

## Interfacing Fast SRAM to Freescale's DSP56300 Family of Digital Signal Processors

by Phil Brewer

### 1 Introduction

This application note describes how to interface external Asynchronous Fast Static Random Access Memory (Fast SRAM) to Motorola's DSP56300 family of devices. This document is a supplement to the *DSP56300 24-Bit Digital Signal Processor Family Manual*, and to the user's manuals and technical data sheets for devices in the DSP56300 family.

The intent is to describe methods for interfacing various types of memory to the DSP56300's Memory Expansion Port in order to assist the DSP hardware engineer to fully utilize the processor's resources and generate an optimized memory design. These memory designs use a minimum of additional devices. Taking advantage of the DSP's available control lines makes virtually glueless external memory interfaces possible, thereby reducing the cost and using fewer devices in an application's memory design.

Specifically, this application note describes implementations for asynchronous Fast SRAM using the DSP56303. The DSP56303 is representative of the DSP56300 family and has all the essential family features. Where appropriate, several memory configurations provide a complete set of examples from which the designer can choose.

## Contents

<b>1</b>	<b>Introduction .....</b>	<b>1-1</b>
1.1	DSP56300 Family .....	1-2
1.2	Application Note Organization.....	1-2
1.3	Static RAM (SRAM) Types .....	1-3
1.4	References .....	1-4
<b>2</b>	<b>Interface Overview .....</b>	<b>2-1</b>
2.1	DSP56300 Control Signals.....	2-1
2.2	DSP56300 External Memory Timing .....	2-1
2.3	DSP56303 Memory Control Registers .....	2-4
<b>3</b>	<b>32K × 8-bit Memory Based Designs .....</b>	<b>3-1</b>
3.1	32K × 24-bit Common Fast SRAM Hardware Design.....	3-2
3.2	32K × 24-bit 'P' Space Fast SRAM Example .....	3-6
3.3	32K × 24-bit 'P'/'X'/'Y' Fast SRAM Example.....	3-13
3.4	16K × 24-bit 'P'/'X' and 16K × 24-bit 'Y' Fast SRAM Example.....	3-20
<b>4</b>	<b>128K × 8-bit Memory Based Designs.....</b>	<b>4-1</b>
4.1	128K × 24-bit Common Fast SRAM Hardware Design.....	4-2
4.2	128K × 24-bit 'P'/'X'/'Y' Fast SRAM Example.....	4-6
4.3	64K × 24-bit 'X' and 64K × 24-bit 'Y' Fast SRAM Example.....	4-13
<b>5</b>	<b>64K × 24-bit Memory Based Designs.....</b>	<b>5-1</b>
5.1	64K × 24-bit Common Fast SRAM Hardware Design.....	5-2
5.2	32K × 24-bit 'P'/'X' and 32K × 24-bit 'Y' Fast SRAM Example.....	5-6
5.3	32K × 24-bit 'X' and 32K × 24-bit 'Y' Fast SRAM Example .....	5-14
<b>6</b>	<b>DSP Memory Space Configurations .....</b>	<b>6-1</b>
6.1	Fast SRAM Advantages .....	6-1

Interfacing Fast SRAM to Motorola's DSP56300 Family

## 1.1 DSP56300 Family

The Motorola DSP56300 family of DSPs uses a programmable 24-bit fixed-point core. This core is a high-performance, single-clock-cycle-per-instruction engine that provides almost twice the performance of Motorola's popular DSP56000 family core while retaining code compatibility.

The DSP56300 family core consists of a Peripheral/Memory Expansion Port (Port A), External Memory and Peripheral DRAM controller, Data Arithmetic Logic Unit (Data ALU), Address Generation Unit (AGU), Instruction Cache Controller, Program Control Unit, on-chip concurrent six-channel DMA controller, PLL Clock Generator, On-Chip Emulation (OnCE™) module, JTAG Test Access Port (TAP) compatible with the IEEE 1149.1 Standard, and a Peripheral and Memory Expansion Bus. The main features of the core include:

- Object code compatibility with the DSP56000 core
- Modified Harvard architecture with 24-bit instruction and 24-bit data width
- Fully pipelined 24 × 24-bit parallel Multiplier-Accumulator (MAC)
- 56-bit parallel barrel shifter
- 16-bit arithmetic mode of operation
- Highly parallel instruction set
- Position Independent Code (PIC) instruction-set support
- Unique DSP addressing modes
- On-chip memory-expandable hardware stack
- Nested hardware DO loops
- Fast auto-return interrupts
- On-chip instruction cache
- External Memory and Peripheral Access Attribute select support
- On-chip Phase Lock Loop (PLL)
- Program address tracing support

The main differences between derivatives of the DSP56300 family are the size of the on-chip memory and the types of on-chip peripherals and hardware accelerators.

## 1.2 Application Note Organization

This document has six sections:

1. An overview of the contents of this application note
  2. A description of the DSP56303 memory expansion port, Port A, including its use and timing characteristics
  3. Three examples of common memory space configurations based on a common hardware memory design that uses 32K x 8-bit 5.0 V Fast SRAM and two Address Attribute Selectors
  4. Two examples of common memory space configurations based on a common hardware design that uses 128K x 8-bit 3.3 V Fast SRAM and two Address Attribute Selectors
  5. Two examples of common memory space configurations based on a common hardware design that uses 64K x 24-bit 3.3 V Fast SRAM and two Address Attributes
-

6. A summary of the memory design implementations presented in the previous sections and the advantages of using Fast SRAM.

## 1.3 Static RAM (SRAM) Types

The following SRAM types were considered for this application note.

- Asynchronous Fast SRAM
  - Asynchronous Fast SRAM 8-bit
    - Fast SRAM 5.0 V
    - Fast SRAM 3.3 V
  - Asynchronous Fast SRAM 16-bit
  - Asynchronous Fast SRAM 24-bit
  - Asynchronous Fast SRAM 32-bit
    - Fast SRAM 5.0 V
    - Fast SRAM 3.3 V
  - Asynchronous Fast SRAM modules
- Cache
  - Dual I/O Synchronous SRAM
  - Burst RAM Synchronous Fast SRAM
  - Pipelined Burst RAM Synchronous Fast SRAM
  - Flow-Through Pipelined Burst RAM Synchronous Fast SRAM
  - Secondary Cache Modules
- Synchronous Fast SRAM (SSRAM)

However, only the most popular memory types and those types requiring little or no supporting hardware (i.e., glue logic) are included in this application note. The examples in this report give designers the insight necessary to implement other memory families and memory types, if needed, with DSP56300 family devices.

To achieve the fastest operation on the external memory bus and, consequently, the fewest number of DSP wait states requires the fastest memory available (i.e., Fast SRAM). Slower SRAM can be substituted for the Fast SRAM resulting in DSP-generated external memory wait states each of which is roughly equivalent to one clock period of the DSP core. Each wait state is 12.5 nS for a DSP core running at 80 MHz.

An asynchronous external access for a DSP56300 family device incurs an automatic one wait state penalty when the address for the external memory device is required to be stable during the entire access. Since asynchronous Fast SRAM and SRAM devices have this address stability requirement, the DSP operates with at least one wait state when using these external memories.

---

## 1.4 References

*Motorola DSP56300 24-bit Digital Signal Processor Family Manual (DSP56300FM/AD)*, Motorola, 1995.

*DSP56301 Technical Data Sheet (DSP56301/D)*, Motorola, 1995.

*DSP56302 Technical Data Sheet (DSP56302/D)*, Motorola, 1996.

*DSP56303 Technical Data Sheet (DSP56303/D)*, Motorola, 1996.

*Motorola DSP56303 24-bit Digital Signal Processor (DSP56303UM/AD)*, Motorola, 1996.

*Motorola Dynamic RAM's & Memory Modules (DL155/D)*, Motorola, 1993.

*Motorola Fast Static RAM Component and Module Data (DL156/D)*, Rev 3, Motorola, 1995.

*Advanced Micro Devices CMOS Memory Products 1991 Data Book/Handbook*, AMD, 1991.

*Advanced Micro Devices FLASH Memory Products 1994/1995 Data Book/Handbook*, AMD, 1995.

*Quality Semiconductor QuickSwitch<sup>®</sup> Products Databook*, Quality Semiconductor, 1995.

Quality Semiconductor Application Note AN-11, *Bus Switches Provide 5 V and 3 V Logic Conversion with Zero Delay*, 1995.

---

## 2 Interface Overview

This section describes how external memory devices are interfaced to Motorola's DSP56300 family of devices using its memory and peripheral expansion port.

### 2.1 DSP56300 Control Signals

Only the DSP control signals used in the memory implementation examples in this note are described below. Other memory configuration implementations may require other support features, which the DSP56300 family of devices have available to assist the designer. These additional memory control signals are described in the Port A Chapter of the *DSP56300 24-Bit Digital Signal Processor Family Manual*.

- Read Data Enable ( $\overline{RD}$ )—An active low output signal that is asserted during an external memory or peripheral read access.
- Write Data Enable ( $\overline{WR}$ )—An active low output signal that is asserted during an external memory or peripheral write access.
- Address Bus (A0–A17)—Eighteen address lines that allow the DSP563003 to directly address 256K words of external memory or peripherals. These active high output signals are asserted only during external memory or peripheral read or write accesses. These signal lines maintain state when external memory spaces are not being accessed.
- Data Bus (D0–D23)—Twenty-four data lines on the DSP56303 that are active high bidirectional signals asserted only during external program and data memory accesses. These signal lines maintain state when external memory spaces are not being accessed.
- Address Attribute/Row Address Strobe (AA[0–3]/ $\overline{RAS}$ [0–3])—Four Address Attribute or Row Address Strobe signals. When the Address Attribute, AA, option is selected for these signal lines, they can function as chip selects or additional address lines. When the Row Address Strobe,  $\overline{RAS}$ , option is selected for these signal lines, they can function as Row Address Strobe lines for DRAM interfacing.

