AN1428
APPLICATION NOTE

Interfacing the PSD813F5 with the TI TMS320C203 DSP

CONTENTS

- Purpose
- PSD813F1 Architecture
- Development Systems
- Programming the PSD813F In-Circuit using the JTAG Interface
- Interfacing the PSD813F5 with theTMS320C203
  - PSD813F5 Bus Interface
  - TMS320C203 Bus Interface Timing Calculation
  - TMS320C203 Memory Map
  - Interfacing to theTMS320C203 External Memory Bus
  - Define the TMS320C203 Interface in PSDsoft Express Define PSD and MCU Utility
  - Define the PSD813F1 DPLD Functions in PSDsoft Express Edit/Add Logic Statements
  - Accessing the PSD813F5 Internal Registers
  - External Boot Flash Memory Format
- In-Application re-Programming (IAP) using the PSD813F5
- TMS320C203 Boot Loader
- Summary
- Appendix

The Digital Signal Processing Marketplace is typically divided into two specific areas:

Function and Algorithm Specific ICs….are non-programmable DSPs integrated with other peripherals. They consist of modem chips, DVDs, MPEG and Video Decoders, etc.

General Purpose Programmable DSPs….are flexible DSPs that are used in a broad spectrum of products. They typically use a microcontroller for control, as well as additional I/O and programmable logic.

Most general purpose DSPs have internal 8-bit Boot Load routines imbedded in ROM which take advantage of slower, less expensive external Flash memory and EPROMs to store non-volatile program code to upload to fast internal SRAM at reset.

PURPOSE

Although the Flash PSD8XX family has become an ideal peripheral for 8-bit microcontrollers, many companies using the PSD in DSP-based products have shown that it makes an excellent peripheral for DSPs. The PSD8XX provides programmable logic and the required bus interfacing to implement a clean two-chip solution.

The PSD JTAG port allows In-System Programming (ISP) of a completely blank PSD8XX device soldered to the board with no involvement of the DSP, which is ideal for first time programming during manufacturing. The PSD8XX also offers In Application re-Programming (IAP), in which the DSP participates by executing UART download code from the small Flash memory in the PSD while writing new code into the large Flash memory in the PSD. This unique concurrent operation of PSD memories offers many IAP options. After IAP is complete, the DSP can copy the contents of the PSD main Flash memory into the fast DSP SRAM for full speed operation.

This Application Note addresses the ease of interfacing the PSD8XXF with the TMS320C203 DSP. Familiarity with the PSD8XXF is assumed. Please reference “PSD813F Data Sheet” for a detailed description of the device. The C203 DSP is optimized for telemetry and consumer applications, including POS terminals, PBX systems and Smart Card readers. The PSD8XXF family of Zero Power parts meets these criteria and enables the core DSP design to be done with two chips.

PSD813F1 ARCHITECTURE

The PSD8XX family is complemented by a lower-cost PSD9XX family. Figure 1 is a block diagram of the PSD8XX and PSD9XXF. Table 1 shows a comparison of the functional differences in the memory and CPLD options. On-chip features supply the key elements to implement a two-chip DSP System. Some devices have 32K bytes of byte-erasable EEPROM that may be used in place of external SRAM in some designs. Flash PSD features include:

- Programmable bus interface to DSPs that are capable of accessing external 8-bit boot code and/or program code.
- Programmable bus interface to DSPs with external 8-bit boot code and/or program code.
- 128-256 Kbytes of main Flash memory, divided into eight equal individually protected sectors.
- Separate 32 Kbytes EEPROM or Flash Boot memory divided into four equal blocks.
- Concurrent programming of the Flash or EEPROM/Boot Flash memories allows execution from one memory while reprogramming the other.
- 2 Kbytes or 8 Kbytes scratch-pad SRAM.
- Two Flash-based PLDs with 16 Output Micro⇔Cells and 24 Input Micro⇔Cells.
- 27 individually configurable I/O Port pins. Each may be defined as DSP I/Os, PLD I/Os, latched DSP address outputs or special function I/Os.
- 8-bit Page Register to expand the address space by a factor of 256.
- JTAG compliant serial port for true In-System Programming (ISP) of blank devices and reprogramming of devices in the factory or field.

Table 1. PSD8XXF and PSD9XX Product Matrix

Device	Flash Main Memory Kbit (8 Sectors)	Additional Memory for Boot and/or Data (4 Sectors)	SRAM Kbit	PLD
PSD813F1	1024	256 Kbit EEPROM	16	Sequential
PSD813F2	1024	256 Kbit Flash	16	Sequential
PSD813F3	1024	None	16	Sequential
PSD813F4	1024	256 Kbit Flash	None	Sequential
PSD813F5	1024	None	None	Sequential
PSD833F2	1024	256 Kbit Flash	64	Sequential
PSD834F2	2048	256 Kbit Flash	64	Sequential
PSD913F2	1024	256 Kbit Flash	16	Combinatorial
PSD934F2	2048	256 Kbit Flash	64	Combinatorial

The C203 has a basic on-chip Boot Loader, but only 544 words on-chip DARAM. Program code is downloaded from external Flash memory to fast external SRAM for execution after system reset. The low-cost PSD813F5 (no secondary boot memory) is selected for this design to take advantage of the TMS320C203 resident boot loader.

The following design parameters are assumed for using the PSD813F5 without the Flash Boot memory:

1. The initial firmware is programmed into the PSD Flash Memory through the JTAG interface on Port C of the PSD during manufacturing.

2. The firmware, containing the C203 Serial Port control code to download future code updates into the PSD813F5 Flash memory, is downloaded to the DSP DARAM during the Boot operation after Power On Reset is over.
3. The PSD813F5 Page Register is used to expand external Local memory beyond 64 K words.

DEVELOPMENT SYSTEMS

The PSD family is supported by PSDsoft Express, a software development tool that runs on Windows 95 and 98 and Windows NT. This tool has point and click features for DSP bus interface configuration, and uses an HDL (PSDabel) to define general programmable logic within the PLD. DSP firmware is imported and merged to create a single object file to program into the PSD. PSDsoft Express supports two device programmers directly (ST PSDpro, ST FlashLink). The generated object file is also compatible with third-party programmers. See web site for list *(www.st.com/psm).*

ST offers two low-cost device programmers:

PSDpro….plugs into a PC/laptop parallel port and replaces the ST MagicPro III.

FlashLink….is a low cost cable that plugs into a PC/laptop parallel port to support JTAG-ISP programming. FlashLink is controlled by PSDsoft and supports device chaining of multiple PSDs and devices from other manufacturers.

