0, the Sector Erase instruction sequence has been received and the time-out period is counting. If the Erase Time-out Flag (DQ3) bit is 1, the time-out period has expired and the device is busy erasing the Flash memory sector(s). Before and during Erase time-out, any instruction sequence other than Suspend Sector Erase and Resume Sector Erase instruction sequences abort the cycle that is currently in progress, and reset the device to Read Array mode. It is not necessary to program the Flash memory sector with 00h as the device does this automatically before erasing (byte=FFh).

During a Sector Erase, the memory status may be checked by reading the Error Flag (DQ5) bit, the Toggle Flag (DQ6) bit, and the Data Polling Flag (DQ7) bit, as detailed in the section entitled “Programming Flash Memory”, on page 21.

During execution of the Erase cycle, the Flash memory accepts only Reset and Suspend Sector Erase instruction sequences. Erasure of one

Flash memory sector may be suspended, in order to read data from another Flash memory sector, and then resumed.

The address provided with the initial Flash Sector Erase command sequence (Table 5) must select the first desired sector (FS0 - FS7) to erase. Subsequent sector erase commands that are appended on within the time-out period must be addressed to other desired segments (FS0 - FS7).

**Suspend Sector Erase.** When a Sector Erase cycle is in progress, the Suspend Sector Erase instruction sequence can be used to suspend the cycle by writing 0B0h to any address when an appropriate Sector Select (FS0-FS7) is selected (See Table 5). This allows reading of data from another Flash memory sector after the Erase cycle has been suspended. Suspend Sector Erase is accepted only during an Erase cycle and defaults to Read mode. A Suspend Sector Erase instruction sequence executed during an Erase time-out period, in addition to suspending the Erase cycle, terminates the time out period.

The Toggle Flag (DQ6) bit stops toggling when the device internal logic is suspended. The status of this bit must be monitored at an address within the Flash memory sector being erased. The Toggle Flag (DQ6) bit stops toggling between 0.1 µs and 15 µs after the Suspend Sector Erase instruction sequence has been executed. The device is then automatically set to Read mode.

If an Suspend Sector Erase instruction sequence was executed, the following rules apply:

– Attempting to read from a Flash memory sector that was being erased outputs invalid data.

– Reading from a Flash memory sector that was *not* being erased is valid.

– The Flash memory *cannot* be programmed, and only responds to Resume Sector Erase and Reset Flash instruction sequences (Read is an operation and is allowed).

– If a Reset Flash instruction sequence is received, data in the Flash memory sector that was being erased is invalid.

**Resume Sector Erase.** If a Suspend Sector Erase instruction sequence was previously executed, the erase cycle may be resumed with this instruction sequence. The Resume Sector Erase instruction sequence consists of writing 030h to any address while an appropriate Sector Select (FS0-FS7) is active. (See Table 5.)

**Flash Memory Sector Protect.**

Each Flash memory sector can be separately protected against Program and Erase cycles. Sector Protection provides additional data security because it disables all Program or Erase cycles. This mode can be activated through the JTAG Port or a Device Programmer. Sector protection can be selected for each sector using PSDsoft Express.

This automatically protects selected sectors when the device is programmed through the JTAG Port or a Device Programmer. Flash memory sectors can be unprotected to allow updating of their contents using the JTAG Port or a Device Programmer. The DSP can read (but cannot change) the sector protection bits.

Any attempt to program or erase a protected Flash memory sector is ignored by the device. The Verify operation results in a read of the protected data. This allows a guarantee of the retention of the Protection status.

The sector protection status can be read by the DSP through the Flash memory protection registers (in the *csiop* block) as defined in Table 7.

**Table 7. Sector Protection/Security Bit Definition – Flash Protection Register**

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| Sec7_Prot | Sec6_Prot | Sec5_Prot | Sec4_Prot | Sec3_Prot | Sec2_Prot | Sec1_Prot | Sec0_Prot |

Note: 1. Bit Definitions:
**Sec<i>_Prot** 1 = Flash memory sector <i> is write protected.
**Sec<i>_Prot** 0 = Flash memory sector <i> is not write protected.