### 2.2 DSP56300 External Memory Timing

The DSP56300 family derives its core clock from one of various sources (see PLL and Clock Generator chapter in the *DSP56300 24-Bit Digital Signal Processor Family Manual* for details). All memory interface timings are derived from the period of the DSP core clock. For example, if the DSP core clock frequency is 80 MHz, then the memory timings are based on a 12.5 nS clock cycle time, and an external memory typically requires less than 12.5 nS access time for one DSP wait state operation. However, these timings are affected by several factors, such as the use of the Phase Lock Loop, the use of an external frequency over or under 4 MHz, the source of the external frequency, and propagation delays in the DSP itself. Any of these factors can cause this value to deviate from 12.5 nS. **Table 2-1** represents expected required memory performance data at an 80 MHz DSP core frequency and various wait states using the DSP56303.

---

**Table 2-1.** External Memory Speeds with DSP Wait States

External Clock (MHz)	DF	MF	PDF	WS	Core Clock (MHz)	T <sub>C</sub> (nS)	t <sub>AA</sub> – max (nS)	t <sub>AW</sub> – min (nS)
4.00	1	20	1	1	80.00	12.5	12.4	17.9
4.00	1	20	1	2	80.00	12.5	24.9	30.4
4.00	1	20	1	3	80.00	12.5	37.4	42.9
4.00	1	20	1	4	80.00	12.5	49.9	55.4
4.00	1	20	1	5	80.00	12.5	62.4	67.9
4.00	1	20	1	6	80.00	12.5	74.9	80.4
4.00	1	20	1	7	80.00	12.5	87.4	92.9
4.00	1	20	1	8	80.00	12.5	99.9	105.4
4.00	1	20	1	9	80.00	12.5	112.4	117.9
4.00	1	20	1	10	80.00	12.5	124.9	130.4
4.00	1	20	1	11	80.00	12.5	137.4	142.9
4.00	1	20	1	12	80.00	12.5	149.9	155.4
4.00	1	20	1	13	80.00	12.5	162.4	167.9
4.00	1	20	1	14	80.00	12.5	174.9	180.4
4.00	1	20	1	15	80.00	12.5	187.4	192.9
4.00	1	20	1	16	80.00	12.5	199.9	205.4
4.00	1	20	1	17	80.00	12.5	212.4	217.9
4.00	1	20	1	18	80.00	12.5	224.9	230.4
4.00	1	20	1	19	80.00	12.5	237.4	242.9
4.00	1	20	1	20	80.00	12.5	249.9	255.4

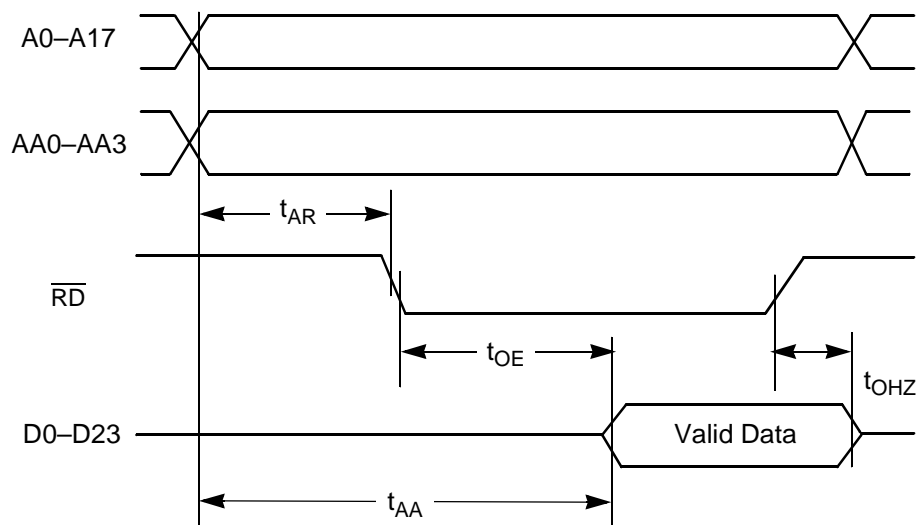
DF = PLL Division Factor  
 MF = PLL Multiplication Factor  
 PDF = PLL Pre-Division Factor  
 WS = wait states  
 TC = Clock Cycle Time  
 t<sub>AA</sub> = Data access time (i.e., address and AA valid to input data valid)  
 t<sub>AW</sub> = Data access time (i.e., address and AA valid to  $\overline{WR}$  deassertion)

## 2.2.1 DSP56303 External Memory Bus Asynchronous Read Timing

When reading from external asynchronous memory, the DSP56303 memory read access is controlled by the following steps.

1. The required memory address is asserted. The memory address is created by combining the address bus, A0–A17, and the Address Attributes, AA0–AA3.
2. After a delay of  $t_{AR}$  (i.e., address valid to  $\overline{RD}$  assertion time), the Read enable signal,  $\overline{RD}$ , is asserted.
3. Before a delay of  $t_{OE}$  (i.e.,  $\overline{RD}$  assertion to input data valid), the memory device puts valid data on the data bus.
4. The DSP latches the data bus data and deasserts  $\overline{RD}$ . The DSP does not require any data hold time,  $t_{OHZ}$ , after deassertion of the  $\overline{RD}$  signal.

The data access time,  $t_{AA}$  (i.e., address and AA valid to input data valid), is the time delay typically used by memory devices to specify data access timing. The  $t_{AA}$  for a memory device must be less than or equal to the DSP's  $t_{AA}$  time for valid data transfers.



**Figure 2-1.** External Memory Bus Asynchronous Read Timing

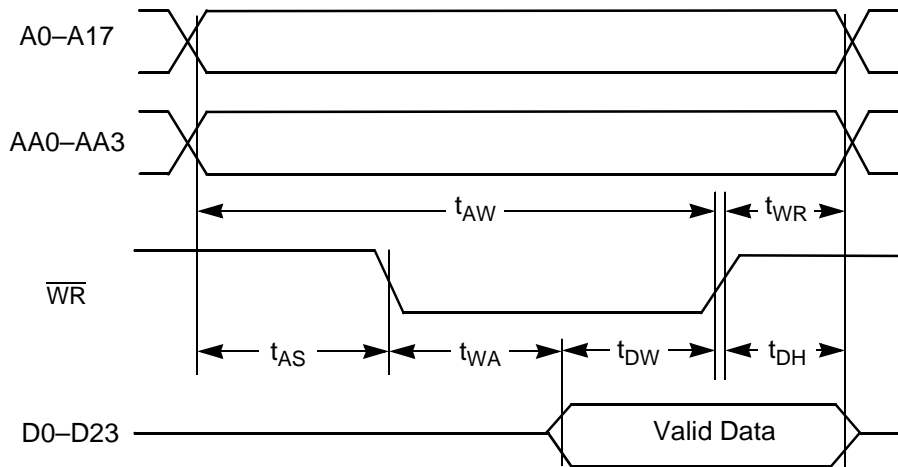
## 2.2.2 DSP56303 External Memory Bus Asynchronous Write Timing

When writing to external asynchronous memory, the DSP56303 memory write access is controlled by the following steps.

1. The memory select address is asserted. The memory select address is created by combining the Address bus, A0–A17, and the Address Attributes AA0–AA3.
2. After a delay of  $t_{AS}$ —address valid to  $\overline{WR}$  assertion time—the Write enable signal,  $\overline{WR}$ , is asserted.
3. Before a delay of  $t_{WA}$ — $\overline{WR}$  assertion to output data valid—the DSP places valid data on the data bus.

4. After a delay of  $t_{DW}$ —data valid to  $\overline{WR}$  deassertion (data setup time)—the DSP deasserts the  $\overline{WR}$  signal.
5. Then the DSP deasserts the address and address attributes after  $t_{WR}$ — $\overline{WR}$  deassertion to address not valid—while holding the data valid for  $t_{DH}$ .

The data access time,  $t_{AW}$  (i.e., address and AA valid to  $\overline{WR}$  deassertion), is typically the critical timing specification for memory devices. The  $t_{AW}$  for a memory device must be less than or equal to the DSP's  $t_{AW}$  time for valid data transfers.



**Figure 2-2.** External Asynchronous Memory Bus Write Timing

## 2.3 DSP56303 Memory Control Registers

You must configure the following control registers to access external memory or peripherals properly when using the DSP56303:

- DSP PLL and Clock Generation register
- Bus Control Register
- DRAM Control Register (if DRAM is used)
- Address Attribute Registers



### 2.3.1 DSP PLL and Clock Generation

You must set the core speed of the DSP for optimum processor and memory performance by configuring the DSP PLL and Clock Generation in the PLL Control (PCTL) register. For detailed information on the PCTL register, see the PLL and Clock Generator chapter in the *DSP56300 24-Bit Digital Signal Processor Family Manual*. The PLL Control register is an X data I/O mapped 24-bit register. The PCTL register can be separated into four sub-functions:

- Frequency Predivider—The input clock frequency can be pre-divided before passing it to the PLL loop frequency multiplier. This frequency predivider has a programmable Division Factor range of 1 to 16. It is set by controlling the values placed in the PCTL register bits 20–23. The Division Factor is the binary value stored in bits 20–23, plus one.
- PLL Loop Frequency Multiplier—The clock frequency output from the predivider is multiplied by the voltage-controlled oscillator (VCO). The Multiplication Factor is set by the value in the PCTL register bits 0–11. The Multiplication Factor is the binary value stored in bits 0–11, plus one.
- Frequency Low-power Divider—The Low-power Divider (LPD) can divide the output frequency of the VCO before it is used by the DSP core. This frequency low power divider has a programmable Division Factor range of 1 to 128. It is set by controlling the values placed in the PCTL register bits 12–14. The low-power division factor is  $2^n$ , where n is the value in PCTL bits 12–14.
- Frequency Control Bits—The following five control bits control the input frequency source, the PLL during Stop mode, the activation of the PLL VCO, and the external availability of the core clock:
  - Crystal frequency is less than 200 kHz, Bit 15
  - Disable XTAL drive output, Bit 16
  - PLL runs during STOP mode, Bit 17
  - Enable PLL operation, Bit 18
  - Disable core clock output, Bit 19

The operating core frequency of the chip is set by the control bits in the PCTL register as follows:

$$F_{\text{CORE}} = \frac{F_{\text{EXTAL}} \times \text{MF}}{\text{PDF} \times \text{DF}}$$

where

- $F_{\text{CORE}}$  is the DSP core frequency.
- $F_{\text{EXTAL}}$  is the external input frequency source present on the EXTAL pin.
- PDF is the Predivider Factor defined by the PD0–PD3 bits in PCTL.
- MF is the PLL Multiplication Factor defined by the MF0–MF11 bits in PCTL.
- DF is the Division Factor defined by the DF0–DF2 bits in PCTL.