PROGRAMMING THE PSD813F IN-CIRCUIT USING THE JTAG INTERFACE

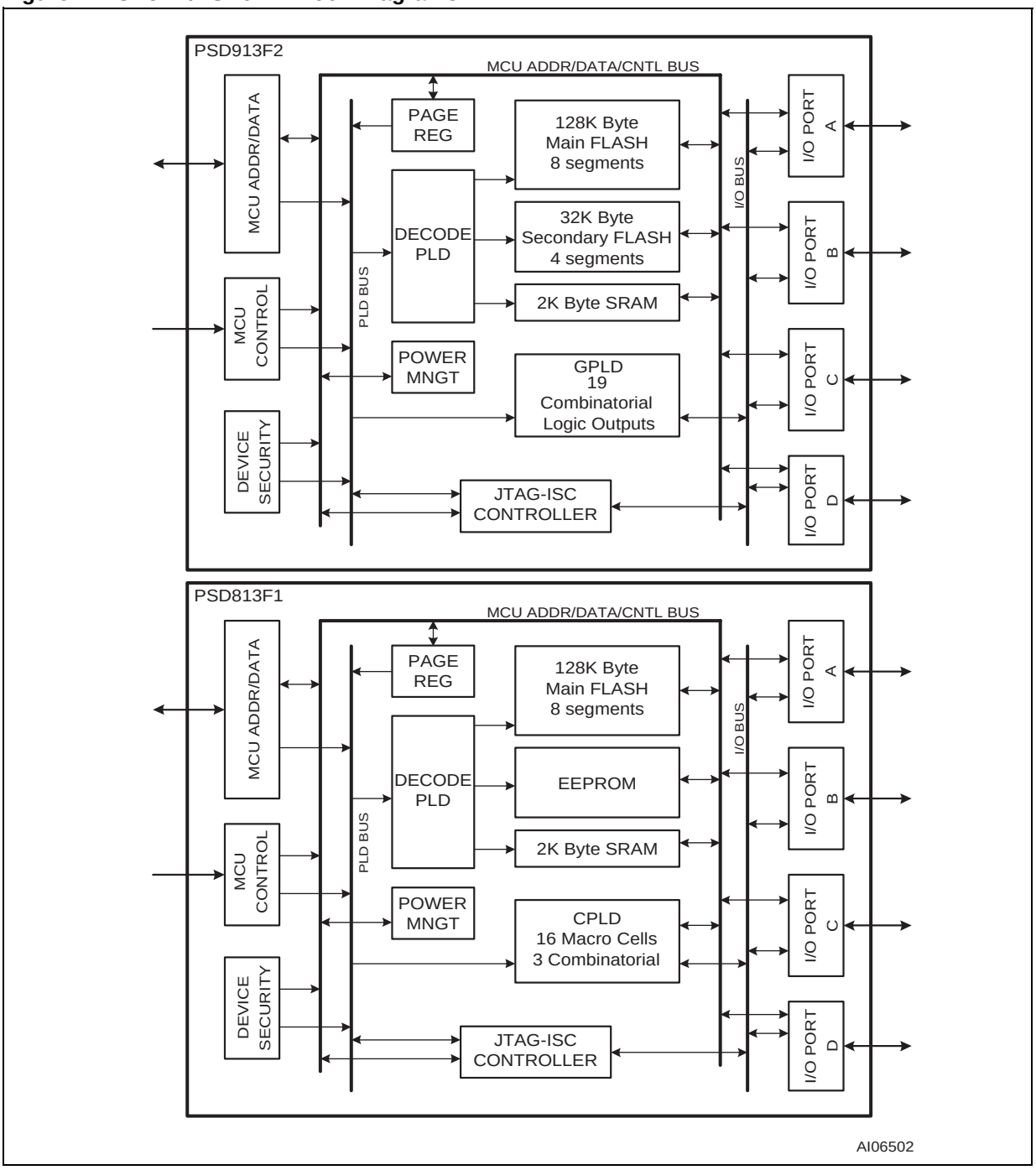
The ability to initially program a new system board with a blank Flash memory soldered directly to it has solved many manufacturing logistics problems – no sockets or individual labels are required; inventory of non-volatile program memory chips is reduced to one package; the PLD is programmed at the same time as the memory chip. One system board can be built and inventoried. Any options can be programmed into the Flash memory at board level testing.

Port C I/O lines are used to interface to the standard JTAG signals – TMS, TCK, TDI and TDO. TSTAT and TERR are optional JTAG-ISP extensions that can be monitored to speed up decrease the programming time of the PSD813F. The PSD configuration, PLD logic, Flash memory and optional Flash Boot/ EEPROM can be programmed simultaneously through this interface.

Port C also gives the option to multiplex its JTAG pins with the PSD813F general I/O lines. This option, if used, frees up the JTAG pins for I/O functions after JTAG programming is completed. This option is enabled by the following three lines of code in PSDabel, and its hardware implementation is illustrated in Application Note 054 “JTAG Information – PSD813F”:

```
jen             pin  11; “Port C pin pc7 is used as external JTAG multiplex enable
jtagsel         node;    “Selects JTAG port active using internal product term
jtagsel = !jen;          Switches Port C between JTAG and I/O
```

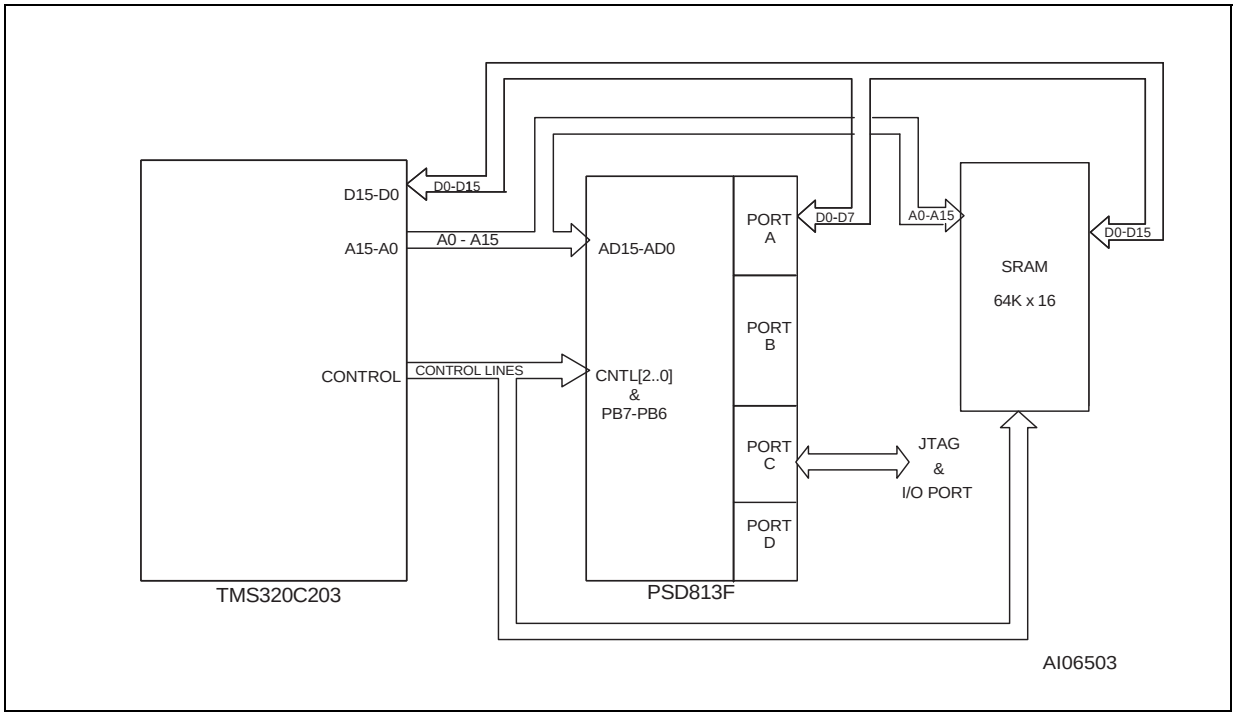
Figure 1. PSD8XX/PSD9XX Block Diagrams



INTERFACING THE PSD813F5 WITH THETMS320C203

Figure 2 is a Block diagram that shows the implementation of a three-chip System, including external SRAM to execute program code, using the PSD813F5 and the C203. All glue logic, Flash memory, bus interface logic, I/O, chip selects and PLDs are contained in one chip.