**Table 8. Security Bit Definition**

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| Security_Bit | not used | not used | not used | not used | not used | not used | not used |

Note: 1. Bit Definitions:
1 = Security Bit in device has been set.

### DSM Security Bit

A programmable security bit in the DSM protects its contents from unauthorized viewing and copying. When set, the security bit will block access of programming devices (JTAG or others) to the DSM Flash memory and PLD configuration. The only way to defeat the security bit is to erase the entire DSM device, after which the device is blank and may be used again. The DSP will always have access to Flash memory contents through the 8-bit data port even while the security bit is set. The DSP can read the status of the security bit (but it cannot change it) by reading the Device Security register in the *csiop* block as defined in Table 8.

### Reset Flash

The Reset Flash instruction sequence resets the internal memory logic state machine and puts Flash memory into Read Array mode. It consists of one write cycle (see Table 5). It must be executed after:

– Reading the Flash Protection Status or Flash ID

– An Error condition has occurred (and the device has set the Error Flag (DQ5) bit to 1) during a Flash memory Program or Erase cycle.

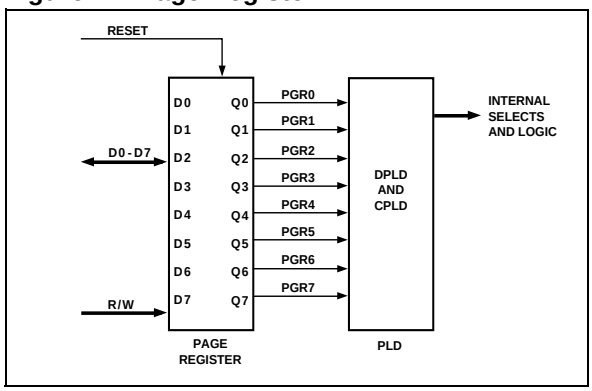
The Reset Flash instruction sequence puts the Flash memory back into normal Read Array mode. It may take the Flash memory up to a few milliseconds to complete the Reset cycle. The Reset Flash instruction sequence is ignored when it is issued during a Program or Bulk Erase cycle of the Flash memory. The Reset Flash instruction sequence aborts any on-going Sector Erase cycle, and returns the Flash memory to the normal Read Array mode within a few milliseconds.

### Page Register

The 8-bit Page Register increases the addressing capability of the DSP by a factor of up to 256. The contents of the register can also be read by the DSP. The outputs of the Page Register (PG0-PG7) are inputs to the DPLD decoder and can be included in the Sector Select (*FS0-FS7*) equations. See Figure 12.

If memory paging is not needed, or if not all 8 page register bits are needed for memory paging, then these bits may be used in the CPLD for general logic. The eight flip-flops in the register are connected to the internal data bus D0-D7. The DSP can write to or read from the Page Register. The Page Register can be accessed at address location *csiop* + E0h. Page Register outputs are cleared to logic 0 at reset.

**Figure 12. Page Register**




### PLDs

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

The PLDs have selectable levels of performance and power consumption.

The device contains two PLDs: the Decode PLD (DPLD), and the Complex PLD (CPLD), as shown in Figure 13.

**Table 9. DPLD and CPLD Inputs**

| Input Source | Input Name | Number of Signals |
|---|---|---|
| DSP Address Bus[1] | A15-A0 | 16 |
| DSP Control Signals[2] | CNTL2-CNTL0 | 3 |
| Reset | $\overline{RST}$ | 1 |
| PortB Input Macrocells | PB7-PB0 | 8 |
| PortC Input Macrocells | PC7-PC0 | 8 |
| Port D Inputs | PD2-PD0 | 3 |
| Page Register | PG7-PG0 | 8 |
| Macrocell AB Feedback | MCELLAB FB7-0 | 8 |
| Macrocell BC Feedback | MCELLBC FB7-0 | 8 |
| Flash memory Program Status Bit | Ready/$\overline{Busy}$ | 1 |

Note: 1. DSP address lines A16, A17, and others may enter the DSM device on any pin on ports B, C, or D. See Figure 6 for recommended connections.

2. Additional DSP control signals may enter the DMS device on any pin on Ports B, C, or D. See Figure 6 for recommended connections.