### 2.3.2 Bus Control Register (BCR)

The Bus Control Register (BCR) is a 24-bit X data I/O register that controls the external bus wait states generated for each Address Attribute area 0–3 and assigns a default value to all memory areas not covered by an Address Attribute area. Each area can have up to 31 wait states. Select the correct number of wait states for each memory configuration using this register.

- Wait states for Address Attribute area 0, allowing 0–31 wait states, Bits 0–4
- Wait states for Address Attribute area 1, allowing 0–31 wait states, Bits 5–9
- Wait states for Address Attribute area 2, allowing 0–7 wait states, Bits 10–12
- Wait states for Address Attribute area 3, allowing 0–7 wait states, Bits 13–15
- Wait states for address areas not specified by areas 0–3, allowing 0–31 wait states, Bits 16–20
- The bus state status, Bit 21
- Enable Bus Lock Hold, Bit 22
- Enable Bus Request Hold, Bit 23

### 2.3.3 Address Attribute Control Registers (AAR0–AAR3)

Four 24-bit Address Attribute Control registers in the X data I/O memory space control the activity of the AA0–AA3/ $\overline{\text{RAS0}}$ – $\overline{\text{RAS3}}$  pins. Each AA/ $\overline{\text{RAS}}$  pin is asserted if the address and memory space of the appropriate AARx matches the requested external memory address and address space.

- Specify external memory access type; select from Synchronous SRAM, Asynchronous SRAM, and DRAM accesses, Bits 0–1
  - Pull the AA pin high, Bit 2
  - Activate the AA pin during external program space accesses, Bit 3
  - Activate the AA pin during external X data space accesses, Bit 4
  - Activate the AA pin during external Y data space accesses, Bit 5
  - Move the eight least significant bits of the address to the eight most significant bits of the external address bus, Bit 6
  - Enable the internal packing/unpacking logic during external DMA accesses, Bit 7
  - Specify the number of address bits to compare, allowing the use of 0–12 address bits, Bits 8–11
  - Specify the most significant portion of the address to compare, Bits 12–23
-

### 2.3.4 Operating Mode Register (OMR)

The Operating Mode Register (OMR) is a 24-bit I/O register that selects the operating mode of the DSP, external memory controls, and stack extension controls. The following flags are applicable to memory interfacing:

- The DSP operating mode is specified by MA–MD, Bits 0–3.
- The External Bus Disable bit disables the external bus controller for power conservation when external memory is not used, Bit 4.
- The Memory Switch mode bit reconfigures internal memory spaces, Bit 7.
- The Transfer Acknowledge Synchronize Select bit selects the synchronization method for the Transfer Acknowledge (TA) pin, Bit 11.
- The Bus Release Timing bit selects between a fast and slow bus release of the  $\overline{BB}$  pin, Bit 12.
- The Address Attribute Priority Disable bit allows the Address Attribute pins, AA0–AA3, to be used in any combination, Bit 14.

### 2.3.5 Status Register (SR)

The Status Register (SR) is a 24-bit I/O register that selects and monitors the results of arithmetic computations and the current state of the DSP. The following flags are applicable to memory interfacing:

- Sixteen-bit Compatibility mode enables full compatibility with object code written for the DSP56000 family, Bit 13.
  - Instruction Cache Enable bit enables the instruction cache controller and changes the last 1K of internal program memory into cache memory, Bit 19.
-



## 3 32K × 8-bit Memory Based Designs

Using one hardware memory design based on three 32K × 8-bit 5.0 V memories, the DSP56303's Memory Expansion Port allows the 32K × 24-bit memory bank to be logically configured for use in various memory space arrangements.

Configuring and using one Memory Expansion Port Address Attribute Control Register, the 32K memory bank can accommodate seven different memory space arrangements:

1. 32K × 24-bit 'P' Space Fast SRAM
2. 32K × 24-bit 'X' Space Fast SRAM
3. 32K × 24-bit 'Y' Space Fast SRAM
4. 32K × 24-bit 'P'/'X' Space Fast SRAM
5. 32K × 24-bit 'P'/'Y' Space Fast SRAM
6. 32K × 24-bit 'X'/'Y' Space Fast SRAM
7. 32K × 24-bit 'P'/'X'/'Y' Space Fast SRAM

Configuring and using two Memory Expansion Port Address Attribute Control Registers, the 32K memory bank can accommodate thirteen different memory space arrangements:

1. 32K × 24-bit 'P' Space Fast SRAM
2. 32K × 24-bit 'X' Space Fast SRAM
3. 32K × 24-bit 'Y' Space Fast SRAM
4. 32K × 24-bit 'P'/'X' Space Fast SRAM
5. 32K × 24-bit 'P'/'Y' Space Fast SRAM
6. 32K × 24-bit 'X'/'Y' Space Fast SRAM
7. 32K × 24-bit 'P'/'X'/'Y' Space Fast SRAM
8. 16K × 24-bit 'P'/'X' and 16K × 24-bit 'Y' Space Fast SRAM
9. 16K × 24-bit 'P'/'Y' and 16K × 24-bit 'X' Space Fast SRAM
10. 16K × 24-bit 'P' and 16K × 24-bit 'X'/'Y' Space Fast SRAM
11. 16K × 24-bit 'P' and 16K × 24-bit 'X' Space Fast SRAM
12. 16K × 24-bit 'P' and 16K × 24-bit 'Y' Space Fast SRAM
13. 16K × 24-bit 'X' and 16K × 24-bit 'Y' Space Fast SRAM

All of these memory space configurations efficiently use the full capacity of the memory chips in the 32K × 24-bit memory bank.

To illustrate these configurations, the remainder of this chapter presents examples based on one common hardware design using two Address Attribute Selectors that implement three of the most common configurations: 32K × 24-bit 'P' Space Fast SRAM, 32K × 24-bit 'P'/'X'/'Y' Space Fast SRAM configuration, and a 16K × 24-bit 'P'/'X' and 16K × 24-bit 'Y' Space Fast SRAM configuration (see **Figure 3-4** for a schematic of the hardware design).

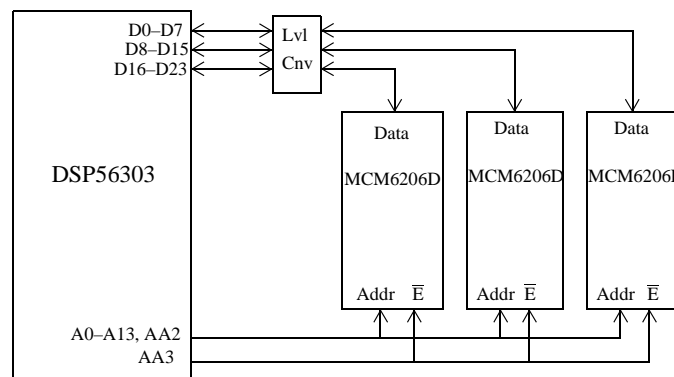
### 3.1 32K × 24-bit Common Fast SRAM Hardware Design

This section describes an asynchronous Fast SRAM 32K × 24-bit memory bank implementation using Motorola’s MCM6206D device. **Figure 3-1** displays the block diagram. Memory bank designs of this size and type can use very low-cost 5.0 V memory devices to satisfy the majority of embedded designs.

The memory bank design uses two Address Attribute Selectors. However, the 32K × 24-bit ‘P’ Space and 32K × 24-bit ‘P’/‘X’/‘Y’ Space Fast SRAM configurations can use a one Address Attribute Selector design with A14 substituted for AA2. The two Address Attribute Selector design demonstrates the ultimate flexibility of the Address Attribute Selectors.

For this common hardware design, the DSP core runs at a maximum of 80 MHz, and the input frequency source is a 4.000 MHz crystal. The asynchronous Fast SRAM requires an access time equal to or less than 12.4 nS to satisfy the DSP’s one wait state external memory requirements. This 5 V memory device is organized as 32K × 8-bits. Therefore, three memory devices are used to achieve the 24-bit wide bus.

Since the DSP’s data bus is not 5 V tolerant, level conversion to and from 3.3 V and 5 V is necessary on the 24-bit data to accommodate the 5 V memory devices. This is accomplished by using three Quality Semiconductor’s QS3245 QuickSwitch® 8-bit bus switches. These switches allow the connection of a 3.3 V CMOS logic DSP data bus on one side and 5 V TTL-compatible logic, memory devices on the other side, effectively providing a 3.3 V-to-5 V level conversion without adding any significant (0.25 nS) propagation delay.