Figure 2. Block Diagram – Minimized DSP System



PSD813F5 Bus Interface

The PSD813F5 has a user-friendly programmable bus interface that is quickly configured to interface directly to most General Purpose DSPs with no “glue logic”. Table 2 lists the bus interface signals from the C203 used to access the Flash memory, PLD logic and I/O inside the PSD813F5. These bus signals are also used to access the EEPROM/Flash Boot memory and SRAM inside the PSD813F, if these options are desired.

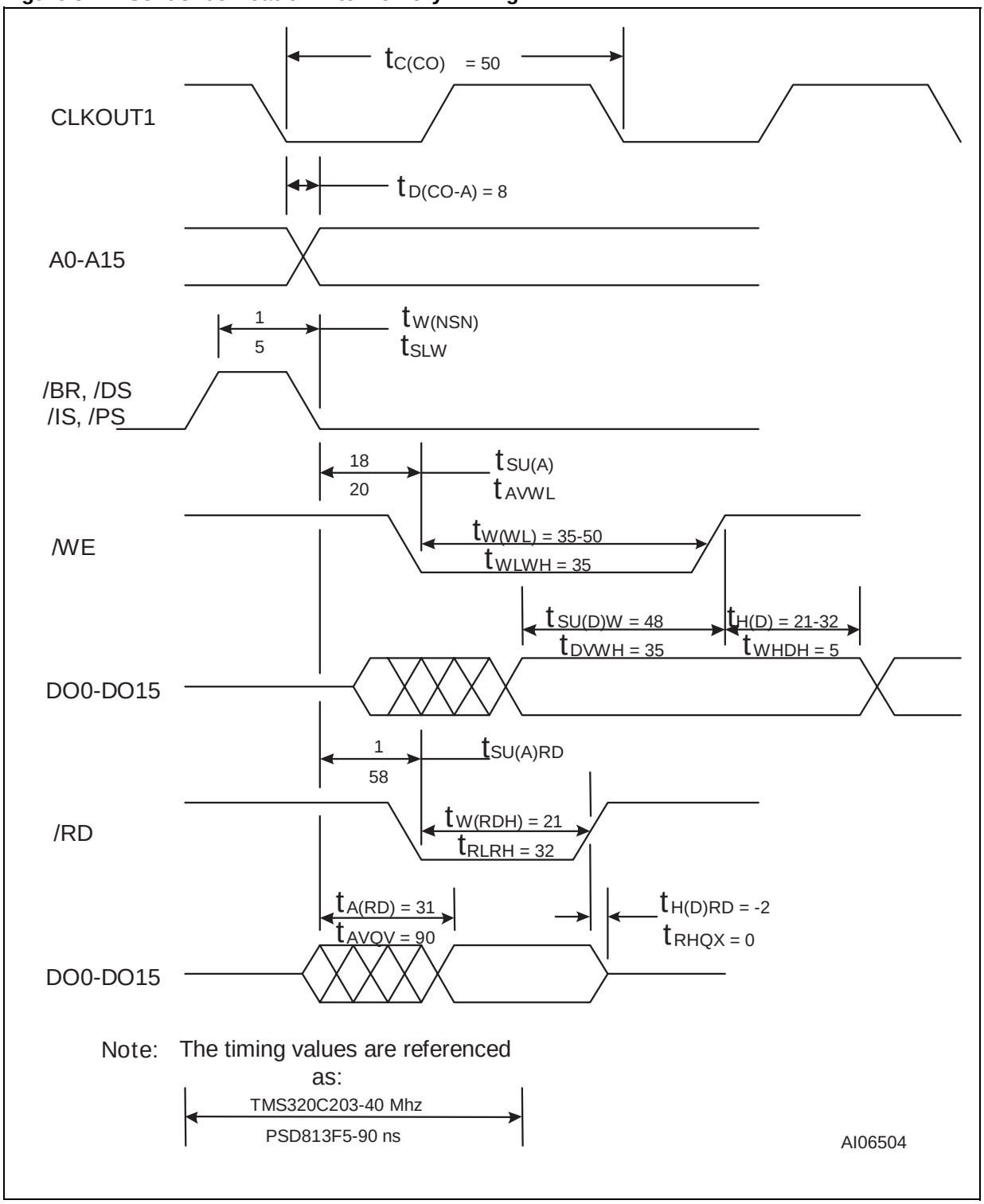
Table 2. TMS320C203 Bus Interface Pin Functions

TMS320C203 Pin Functions	PSD813F5 Pin Functions	Pin Description
A15 – A0	AD15 – AD0	External Address Bus addresses up to 64 Kwords of external memory or I/O space
D15 – D8 D7 – D0	NC PortA PA7 – PA0	16 bi-directional external data bus lines D7 – D0 connect to PortA (used as 8-bit data bus in non-multiplexed applications)
/BR	CNTL2	Bus Request pin. When active, /BR accesses global Boot Flash memory resident in the PSD813F5
/DS	PortB PB7	Data Memory Select. When active, /DS access external data memory – local or global
/PS	NC	Program Memory Select. When active /PS accesses External program memory in SRAM
/IS	PortB PB6	I/O Space Select. When active, /IS accesses External I/O space
/RD	CNTL1	Active low Read Select requests a read from External Program, Data or I/O space
/WE	CNTL0	Active low Write Select requests a write to External Program, Data or I/O space

TMS320C203 Bus Interface Timing Calculation

The TMS320C203 has a programmable wait-state generator that generates between 0 and 7 separate wait states for Program, Data and I/O accesses. To enable the TMS320C203 to interface to slow memory after reset, all wait state generators are reset to 7 wait-states. Figure 3 shows the read/write timing differences between the TMS320C203-40 MHz and PSD813F5-90ns. Three wait states would have to be programmed to access the Flash memory in the PSD813F5 during normal program execution.

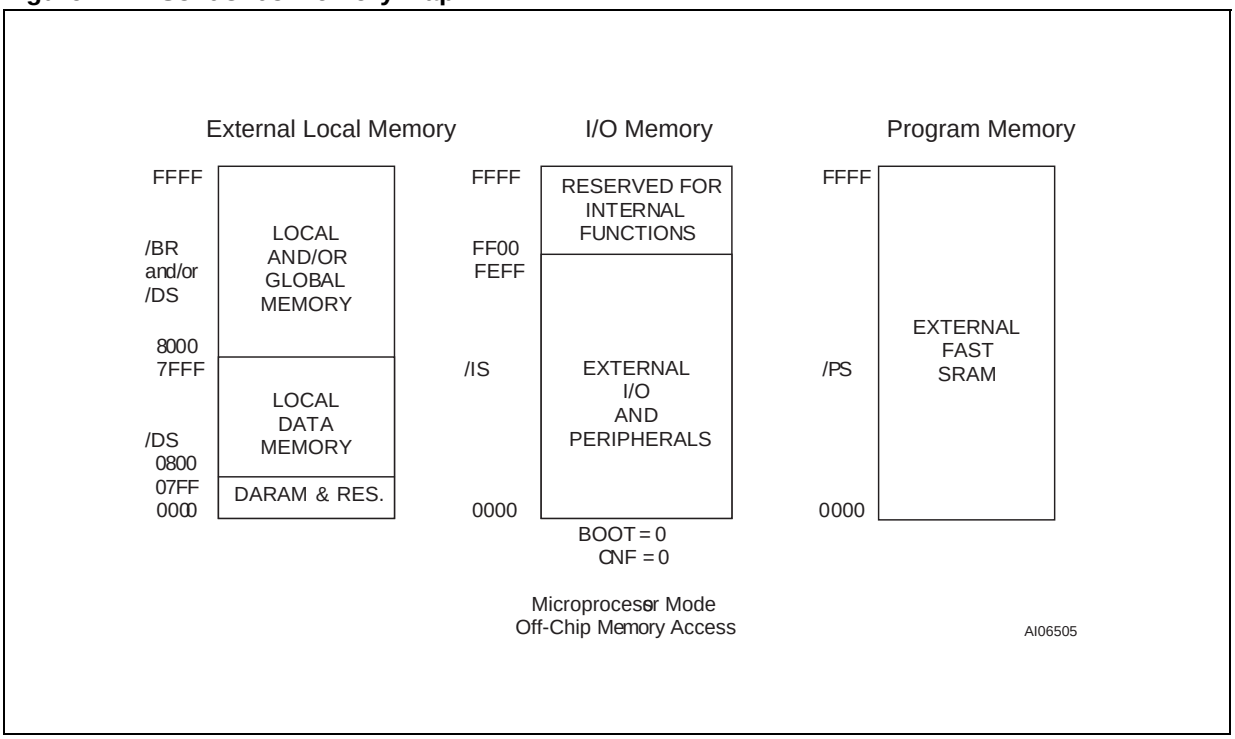
Figure 3. TMS320C203 Read / Write Memory Timing



TMS320C203 Memory Map

The TMS320C203 has four separate address spaces which can access a maximum of 224K x 16-bit external memory space – 64K Program, 64K Data, 64K I/O and 32K Global. In addition, there are 544 words of internal DARAM divided into three blocks – 256 words in data or program memory (B0), 256 words in data memory (B1), and 32 words in data memory (B2). The memory map is illustrated in Figure 4. Their respective select signals are also shown. At reset, the on-chip boot loader boots software from an 8-bit external EPROM/Flash, located in external global data memory, to 16-bit external fast SRAM located in program memory space.

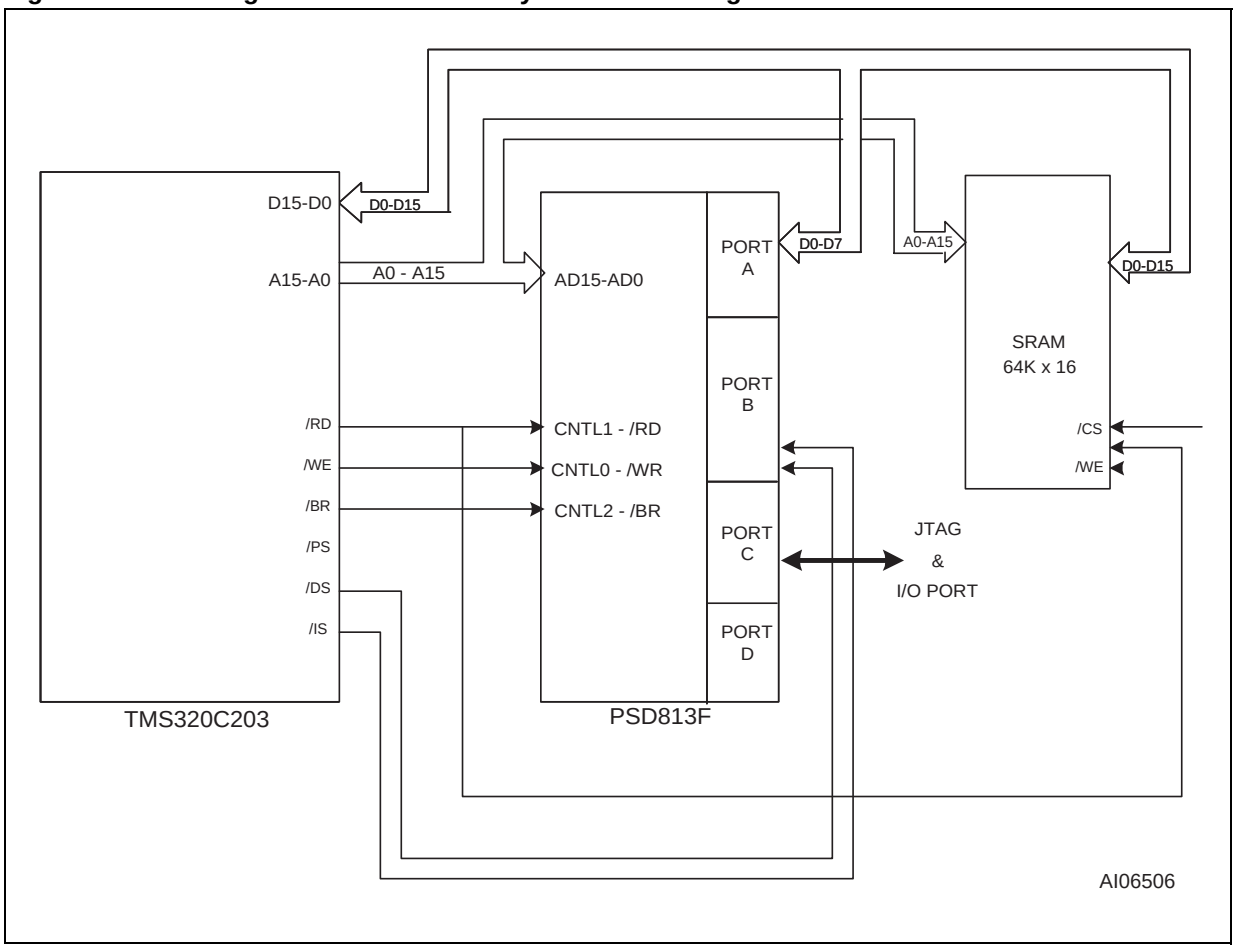
Figure 4. TMS320C203 Memory Map



Interfacing to theTMS320C203 External Memory Bus

The Block Diagram of Figure 5 shows the bus interface between the TMS320C203 and the PSD813F5. The TMS320C5203 has 16 address lines and 16 data lines. The PSD813F5 internal page register is used to configure the Flash memory into multiple pages. Paging will increase the program space that is normally accessible to the C203, if system requirements necessitate this.

Figure 5. Block Diagram – ADSP-21061 System Block Diagram



Define the TMS320C203 Interface in PSDsoft Express Define PSD and MCU Utility

Figure 6 is the MCU and PSD Selection screen from the PSDsoft Express development software. For more information on the PSDsoft Express, see the on-line User Manual on the ST website listed on the back page. Selecting the following appropriate signals in this screen quickly configures the bus configuration between the TMS320C203 and PSD813F5. /DS and /IS are connected to the PSD813F5 DPLD through Port B (pins PD6 and PD7) and included in the internal chip select equations generated in the Chip Select Equations screen.