**Figure 3-1.** 32K × 24-bit Fast SRAM Memory Example

#### 3.1.1 MCM6206D-12 Memory Timing Requirements

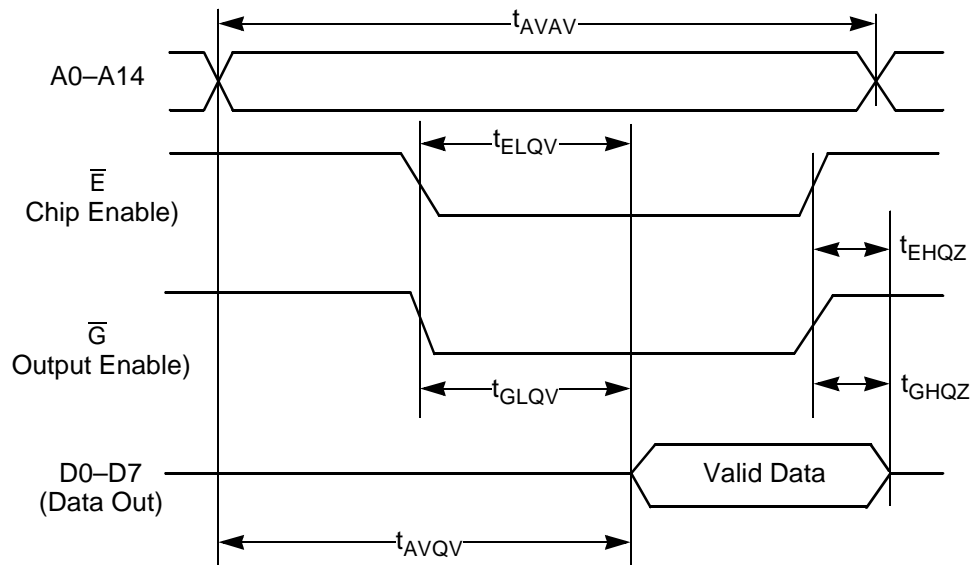
For the asynchronous Fast SRAM device to work properly, its timing requirements must be met. Following are the timing requirements for the MCM6206D-12 32K × 8-bit 12 nS Fast SRAM.

##### 3.1.1.1 MCM6206D-12 Read Cycle Timing

**Table 3-1** shows the memory read timing specification values in the memory read cycle timing diagram, **Figure 3-2**.

**Table 3-1.** MCM6206D-12 Memory Read Timing Specifications

Read Cycle Parameter	Symbol	Min	Max
Read Cycle Time	$t_{AVAV}$	12 nS	—
Address Access Time	$t_{AVQV}$	—	12 nS
Enable Access Time	$t_{ELQV}$	—	12 nS
Output Enable Access Time	$t_{GLQV}$	—	6 nS
Enable High to Output High-Z	$t_{EHQZ}$	0 nS	7 nS
Output Enable High to Output High-Z	$t_{GHQZ}$	0 nS	6 nS

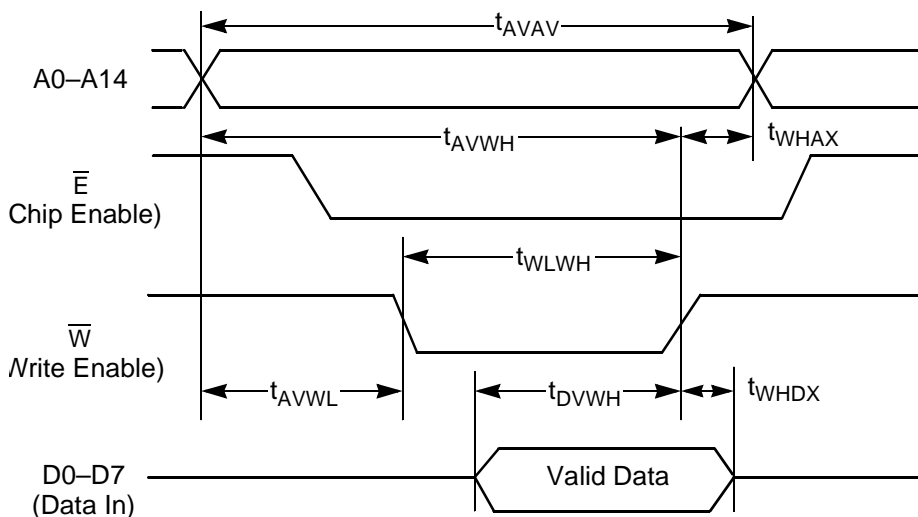

**Figure 3-2.** MCM6206D-12 Memory Read Cycle Timing Diagram

### 3.1.1.2 MCM6206D-12 Write Cycle Timing

**Table 3-2** shows the memory write timing specification values in the memory write cycle timing diagram, **Figure 3-3**.

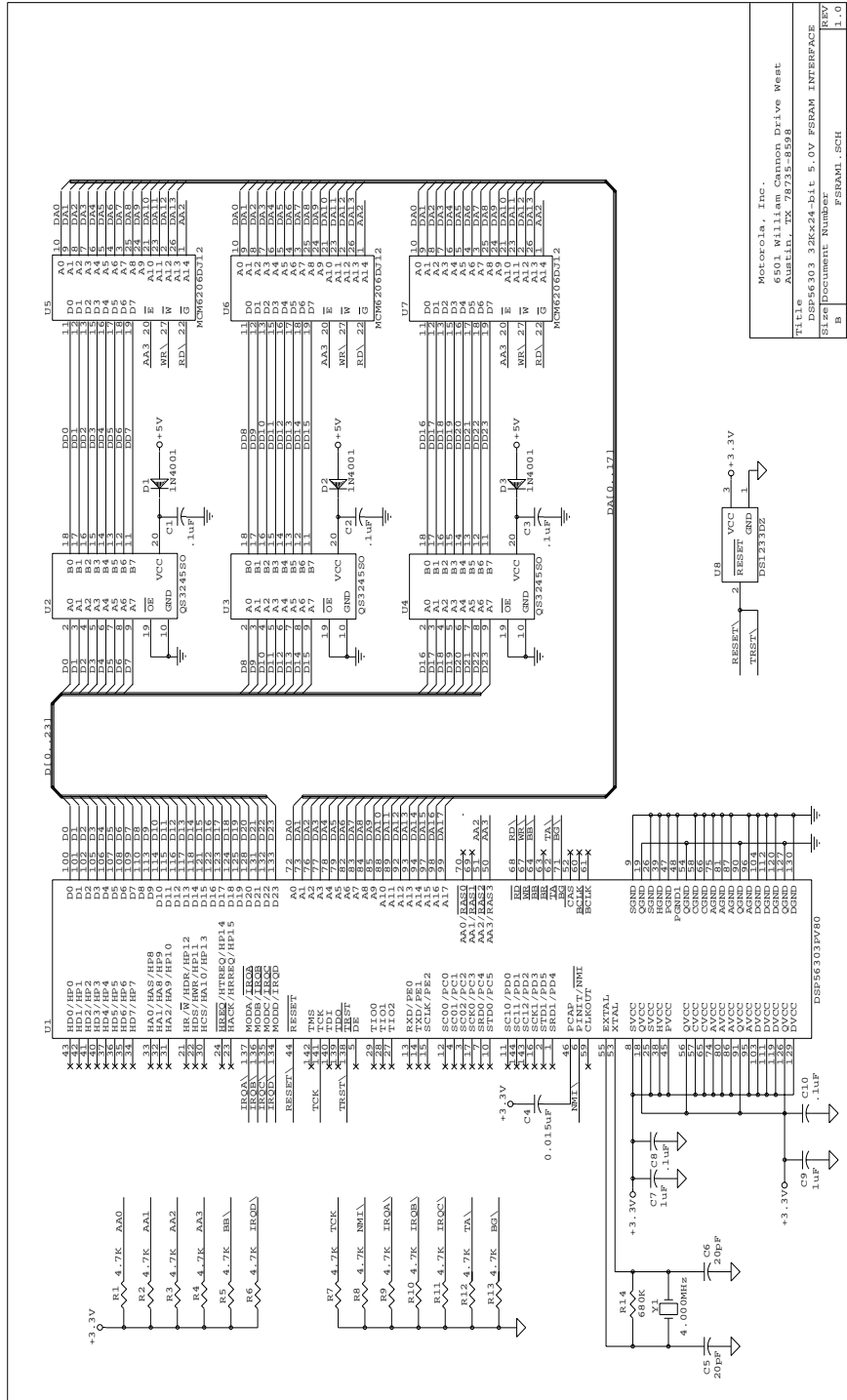
**Table 3-2.** MCM6206D-12 Memory Write Timing Specifications

Write Cycle Parameter	Symbol	Min	Max
Write Cycle Time	$t_{AVAV}$	12 nS	—
Address Setup Time	$t_{AVWL}$	0 nS	—
Address Valid to End off Write	$t_{AVWH}$	10 nS	—
Write Pulse Width	$t_{WLWH}$	10 nS	—
Data Valid to End of Write	$t_{DVWH}$	6 nS	—
Data Hold Time	$t_{WHDX}$	0 nS	—
Write Recovery Time	$t_{WHAX}$	0 nS	—