- Type: Other
- Data Bus Width: 8-bit
- Address / Data Mode: Non-Mux
- Control Setting: /RD, /WE

Figure 6. PSDsoft Express Define PSD and MCU

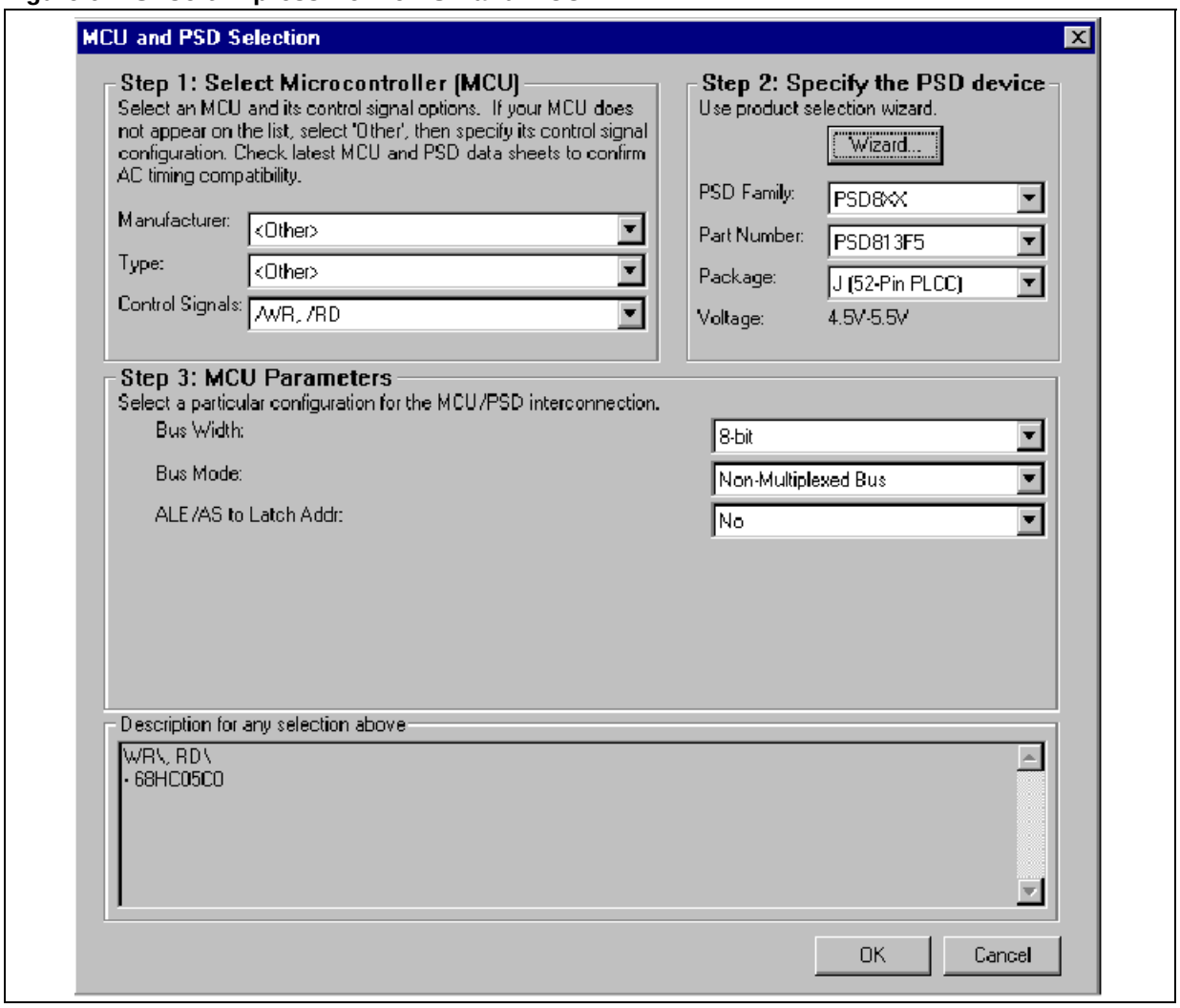
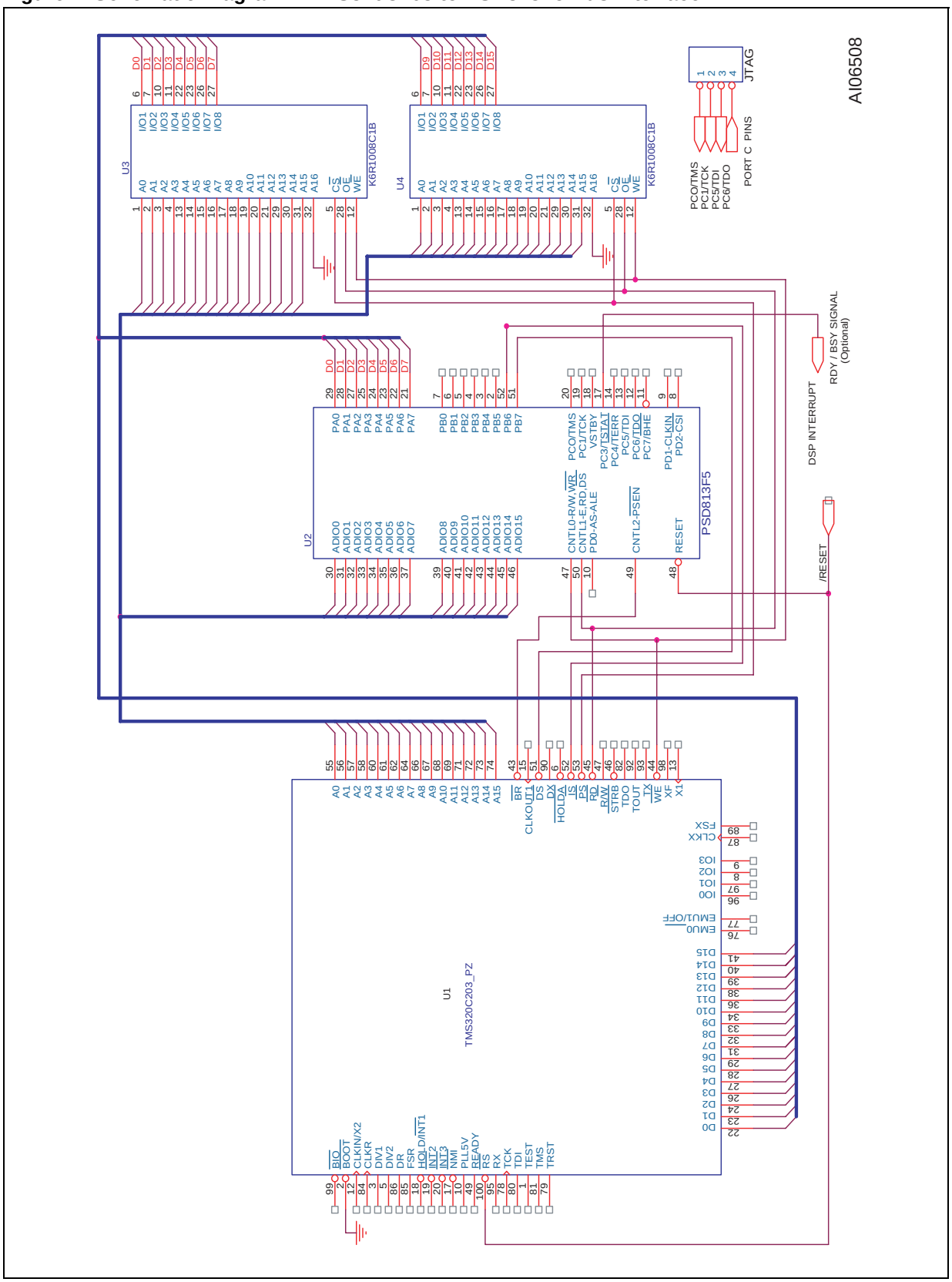


Figure 7 is the schematic diagram of the TMS320C203 / PSD813F5 bus interface. The 128K bytes of PSD813F5 Flash Program Memory reside in External Global memory space and are divided into four pages. It is downloaded to external SRAM for program execution during booting. Executing the program code from SRAM allows updated program code to be uploaded to the Flash Program memory through the DSP Serial Port for future download to the DARAM for program execution.