**Figure 3-3.** MCM6206D-12 Memory Write Cycle Timing Diagram

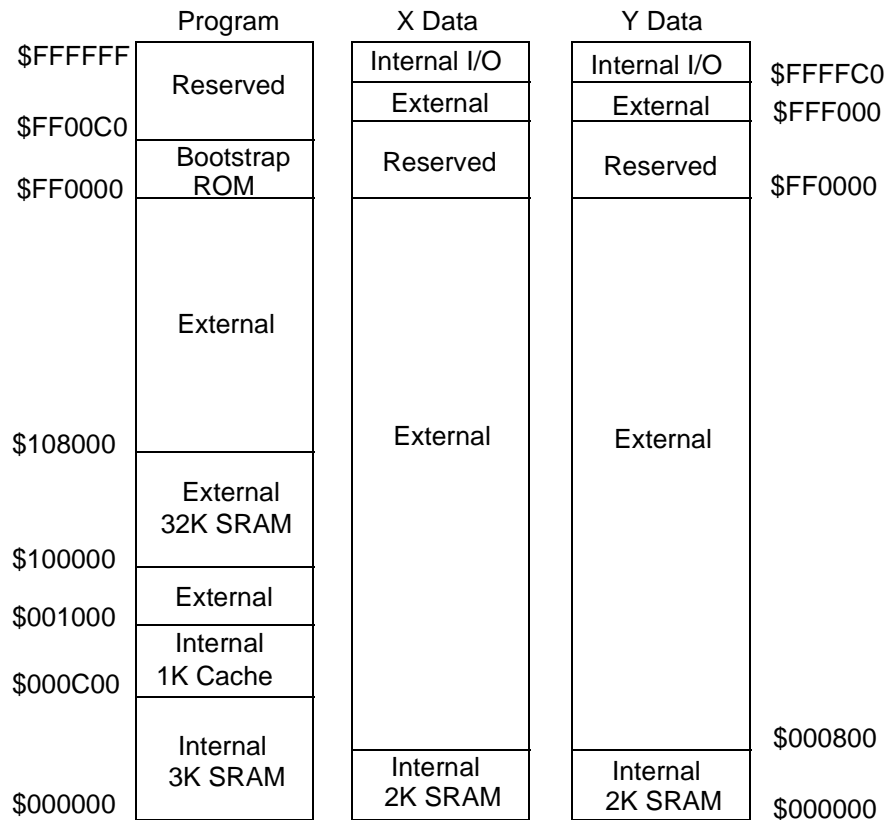




## 3.2 32K × 24-bit ‘P’ Space Fast SRAM Example

This section describes a 32K × 24-bit ‘P’ memory space, asynchronous Fast SRAM implementation using Motorola’s MCM6206D device. **Figure 3-5** shows the memory map layout, **Figure 3-1** shows the block diagram, and **Example 3-1** shows the example code.

The DSP core runs at 80 MHz, and the input frequency source is a 4.000 MHz crystal. The asynchronous Fast SRAM requires an access time equal to or less than 12.4 nS to satisfy the DSP’s one wait state external memory requirements.



**Figure 3-5.** 32K × 24-bit ‘P’ Space Fast SRAM Memory Map

### 3.2.1 DSP56303 Port A Timing Requirements and Register Settings

For optimal use of the 32K × 24-bit ‘P’ space memory configuration, set up the following DSP control registers.

Set up the core speed of the DSP for optimum processor and memory performance using the DSP PLL and Clock Generation (PCTL) register. For this example, the DSP core runs at 80 MHz, and the input frequency source is a 4.000 MHz crystal. The PCTL register value combines the following bits for each feature:

- Desired Core Frequency = 80 MHz
- Given the External Frequency = 4.000 MHz
- Predivider value = 1, bits 20–23 = \$0
- Low-power Divider value = 1, bits 12–14 = \$0
- VCO Multiplication value = 20, bits 0–11 = \$013
- Crystal less than 200 kHz, Bit 15 = 0
- Disable XTAL drive output, Bit 16 = 0
- PLL runs during STOP, Bit 17 = 1
- Enable PLL operation, Bit 18 = 1
- Disable core clock output, Bit 19 = 1

The value loaded into the PCTL register is \$0E0013.

Address Attribute Pin 3 enables, via Fast SRAM  $\bar{E}$ , external 32K SRAM bank accesses in the address range from \$100000 to \$107FFF during program space requests. Configure the memory address space requirements for the Address Attribute Pin 3 using Address Attribute Register 3 (AAR3). The AAR3 value combines the following bits for each feature:

- Specify the external memory access type as asynchronous SRAM, Bits 0–1 = \$1.
- Pull the AA pin high when selected, Bit 2 = 0.
- Activate the AA pin during external program space accesses, Bit 3 = 1.
- Activate the AA pin during external X data space accesses, Bit 4 = 0.
- Activate the AA pin during external Y data space accesses, Bit 5 = 0.
- Move the eight least significant bits of the address to the eight most significant bits of the external address bus, Bit 6 = 0.
- Enable the internal packing/unpacking logic during external DMA accesses, Bit 7 = 0
- Specify the number of address bits to compare, Bits 8–11 = \$9
- Specify the most significant portion of the address to compare, Bits 12–23 = \$100

The value loaded into the AAR3 is \$100909.

---

Address Attribute Pin 2 selects, via Fast SRAM A14, address line A14 in the external 32K SRAM bank during accesses in the address range from \$100000 to \$107FFF during program space requests. Configure the memory address space requirements for the Address Attribute Pin 2 using Address Attribute Register 2 (AAR2). The AAR2 value combines the following bits for each feature:

- Specify the external memory access type as asynchronous SRAM, Bits 0–1 = \$1.
- Pull the AA pin high when selected, Bit 2 = 1.
- Activate the AA pin during external program space accesses, Bit 3 = 1.
- Activate the AA pin during external X data space accesses, Bit 4 = 0.
- Activate the AA pin during external Y data space accesses Bit 5 = 0.
- Move the eight least significant bits of address to eight most significant bits of the external address bus, Bit 6 = 0.
- Enable the internal packing/unpacking logic during external DMA accesses, Bit 7 = 0.
- Specify the number of address bits to compare, Bits 8–11 = \$A
- Specify the most significant portion of the address to compare, Bits 12–23 = \$104

The value loaded into the AAR2 is \$104A0D.

The value loaded into AAR0 and AAR1 is \$000000.

Select the proper number of wait states for the memory configuration using the Bus Control Register (BCR). The BCR value combines the following bits for each feature:

- Address Attribute area 0 wait states, Bits 0–4 = \$0
- Address Attribute area 1 wait states, Bits 5–9 = \$0
- Address Attribute area 2 wait states, Bits 10–12 = \$1
- Address Attribute area 3 wait states, Bits 13–15 = \$1
- Default address area wait states, Bits 16–20 = \$0
- Bus state status, Bit 21 = 0
- Enable Bus Lock Hold, Bit 22 = 0
- Enable Bus Request Hold, Bit 23 = 0

The value loaded into the BCR is \$002400.

---

Configure the operating mode and external memory controls using the Operating Mode Register. The OMR value combines the following bits for each feature:

- MA–MD bits select the DSP operating mode, Bits 0–3 = \$0.
- External Bus Disable bit disables the external bus controller for power conservation when external memory is not used, Bit 4 = \$0.
- Memory Switch Mode bit reconfigures internal memory spaces, Bit 7 = \$0.
- Transfer Acknowledge Synchronize Select bit selects the synchronization method for the Transfer Acknowledge (TA) pin, Bit 11 = \$0.
- Bus Release Timing bit selects between a fast and slow bus release of the  $\overline{BB}$  pin, Bit 12 = \$0.
- Address Attribute Priority Disable bit allows the Address Attribute pins, AA0–AA3, to be used in any combination, Bit 14 = \$1.
- All other OMR bits are selected for their defaults of \$000000.

The value loaded into the OMR is \$004000.

Configure the memory mode of the DSP using the Status Register (SR). The SR value combines the following bits for each feature:

- Sixteen-Bit Compatibility mode enables full compatibility to object code written for the DSP56000 family of DSPs, Bit 13 = \$0.
- Instruction Cache Enable bit enables the instruction cache controller and changes the last 1K of internal program memory into cache memory, Bit 19 = \$1.
- All other Status Register bits are selected for their defaults of \$000000.

The value loaded into the SR is \$080000, which is the value loaded during reset.

---

**Example 3-1. 32K × 24-bit 'P' Space Fast SRAM Memory Exercise Program**

Motorola DSP56300 Assembler Version 6.0.1.6 97-01-25 08:05:13 asram1.asm

```
1
2
3
4
5
6
7
8
9
10 100000 PMemStart equ $100000
11 108000 PMemEnd equ $108000
12 008000 PMemSize equ PMemEnd-PMemStart
13
14 ;--- Program Specific Storage Locations (X DATA SPACE)
15 MEM_FAIL_ADDRESS
16 000000 equ $000000
17 MEM_FAIL_WROTE
18 000001 equ $000001
19 MEM_FAIL_READ
20 000002 equ $000002
21 MEM_PASS_COUNTER
22 000003 equ $000003
23
24 ;--- DSP56303 Control Registers (X I/O SPACE)
25 FFFFFB BCR equ $FFFFFB ; Bus Control Register
26 FFFFFD PCTL equ $FFFFFD ; PLL Control Register
27 FFFFF6 AAR3 equ $FFFFF6 ; Address Attribute Register #3
28 FFFFF7 AAR2 equ $FFFFF7 ; Address Attribute Register #2
29
30 ;--- PCTL value = 0x0E0013
31 000000 prediv equ 0 ; Pre-Divider = 1
32 000000 lowdiv equ 0 ; Low Power Divider = 1
33 000013 pllmul equ 19 ; VCO Mult = 20; (19+1)*4.00MHz=80.00MHz
34 000000 crystal equ 0 ; No, Crystal not less than 200kHz
35 000000 disXTAL equ 0 ; No, do not disable crystal use
36 020000 pllstop equ $020000 ; Yes, PLL runs during STOP
37 040000 enpll equ $040000 ; Yes, enable PLL operation
38 080000 disclk equ $080000 ; Yes, disable CORE clock output
39 0E0013 PCTL_value equ rediv+lowdiv+pllmul+crystal+disXTAL+pllstop+enpll+disclk
40
41 ;--- AAR3 value = 0x100909
42 000001 acctype3 equ 1 ; External Memory access type = 0x1
43 000000 aahigh3 equ 0 ; Enable AA3 pin to be low when selected
44 000008 aap3 equ $8 ; Yes, Enable AA3 pin on ext 'P' accesses
45 000000 aax3 equ 0 ; No, Enable AA3 pin on ext 'X' accesses
46 000000 aay3 equ 0 ; No, Enable AA3 pin on ext 'Y' accesses
47 000000 aswap3 equ 0 ; No, Enable address bus swap
48 000000 enpack3 equ 0 ; No, Enable packing/unpacking logic
49 000900 nadd3 equ $000900 ; Compare 9 address bits
50 100000 msadd3 equ $100000 ; Most significant portion of address,
51 ; $100000 - $107fff, to compare.
52 ; (0001,0000,0xxx,xxxx,xxxx,xxxx)
53 100909 AAR3_value equ acctype3+aahigh3+aap3+aax3+aay3+aswap3+enpack3+nadd3+msadd3
54
55 ;--- AAR2 value = 0x104A0D
56 000001 acctype2 equ 1 ; External Memory access type = 0x1
57 000004 aahigh2 equ $4 ; Enable AA2 pin to be high when selected
58 000008 aap2 equ $8 ; Yes, Enable AA2 pin on ext 'P' accesses
59 000000 aax2 equ 0 ; No, Enable AA2 pin on ext 'X' accesses
60 000000 aay2 equ 0 ; No, Enable AA2 pin on ext 'Y' accesses
61 000000 aswap2 equ 0 ; No, Enable address bus swap
62 000000 enpack2 equ 0 ; No, Enable packing/unpacking logic
63 000A00 nadd2 equ $000A00 ; Compare 10 address bits
64 104000 msadd2 equ $104000 ; Most significant portion of address,
65 ; $104000 - $107fff, to compare.
66 ; (0001,0000,01xx,xxxx,xxxx,xxxx)
67 104A0D AAR2_value equ acctype2+aahigh2+aap2+aax2+aay2+aswap2+enpack2+nadd2+msadd2
68
69 ;--- BCR value = 0x002400
70 000000 aaa0ws equ 0 ; Address Attribute Area 0 w/s = 0
71 000000 aaalws equ 0 ; Address Attribute Area 1 w/s = 0
72 000400 aaa2ws equ $000400 ; Address Attribute Area 2 w/s = 0
73 002000 aaa3ws equ $002000 ; Address Attribute Area 3 w/s = 1
74 000000 defws equ 0 ; Default Address Area w/s = 0
```



```
75 000000  busss      equ    0          ; Bus state status = 0
76 000000  enblh     equ    0          ; Enable Bus Lock Hold = 0
77 000000  enbrh     equ    0          ; Enable Bus Request Hold = 0
78 002400  BCR_value equ    aaa0ws+aaa1ws+aaa2ws+aaa3ws+defws+busss+enblh+enbrh
79
80          ;-----
81 P:000100  org      p:$100          ;Keep the program in internal RAM
82
83          memtst
84
85          ; Initialization Section
86 P:000100  08F4BD   movep   #PCTL_value,x:PCTL   ; Set PLL Control Register
87          0E0013
88 P:000102  05F43A   movec   #$004000,OMR        ; Disable Address Attribute Priority
89          004000
90 P:000104  05F439   movec   #$080000,SR        ; Enable 1K Cache
91          080000
92 P:000106  08F4BB   movep   #BCR_value,x:BCR    ; Set external wait states
93          002400
94 P:000108  08F4B7   movep   #AAR2_value,x:AAR2  ; Set Address Attribute Reg2
95          104A0D
96 P:00010A  08F4B6   movep   #AAR3_value,x:AAR3  ; Set Address Attribute Reg3
97          100909
98
99 P:00010C  05F420   move    #-1,m0             ; Set LINEAR addressing mode
100          FFFFFFFF
101 P:00010E  05F423   move    #-1,m3
102          FFFFFFFF
103
104          main
105          ;-----
106          ;--- fill P:memory with initial pattern ---
107          ;-----
108 P:000116  63F400   move    #PATT,r3          ; r3 points to Test Patterns
109          000138
110 P:000118  000000   nop
111 P:000119  000000   nop
112
113 P:00011A  07DB84   move    p:(r3)+,x0        ; Get the Write Pattern for P:MEM
114
115 P:00011B  70F400   move    #PMemSize,n0      ; Get memory size
116          008000
117 P:00011D  60F400   move    #PMemStart,r0     ; Get starting address for fill
118          100000
119
120 P:00011F  06D820   rep    n0                 ; Fill RAM with first pattern data
121 P:000120  075884   move    x0,p:(r0)+
122
123          ;-----
124          ;--- Check for expected pattern data in each RAM location ---
125          ;--- and then replace with a new data pattern. ---
126          ;--- ...This provides an address check. Since erroneous ---
127          ;--- ...addressing will cause the data to be written into ---
128          ;--- ...incorrect locations and this will be evident in ---
129          ;--- ...the next read pass. ---
130          ;-----
131 P:000121  063890   DOR    #PATTN,test_Pm     ; Start Pattern Test Loop
132          00000D
133 P:000123  60F400   move    #PMemStart,r0     ; Get starting address of Test Memory
134          100000
135 P:000125  200041   tfr    x0,a               ; Save the last test pattern -> a
136 P:000126  07DB84   move    P:(r3)+,x0        ; Get the next test pattern -> x0
137
138          ; Test this pattern through external RAM
139          ; Test all external RAM locations
140 P:000127  06D810   DOR    n0,next_loc
141          000006
142 P:000129  07E085   move    P:(r0),x1         ; Read RAM location
143 P:00012A  200065   cmp    x1,a               ; Read data = last test pattern?
144 P:00012B  052409   bne    <ERR               ; No, error if compare fails
145 P:00012C  075884   move    x0,P:(r0)+        ; Yes, Write next test pattern -> RAM
146 P:00012D  000000   nop
147          next_loc
148
```



## 32K x 8-bit Memory Based Designs

```
142 P:00012E 000000 nop ; Time to start next test pattern
143
144 test_Pm
145
146 ; One Pass Complete
147 ; All test patterns have been tried and passed in external RAM
148 P:00012F 518300 move x:MEM_PASS_COUNTER,b0
149 P:000130 000009 inc b ; Update pass loop counter
150 P:000131 000000 nop
151 P:000132 510300 move b0,x:MEM_PASS_COUNTER
152
153 P:000133 050FC3 bra main ; Do it all over again
154
155 ;-----
156 ; ERR--handles error messaging with user
157 ;
158 ; Expected Data --> a
159 ; Read Data --> x1
160 ; Address of failure --> r0
161
162 ERR
163 P:000134 600000 move r0,x:MEM_FAIL_ADDRESS ; Save off address of failure
164 P:000135 560100 move a,x:MEM_FAIL_WROTE ; Save off expected data
165 P:000136 450200 move x1,x:MEM_FAIL_READ ; Save off data read
166
167 P:000137 050C00 bra * ; Dynamically HALT here
168
169 ; Memory Test Patterns
170
171
172 P:000138 PATT dc $000000,$FFFFFF,$AAAAAA,$555555,$2BAD2C
173 P:00013D dc $800000,$400000,$200000,$100000
174 P:000141 dc $080000,$040000,$020000,$010000
175 P:000145 dc $008000,$004000,$002000,$001000
176 P:000149 dc $000800,$000400,$000200,$000100
177 P:00014D dc $000080,$000040,$000020,$000010
178 P:000151 dc $000008,$000004,$000002,$000001
179 P:000155 dc $7FFFFFF,$BFFFFFF,$DFFFFFF,$EFFFFFF
180 P:000159 dc $F7FFFF,$FBFFFF,$FDFFFF,$FEFFFF
181 P:00015D dc $FF7FFF,$FFBFFF,$FFDFFF,$FFEFFF
182 P:000161 dc $FFF7FF,$FFFBFF,$FFFDFF,$FFF7FF
183 P:000165 dc $FFFF7F,$FFFBFF,$FFFDFF,$FFF7FF
184 P:000169 dc $FFFFFF7,$FFFFFFB,$FFFFFFD,$FFFFFFE
185 P:00016D dc $FEDCBA,$123456,$012345,$EDCBA9
186
187 000038 PATTN equ *-PATT-1
188
189 end memtst

0 Errors
0 Warnings
```



### 3.3 32K × 24-bit ‘P’/‘X’/‘Y’ Fast SRAM Example

This section describes a 32K × 24-bit shared ‘P’/‘X’/‘Y’ memory space, asynchronous Fast SRAM implementation using Motorola’s MCM6206D device. **Figure 3-7** shows the memory map layout, **Figure 3-1** shows the block diagram, and **Example 3-2** shows the example code. A shared memory space means that data written to one address in one memory space can be accessed by the same address in another memory space, (e.g., writing \$012345 to P:\$100000 could be read by X:\$100000, Y:\$100000 or P:\$100000).

The DSP core runs at 80 MHz, and the input frequency source is a 4.000 MHz crystal. The asynchronous Fast SRAM requires an access time equal to or less than 12.4 nS to satisfy the DSP’s one wait state external memory requirements.

	Program	X Data	Y Data	
\$FFFFFF	Reserved	Internal I/O	Internal I/O	\$FFFFC0
		External	External	\$FFF000
\$FF00C0	Bootstrap ROM	Reserved	Reserved	\$FF0000
\$FF0000				
	External	External	External	
\$108000	Shared External 32K SRAM	Shared External 32K SRAM	Shared External 32K SRAM	
\$100000				
\$001000	External	External	External	
\$000C00	Internal 1K Cache			
	Internal 3K SRAM	Internal 2K SRAM	Internal 2K SRAM	\$000800
\$000000				\$000000

**Figure 3-6.** 32K × 24-bit ‘P’/‘X’/‘Y’ Fast SRAM Memory Map

#### 3.3.1 DSP56303 Port A Timing Requirements and Register Settings

For optimal use of the 32K × 24-bit ‘P’/‘X’/‘Y’ space memory configuration, set up the following DSP control registers.

You must set the core speed of the DSP for optimum processor and memory performance using the DSP PLL and Clock Generation (PCTL) register. For this example, the DSP core runs at 80 MHz and the input frequency source is a 4.000 MHz crystal. The PCTL register value combines the following bits for each feature:

- Desired Core Frequency = 80 MHz
- Given the External Frequency = 4.000 MHz
- Predivider value = 1, Bits 20–23 = \$0
- Low-power Divider value = 1, Bits 12–14 = \$0
- VCO Multiplication value = 20, Bits 0–11 = \$013
- Crystal less than 200 kHz, Bit 15 = 0
- Disable XTAL drive output, Bit 16 = 0
- PLL runs during STOP, Bit 17 = 1
- Enable PLL operation, Bit 18 = 1
- Disable core clock output, Bit 19 = 1

The value loaded into the PCTL register is \$0E0013.