Figure 7. Schematic Diagram – TMS320C203 to PSD813F5 Bus Interface



Define the PSD813F1 DPLD Functions in PSDsoft Express Edit/Add Logic Statements

Figure 8 is the system memory map created for this A/N. The data, I/O and program addresses are defined in the PSDsoft Express Edit/Add Logic Statements screen and implemented in the internal PSD813F5 Decoding PLD (DPLD). Three bits of the PSD813F5 Page Register are used to extend the external data address range beyond the 64K limitation of the TMS320C203. The 128K bytes in the PSD813F5 are divided into four pages and reside in Global Memory space. Extending the DSP address range for program memory increases the value of the PSD813F5 for this application. Since paging is used, an area in data memory containing routines common to all data memory pages – memory-mapped registers, DARAM, I/O and external peripheral – must be accessible independent of which page the DSP is addressing. The following DSP control bits are set to implement the System Memory map in Figure 8:

1. CNF (bit 12 of Status Register 1) = 0….DARAM B0 is mapped to data space and is accessible at data addresses 0200h-02FFh. This increases the addressable external program memory by 512 words.
2. GREG (Global Memory Allocation Register) = XX80h….This value in the GREG sets the address range of the external Local Data Memory to 0000h-7FFFFh and the address range of the external Global memory to 800h-FFFFh.

Important: When the boot loader accesses global memory, both /BR and /DS are driven low. The system memory interface must be designed to prevent /DS from initiating erroneous accesses to data memory during boot loading.

Disabling /DS from accessing external Global Memory during boot loading is easily accomplished in the DPLD of the PSD813F5. Internal DPLD Flash Memory chip selects fs0–fs7 are used to select the 128K bytes of Flash memory in Global memory space (4 pages – 32K bytes per page). The following typical DPLD chip select equation illustrates how this is done:

```
fs0 =    ((address >= ^h8000) & (address <= ^hBFFF) & (page == 0) & /br
          # ((address >= ^h8000) & (address <= ^hBFFF) & (page == 0) & /ds & br;
```

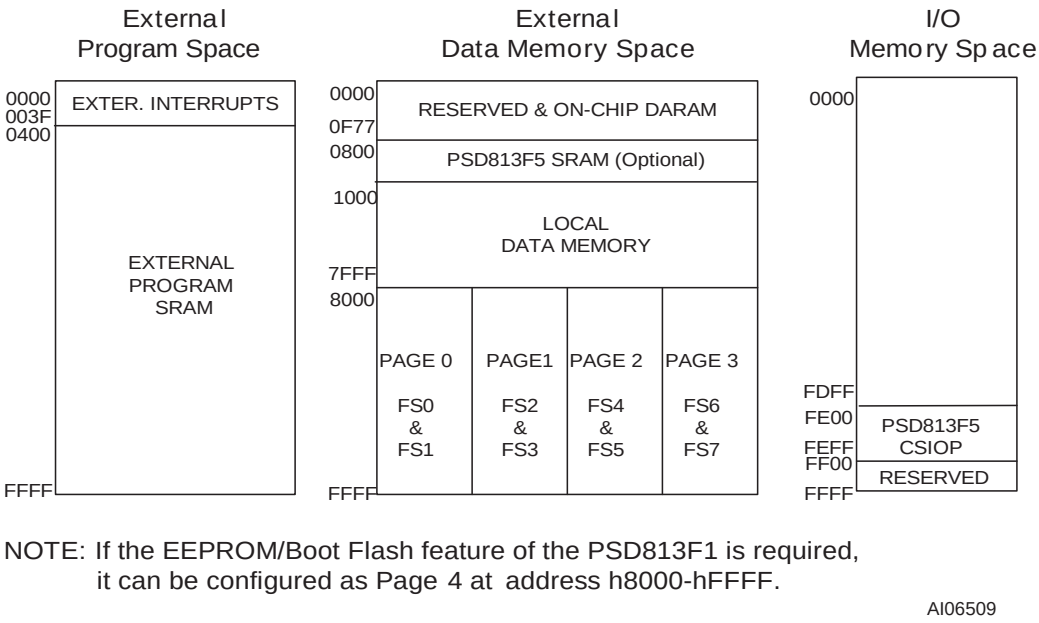
Address locations h8000-hBFFF are accessible as both Global and Local memory. When /BR is accessing Global memory for boot loading, the inverse of /BR disables /DS from accessing the shared memory space. When /DS is accessing Global memory, /BR is inactive.

Accessing the PSD813F5 Internal Registers

The bank of internal control registers in the PSD813F5 (csiop + hXX) are 8-bits wide. The DSP data bus is 16-bits wide and accesses external memory locations on a word boundary. When an internal register in the PSD813F5 is accessed, the DSP reads or writes a 16-bit word; only the lower 8 lines of the data bus are used to transfer data between the DSP and PSD.

When data is read from the internal registers, 16-bit data is read into the DSP accumulator. The high byte in the DSP accumulator is either ignored or masked out. When data is written to the internal registers, valid data must be located in the lower byte of the accumulator. The high byte is ignored by the PSD813F5.

Figure 8. TMS320C203 System Memory Map



IN-APPLICATION RE-PROGRAMMING (IAP) USING THE PSD813F5

The PSD813F5 (without secondary memory) was selected to reduce the system cost and take advantage of the DSP DARAM that can contain the program code for IAP of PSD Flash memory through the DSP Serial Port.

The asynchronous serial port on the TMS320C203 is used for field updates to the program code that resides in one or more sectors of the PSD813F5 Flash memory; these sectors are located in Bank 0 – Bank 3 in external Global Data memory space. The three product terms of each of the eight internal Flash memory Sector chip selects – fs0-fs7 – enable the control of the Flash Boot memory to switch between /BR for downloading Boot code, and /DS to field upgrade the Flash program memory as though it were data memory