Address Attribute Pin 3 enables, via Fast SRAM  $\bar{E}$ , external 32K SRAM bank accesses in the address range from \$100000 to \$107FFF during program space requests. Configure the memory address space requirements for the Address Attribute Pin 3 using Address Attribute Register 3 (AAR3). The AAR3 value combines the following bits for each feature:

- Specify the external memory access type as asynchronous SRAM, Bits 0–1 = \$1.
- Pull the AA pin high when selected, Bit 2 = 0.
- Activate the AA pin during external program space accesses, Bit 3 = 1.
- Activate the AA pin during external X data space accesses, Bit 4 = 1.
- Activate the AA pin during external Y data space accesses, Bit 5 = 1.
- Move the eight least significant bits of the address to the eight most significant bits of the external address bus, Bit 6 = 0.
- Enable the internal packing/unpacking logic during external DMA accesses, Bit 7 = 0.
- Specify the number of address bits to compare, Bits 8–11 = \$9
- Specify the most significant portion of the address to compare, Bits 12–23 = \$100

The value loaded into the AAR3 is \$100939.

---

Address Attribute Pin 2 selects, via Fast SRAM A14, address line A14 in the external 32K SRAM bank during accesses in the address range from \$104000 to \$107FFF during program space requests. Configure the memory address space requirements for the Address Attribute Pin 2 using the Address Attribute Register 2 (AAR2). The AAR2 value combines the following bits for each feature:

- Specify the external memory access type as asynchronous SRAM, Bits 0–1 = \$1.
- Pull the AA pin high when selected, AAR Bit 2 = 1.
- Activate the AA pin during external program space accesses, Bit 3 = 1.
- Activate the AA pin during external X data space accesses, Bit 4 = 1.
- Activate the AA pin during external Y data space accesses, Bit 5 = 1.
- Move the eight least significant bits of the address to the eight most significant bits of the external address bus, Bit 6 = 0.
- Enable the internal packing/unpacking logic during external DMA accesses, Bit 7 = 0.
- Specify the number of address bits to compare, Bits 8–11 = \$A
- Specify the most significant portion of the address to compare, Bits 12–23 = \$104

The value loaded into the AAR2 is \$104A3D.

The value loaded into AAR0 and AAR1 is \$000000.

Select the proper number of wait states for the memory configuration using the Bus Control Register (BCR). The BCR value combines the following bits for each feature:

- Address Attribute area 0 wait states, Bits 0–4 = \$0
- Address Attribute area 1 wait states, Bits 5–9 = \$0
- Address Attribute area 2 wait states, Bits 10–12 = \$1
- Address Attribute area 3 wait states, Bits 13–15 = \$1
- Default address area wait states, Bits 16–20 = \$0
- Bus state status, Bit 21 = 0
- Enable Bus Lock Hold, Bit 22 = 0
- Enable Bus Request Hold, Bit 23 = 0

The value loaded into the BCR is \$002400.

---

Configure the operating mode and external memory controls using the Operating Mode Register (OMR). The OMR value is combines the following bits for each feature:

- MA–MD bits specify the DSP operating mode, Bits 0–3 = \$0.
- External Bus Disable bit disables the external bus controller for power conservation when external memory is not used, Bit 4 = \$0.
- Memory Switch Mode bit reconfigures internal memory spaces Bit 7 = \$0.
- Transfer Acknowledge Synchronize Select bit selects the synchronization method for the Transfer Acknowledge (TA) pin, Bit 11 = \$0.
- Bus Release Timing bit selects between a fast and slow bus release of the  $\overline{BB}$  pin, Bit 12 = \$0.
- Address Attribute Priority Disable bit allows the Address Attribute pins, AA0–AA3, to be used in any combination, Bit 14 = \$1.
- All other OMR bits are selected for their defaults of \$000000.

The value loaded into the OMR is \$004000.

Configure the memory mode of the DSP using the Status Register (SR). The SR value combines the following bits for each feature:

- Sixteen-Bit Compatibility mode enables full compatibility to object code written for the DSP56000 family of DSPs, Bit 13 = \$0.
- Instruction Cache Enable bit enables the instruction cache controller and changes the last 1K of internal program memory into cache memory, Bit 19 = \$1.
- All other Status Register bits are selected for their defaults of \$000000.

The value loaded into the SR is \$080000, which is the value loaded during reset.

---

**Example 3-2. 32K × 24-bit 'P'/'X'/'Y' Space Fast SRAM Memory Exercise Program**

Motorola DSP56300 Assembler Version 6.0.1.6 97-01-25 08:05:18 asram2.asm

```
1          page      132,60,3,3,
2          ;
3          ;        ASRAM2.ASM - Simple program to test 32Kx24-bits of
4          ;        Program/X-Data/Y-Data memory using a DSP56303
5          ;
6          ;        The program uses Internal P:RAM to test External P/X/Y:RAM
7          ;        from $100000 - $107FFF @ 1w/s
8
9
10         100000    PMemStart    equ      $100000
11         108000    PMemEnd      equ      $108000
12         008000    PMemSize     equ      PMemEnd-PMemStart
13
14         ;--- Program Specific Storage Locations (X DATA SPACE)
15         MEM_FAIL_ADDRESS
16         000000    MEM_FAIL_WROTE equ    $000000
17
18         000001    MEM_FAIL_READ  equ    $000001
19
20         000002    MEM_FAIL_READ  equ    $000002
21
22         000003    MEM_PASS_COUNTER equ    $000003
23
24         ;--- DSP56303 Control Registers (X I/O SPACE)
25         FFFFFB    BCR            equ    $FFFFFB      ; Bus Control Register
26         FFFFFD    PCTL          equ    $FFFFFD      ; PLL Control Register
27         FFFFF6    AAR3         equ    $FFFFF6      ; Address Attribute Register #3
28         FFFFF7    AAR2         equ    $FFFFF7      ; Address Attribute Register #2
29
30         ;--- PCTL value = 0x0E0013
31         000000    prediv        equ    0           ; Pre-Divider = 1
32         000000    lowdiv        equ    0           ; Low Power Divider = 1
33         000013    pllmul        equ    19          ; VCO Mult = 20; (19+1)*4.00MHz=80.00MHz
34         000000    crystal       equ    0           ; No, Crystal not less than 200kHz
35         000000    disXTAL      equ    0           ; No, do not disable crystal use
36         020000    pllstop      equ    $020000     ; Yes, PLL runs during STOP
37         040000    enpll        equ    $040000     ; Yes, enable PLL operation
38         080000    disclk       equ    $080000     ; Yes, disable CORE clock output
39         0E0013    PCTL_value    equ    prediv+lowdiv+pllmul+crystal+disXTAL+pllstop+enpll+disclk
40
41         ;--- AAR3 value = 0x100939
42         000001    acctype3      equ    1           ; External Memory access type = 0x1
43         000000    aahigh3      equ    0           ; Enable AA3 pin to be low when selected
44         000008    aap3         equ    $8          ; Yes, Enable AA3 pin on ext 'P' accesses
45         000010    aax3         equ    $10         ; Yes, Enable AA3 pin on ext 'X' accesses
46         000020    aay3         equ    $20         ; Yes, Enable AA3 pin on ext 'Y' accesses
47         000000    aswap3       equ    0           ; No, Enable address bus swap
48         000000    enpack3      equ    0           ; No, Enable packing/unpacking logic
49         000900    nadd3        equ    $000900     ; Compare 9 address bits
50         100000    msadd3       equ    $100000     ; Most significant portion of address,
51         ; $100000 - $107fff, to compare.
52         ; (0001,0000,0xxx,xxxx,xxxx,xxxx)
53         100939    AAR3_value   equ    acctype3+aahigh3+aap3+aax3+aay3+aswap3+enpack3+nadd3+msadd3
54
55         ;--- AAR2 value = 0x104A3D
56         000001    acctype2      equ    1           ; External Memory access type = 0x1
57         000004    aahigh2      equ    $4          ; Enable AA2 pin to be high when selected
58         000008    aap2         equ    $8          ; Yes, Enable AA2 pin on ext 'P' accesses
59         000010    aax2         equ    $10         ; Yes, Enable AA2 pin on ext 'X' accesses
60         000020    aay2         equ    $20         ; Yes, Enable AA2 pin on ext 'Y' accesses
61         000000    aswap2       equ    0           ; No, Enable address bus swap
62         000000    enpack2      equ    0           ; No, Enable packing/unpacking logic
63         000A00    nadd2        equ    $000A00     ; Compare 10 address bits
64         104000    msadd2       equ    $104000     ; Most significant portion of address,
65         ; $104000 - $107fff, to compare.
66         ; (0001,0000,01xx,xxxx,xxxx,xxxx)
67         104A3D    AAR2_value   equ    acctype2+aahigh2+aap2+aax2+aay2+aswap2+enpack2+nadd2+msadd2
68
69         ;--- BCR value = 0x002400
70         000000    aaa0ws       equ    0           ; Address Attribute Area 0 w/s = 0
71         000000    aaalws       equ    0           ; Address Attribute Area 1 w/s = 0
72         000400    aaa2ws       equ    $000400     ; Address Attribute Area 2 w/s = 1
73         002000    aaa3ws       equ    $002000     ; Address Attribute Area 3 w/s = 1
```