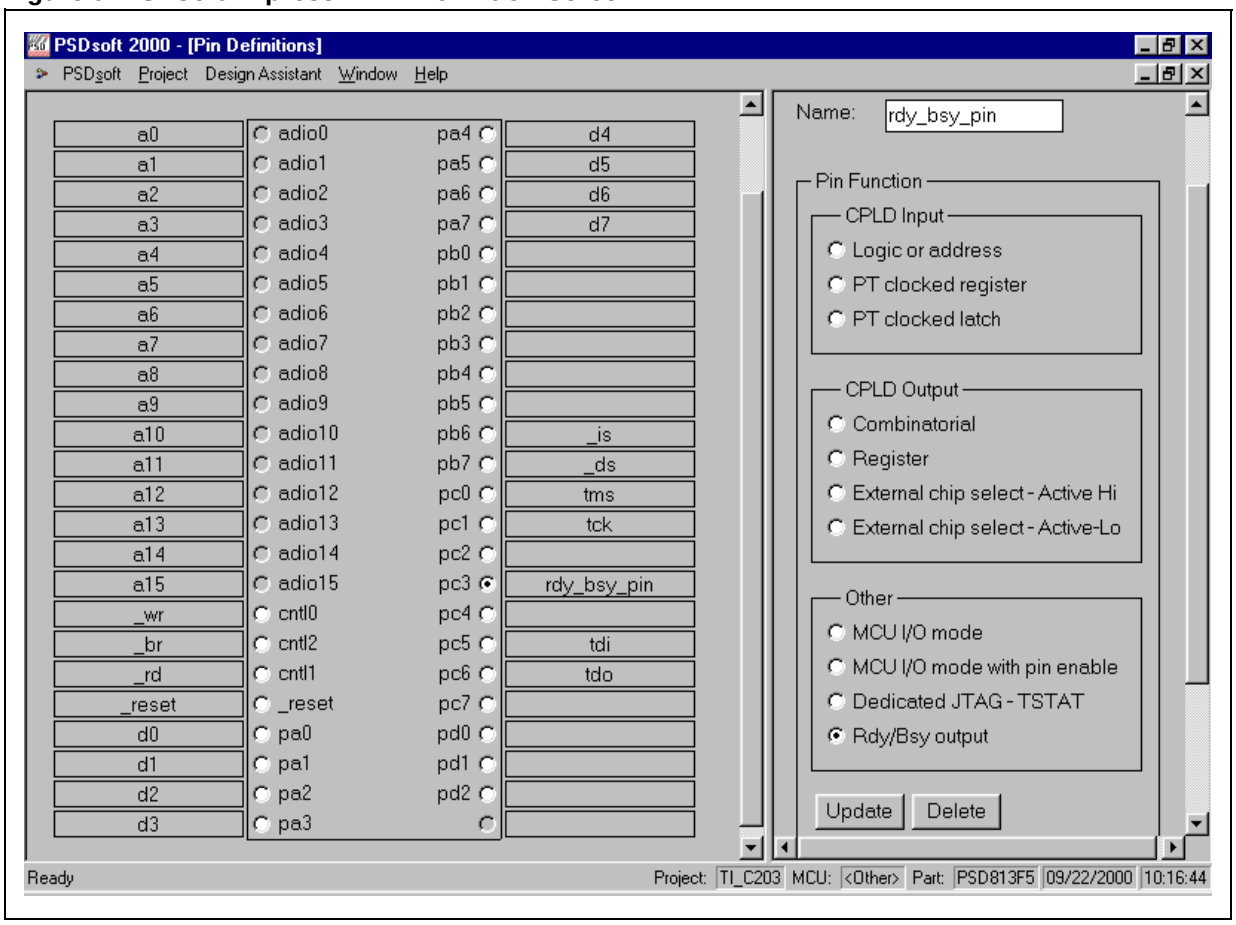
The TMS320C203 transmit and receive data buffers of the asynchronous serial port are 16-bit buffers. An 8-bit word is stored in the lower eight bits of the 16-bit receive and transmit buffers. When a boot program update is received by the serial port, the 8-bit data in the receive data buffer is written to the accumulator to store either in an assigned section of the DSP DARAM allocated as a buffer for the uploaded program code, or write to the Flash memory on a byte-by-byte basis.

The byte-by byte write sequence to the Flash memory can be speeded up dramatically by configuring the RDY/Busy polling bit to Port C (pin PC3) and using it as an interrupt input to the DSP. Once a byte write command is issued, the time required to program the byte can now be executed in background mode. The DSP can be performing other tasks until the RDY/Busy pin signals that the byte has been successfully programmed and generates an interrupt.

The RDY/Busy bit is hardware configured as an output interrupt pin as shown in Figure 9 with the configuration sequence as follows:

1. Select pc3 from the PSDsoft Express Pin Definition screen.
2. Select “Rdy/Bsy output” under “Other” block.

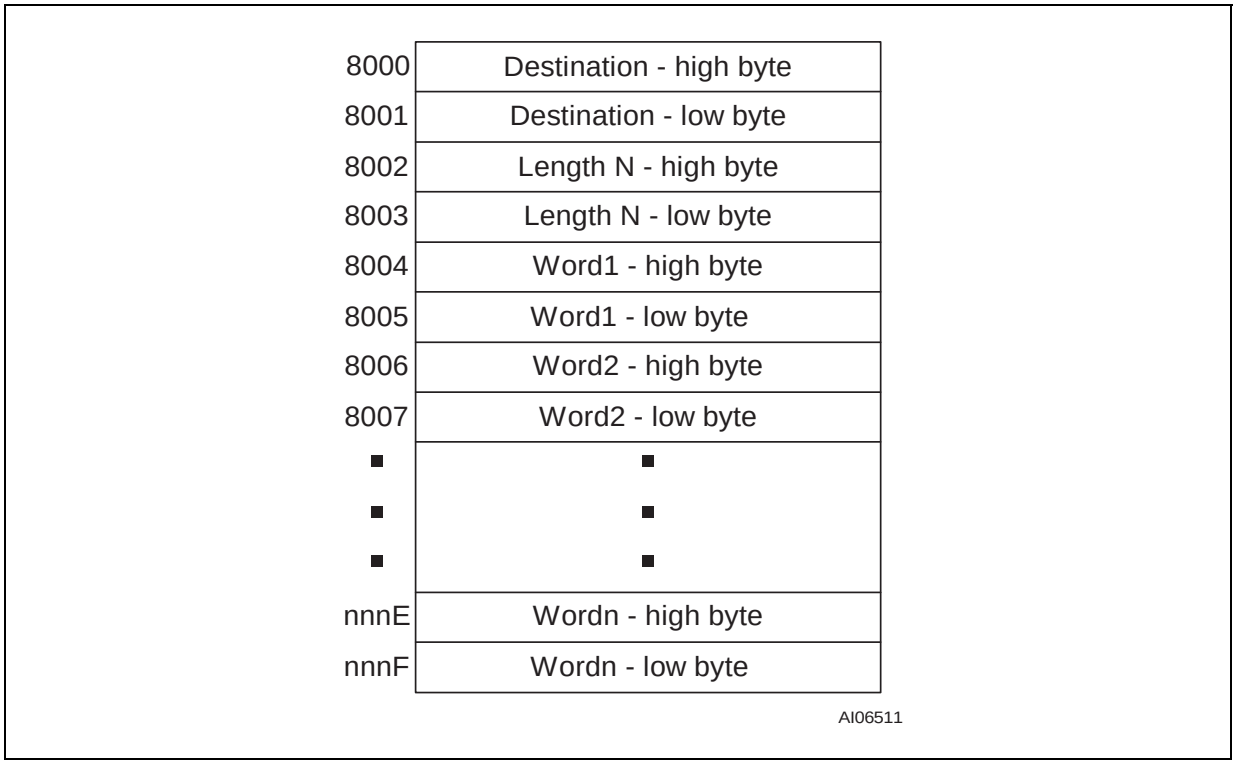
Figure 9. PSDsoft Express – Pin Definition Screen



TMS320C203 BOOT LOADER

The TMS320C203 has an on-chip boot loader that downloads program code from external 8-bit Flash/ ROM to external fast SRAM at reset. Figure 10 shows how the 16-bit program code is programmed in external Flash.

Figure 10. Flash Memory Boot Loader Store



External Boot Flash Memory Format

The boot code is stored in External Global Flash memory space in the following format:

- Destination….stores the 16-bit destination address of the external SRAM where the boot memory is to be downloaded.
- Length….stores the length of the boot program code to be downloaded to external SRAM.
- Word….stores the number of 16-bit program words, high byte first, indicated by the program length value.

Note: 1. The first four bytes are not included in the calculation of the length. The program downloaded from Flash memory starts with the fifth byte.
2. The first four words of the program memory in the Flash boot memory must contain code for the reset and interrupt vectors and must be stored in external SRAM first, at program memory addresses 000h-0003h.

Boot Loader Sequence

The boot loader is enabled when the /BOOT pin is tied low and sampled only at reset. The wait state register defaults to seven wait states to access program and data spaces. The TMS320C203 then branches to the location of the on-chip boot loader program and starts the following sequence of events:

The boot loader loads the first two bytes from Flash memory and uses this word as the destination address for the program code.

1. The boot loader loads the next two bytes to establish the code length.

2. The boot loader transfers the next two bytes, combines the two bytes into one word, and stores the new program word in external SRAM at the address location pointed to by the destination address in Flash boot memory.
3. The source and destination addresses are incremented, and the boot loader process checks and repeats itself until the entire program has been downloaded to SRAM.
4. The boot loader disables the entire Global Memory and forces a branch to the reset vector at address 0000h in program memory. When boot loading is finished, TMS320C203 switches the on-chip boot loader out of the memory map.

SUMMARY

As DSPs continue to rapidly proliferate into markets such as communications, industrial, medical, signal conditioning, and hand held test equipment, the PSD813F and DSP form an ideal 2-chip core with on-chip PLD and 27 I/O lines that can be individually configured to perform any function required by the system design. Using the PSD813F as an 8-bit boot loader in both high speed and low speed systems is an ideal and rapid design alternative to a discrete solution. Inexpensive slower memory and PLDs integrated in the PSD813F now become both cost and performance effective.

Several features internal to the PSD813F5 were used to expand the limitations of the TMS320C203, and DSPs in general:

1. Flash memory allows IAP update of the program code in the field through the serial port of the DSP while the DSP is running program code in the internal DARAM.
2. JTAG-ISP simplifies manufacturing.
3. Expanded I/O was added to the system.
4. The internal Flash PLD allows design changes, in logic, I/O and memory mapping, to be made by software modifications instead of board level hardware changes.

These changes have added to both the versatility and performance of the TMS320C203; future changes most likely will not require a hardware change to the 2-chip core.

APPENDIX

The Appendix contains the PSDsoft Express Design Assistant Summary listing all logic equations and showing how the PSD813F5 is configured to implement the example in this Application Note. Application Note AN1356 presents a step-by-step illustration of how to configure the Flash PSD Family. Although AN1356 uses the 16-bit Flash PSD4235G2 in the example, the software and procedure is the same for the 8-bit PSD813F5.

```
*********************************************************************
                        PSDsoft Express Version 6.02
                    Summary of Design Assistant
*********************************************************************
PROJECT    : TI_C203                      DATE : 09/22/2000
DEVICE     : PSD813F5                     TIME : 10:33:19
MCU        : <Other>
*********************************************************************

Pin Definitions:
================

Pin          Signal       Pin
Name         Name         Type
```

```
------------ ------------ ------------
adio0        a0           Address line
adio1        a1           Address line
adio2        a2           Address line
adio3        a3           Address line
adio4        a4           Address line
adio5        a5           Address line
adio6        a6           Address line
adio7        a7           Address line
adio8        a8           Address line
adio9        a9           Address line
adio10       a10          Address line
adio11       a11          Address line
adio12       a12          Address line
adio13       a13          Address line
adio14       a14          Address line
adio15       a15          Address line
cntl0        _wr          MCU bus control signal
cntl2        _br          Logic or address
cntl1        _rd          MCU bus control signal
reset        _reset       RESET input
pa0          d0           Data line
pa1          d1           Data line
pa2          d2           Data line
pa3          d3           Data line
pa4          d4           Data line
pa5          d5           Data line
pa6          d6           Data line
pa7          d7           Data line
pb6          _is          Logic or address
pb7          _ds          Logic or address
pc0          tms          Dedicated JTAG - TMS
pc1          tck          Dedicated JTAG - TCK
pc3          rdy_bsy_pin  Rdy/Bsy output
pc5          tdi          Dedicated JTAG - TDI
pc6          tdo          Dedicated JTAG - TDO

User defined nodes:
===================

None defined

Page Register settings:
=======================

pgr0 is used for paging
pgr1 is used for paging
pgr2 is used for paging
pgr3 is not used
pgr4 is not used
pgr5 is not used
pgr6 is not used
pgr7 is not used

Equations:
==========

csiop = ((address >= ^hFE00) & (address <= ^hFEFF) & (!_is));
fs0 = ((page == 0) & (address >= ^h8000) & (address <= ^hBF00) & (!_br))
```

```
     # ((page == 0) & (address >= ^h8000) & (address <= ^hBF00) & (!_ds & _br));
fs1 = ((page == 0) & (address >= ^hC000) & (address <= ^hFF00) & (!_br))
     # ((page == 0) & (address >= ^hC000) & (address <= ^hFFFF) & (!_ds & _br));
fs2 = ((page == 1) & (address >= ^h8000) & (address <= ^hBFFF) & (!_br))
     # ((page == 1) & (address >= ^h8000) & (address <= ^hBFFF) & (!_ds & _br));
fs3 = ((page == 1) & (address >= ^hC000) & (address <= ^hFFFF) & (!_br))
     # ((page == 1) & (address >= ^hC000) & (address <= ^hFFFF) & (!_ds & _br));
fs4 = ((page == 2) & (address >= ^h8000) & (address <= ^hBFFF) & (!_br))
     # ((page == 2) & (address >= ^h8000) & (address <= ^hBFFF) & (!_ds & _br));
fs5 = ((page == 2) & (address >= ^hC000) & (address <= ^hFFFF) & (!_br))
     # ((page == 2) & (address >= ^hC000) & (address <= ^hFFFF) & (!_ds & _br));
fs6 = ((page == 3) & (address >= ^h8000) & (address <= ^hBFFF) & (!_br))
     # ((page == 3) & (address >= ^h8000) & (address <= ^hBFFF) & (!_ds & _br));
fs7 = ((page == 3) & (address >= ^hC000) & (address <= ^hFFFF) & (!_br))
     # ((page == 3) & (address >= ^hC000) & (address <= ^hFFFF) & (!_ds & _br));
```

Table 3. Document Revision History

Date	Rev.	Description of Revision
Oct-2000	1.0	Document written in the WSI format (AN072)
01-Mar-2002	2.0	Document converted to the ST format (AN1428)

For current information on PSD products, please consult our pages on the world wide web:

www.st.com/psm

If you have any questions or suggestions concerning the matters raised in this document, please send them to the following electronic mail addresses:

apps.psd@st.com (for application support)

ask.memory@st.com (for general enquiries)

Please remember to include your name, company, location, telephone number and fax number.