## 32K x 8-bit Memory Based Designs

```
74      000000      defw     equ      0          ; Default Address Area w/s = 0
75      000000      busss     equ      0          ; Bus state status = 0
76      000000      enblh     equ      0          ; Enable Bus Lock Hold = 0
77      000000      enbrh     equ      0          ; Enable Bus Request Hold = 0
78      002400      BCR_value  equ      aaa0ws+aaa1ws+aaa2ws+aaa3ws+defws+busss+enblh+enbrh
79
80      ;-----
81      P:000100      org      p:$100      ;Keep the program in internal RAM
82
83      memtst
84
85      ; Initialization Section
86      P:000100      08F4BD      movep   #PCTL_value,x:PCTL; Set PLL Control Register
87      P:000102      05F43A      movec   #$004000,OMR          ; Disable Address Attribute Priority
88      P:000104      05F439      movec   #$080000,SR          ; Enable 1K Cache
89      P:000106      08F4BB      movep   #BCR_value,x:BCR    ; Set external wait states
90      P:000108      08F4B7      movep   #AAR2_value,x:AAR2  ; Set Address Attribute Reg2
91      P:00010A      08F4B6      movep   #AAR3_value,x:AAR3  ; Set Address Attribute Reg3
92
93      P:00010C      05F420      move    #-1,m0             ; Set LINEAR addressing mode
94      P:00010E      05F423      move    #-1,m3
95
96      P:000110      20001B      clrb
97      P:000111      000000      nop
98      P:000112      570000      move    b,x:MEM_FAIL_ADDRESS ; Initialize Failed Address -> $000000
99      P:000113      570100      move    b,x:MEM_FAIL_WROTE   ; Initialize Expected Data -> $000000
100     P:000114      570200      move    b,x:MEM_FAIL_READ     ; Initialize Data Read -> $000000
101     P:000115      570300      move    b,x:MEM_PASS_COUNTER  ; Initialize Pass Counter -> $000000
102
103     main
104     ;-----
105     ;--- fill P:memory with initial pattern ---
106     ;-----
107     P:000116      63F400      move    #PATT,r3           ; r3 points to Test Patterns
108     P:000118      000000      nop
109     P:000119      000000      nop
110
111     P:00011A      07DB84      move    p:(r3)+,x0         ; Get the Write Pattern for P:MEM
112
113     P:00011B      70F400      move    #PMemSize,n0       ; Get memory size
114     P:00011D      60F400      move    #PMemStart,r0      ; Get starting address for fill
115     P:00011F      06D820      rep     n0                  ; Fill RAM with first pattern data
116     P:000120      075884      move    x0,p:(r0)+
117
118     ;-----
119     ;--- Check for expected pattern data in each RAM location ---
120     ;--- and then replace with a new data pattern. ---
121     ;--- ...This provides an address check. Since erroneous ---
122     ;--- ...addressing will cause the data to be written into ---
123     ;--- ...incorrect locations and this will be evident in ---
124     ;--- ...the next read pass. ---
125     ;-----
126
127
128     P:000121      063890      DOR     #PATTN,test_Pm     ; Start Pattern Test Loop
129     P:000123      60F400      move    #PMemStart,r0      ; Get starting address of Test Memory
130     P:000125      200041      tfr     x0,a               ; Save the last test pattern -> a
131     P:000126      07DB84      move    P:(r3)+,x0         ; Get the next test pattern -> x0
132
133     ; Test this pattern through external RAM
134     P:000127      06D810      DOR     n0,next_loc        ; Test all external RAM locations
135     P:000129      07E085      move    P:(r0),x1          ; Read RAM location
136     P:00012A      200065      cmp     x1,a               ; Read data = last test pattern?
137     P:00012B      052409      bne     <ERR               ; No, error if compare fails
138     P:00012C      075884      move    x0,P:(r0)+         ; Yes, Write next test pattern -> RAM
139     P:00012D      000000      nop
140     next_loc
```

```

141
142     P:00012E 000000     nop
143                                     ; Time to start next test pattern
144
145         test_Pm
146
147     ;                                     One Pass Complete
148     ; All test patterns have been tried and passed in external RAM
148     P:00012F 518300     move    x:MEM_PASS_COUNTER,b0
149     P:000130 000009     inc     b                                     ; Update pass loop counter
150     P:000131 000000     nop
151     P:000132 510300     move    b0,x:MEM_PASS_COUNTER
152
153     P:000133 050FC3     bra     main                                     ; Do it all over again
154
155 ;-----
156                                     ; ERR -- handles error messaging with user
157                                     ;
158                                     ; Expected Data --> a
159                                     ; Read Data --> x1
160                                     ; Address of failure --> r0
161
162     ERR
163     P:000134 600000     move    r0,x:MEM_FAIL_ADDRESS ; Save off address of failure
164     P:000135 560100     move    a,x:MEM_FAIL_WROTE   ; Save off expected data
165     P:000136 450200     move    x1,x:MEM_FAIL_READ   ; Save off data read
166
167     P:000137 050C00     bra     *                                     ; Dynamically HALT here
168
169
170                                     ; Memory Test Patterns
171
172     P:000138           PATT    dc     $000000,$FFFFFF,$AAAAAA,$555555,$2BAD2C
173     P:00013D           dc     $800000,$400000,$200000,$100000
174     P:000141           dc     $080000,$040000,$020000,$010000
175     P:000145           dc     $008000,$004000,$002000,$001000
176     P:000149           dc     $000800,$000400,$000200,$000100
177     P:00014D           dc     $000080,$000040,$000020,$000010
178     P:000151           dc     $000008,$000004,$000002,$000001
179     P:000155           dc     $7FFFFFF,$BFFFFFF,$DFFFFFF,$EFFFFFF
180     P:000159           dc     $F7FFFF,$FBFFFF,$FDFFFF,$FEFFFF
181     P:00015D           dc     $FF7FFF,$FFBFFF,$FFDFFF,$FFEFFF
182     P:000161           dc     $FFF7FF,$FFFBFF,$FFFDFF,$FFF7FF
183     P:000165           dc     $FFFF7F,$FFFFBF,$FFFFDF,$FFFFEF
184     P:000169           dc     $FFFFF7,$FFFFFB,$FFFFFD,$FFFFFE
185     P:00016D           dc     $FEDCBA,$123456,$012345,$EDCBA9
186
187     000038           PATTN    equ    *-PATT-1
188
189                                     end    memtst

0     Errors
0     Warnings
    
```

### 3.4 16K × 24-bit ‘P’/‘X’ and 16K × 24-bit ‘Y’ Fast SRAM Example

This section describes a 16K × 24-bit shared ‘P’/‘X’ and 16K × 24-bit ‘Y’ memory space, asynchronous Fast SRAM implementation using Motorola’s MCM6206D device. **Figure 3-9** shows the memory map layout; **Figure 3-1** shows the block diagram; and **Example 3-3** shows the example code.

The DSP core runs at 80 MHz and the input frequency source is a 4.000 MHz crystal. The asynchronous Fast SRAM requires an access time equal to or less than 12.4 nS to satisfy the DSP’s one wait state external memory requirements.

	Program	X Data	Y Data	
\$FFFFFFF	Reserved	Internal I/O	Internal I/O	\$FFFFC0
		External	External	\$FFF000
\$FF00C0	Bootstrap ROM	Reserved	Reserved	\$FF0000
\$FF0000				
	External	External	External	
\$104000	Shared External 16K SRAM	Shared External 16K SRAM	External 16K SRAM	
\$100000				
\$001000	External	External	External	
\$000C00	Internal 1K Cache			
\$000000	Internal 3K SRAM	Internal 2K SRAM	Internal 2K SRAM	\$000800
				\$000000

**Figure 3-7.** 16K × 24-bit ‘P’/‘X’ and 16K × 24-bit ‘Y’ Memory Map

#### 3.4.1 DSP56303 Port A Timing Requirements and Register Settings

For optimal use of the 16K × 24-bit ‘P’/‘X’ and 16K × 24-bit ‘Y’ space memory configuration, set up the following DSP control registers.



Set the core speed of the DSP for optimum processor and memory performance using the DSP PLL and Clock Generation register (PCTL). For this example, the DSP core runs at 80 MHz and the input frequency source is a 4.000 MHz crystal. The PCTL register value combines the following bits for each feature:

- Desired Core Frequency = 80 MHz
- External Frequency = 4.000 MHz
- Predivider value = 1, Bits 20–23 = \$0
- Low-power Divider value = 1, Bits 12–14 = \$0
- VCO Multiplication value = 20, Bits 0–11 = \$013
- Crystal less than 200 kHz, Bit 15 = 0
- Disable XTAL drive output, Bit 16 = 0
- PLL runs during STOP, Bit 17 = 1
- Enable PLL operation, Bit 18 = 1
- Disable core clock output, Bit 19 = 1

The value loaded into the PCTL register is \$0E0013.

Address Attribute Pin 3 enables, via Fast SRAM  $\bar{E}$ , external 32K SRAM bank accesses in the address range from \$100000 to \$103FFF during program, X data and Y data space requests. Configure the memory address space requirements for the Address Attribute Pin 3 using Address Attribute Register 3 (AAR3). The AAR3 value combines the following bits for each feature:

- Specify the external memory access type as asynchronous SRAM, bits 0–1 = \$1.
- Pull the AA pin high when selected, Bit 2 = 0.
- Activate the AA pin during external program space accesses, Bit 3 = 1.
- Activate the AA pin during external X data space accesses, Bit 4 = 1.
- Activate the AA pin during external Y data space accesses, Bit 5 = 1.
- Move the eight least significant bits of the address to the eight most significant bits of the external address bus, Bit 6 = 0.
- Enable the internal packing/unpacking logic during external DMA accesses, Bit 7 = 0.
- Specify the number of address bits to compare, Bits 8–11 = \$A
- Specify the most significant portion of the address to compare, Bits 12–23 = \$100

The value loaded into the AAR3 is \$100A39.

---

Address Attribute Pin 2 selects, via Fast SRAM A14, address line A14 in the external 32K SRAM bank during accesses in the address range from \$100000 to \$103FFF to differentiate program/X data space requests from Y data space requests. Configure the memory address space requirements for the Address Attribute Pin 2 using the Address Attribute Register 2 (AAR2). The AAR2 value combines the following bits for each feature:

- Specify the external memory access type as asynchronous SRAM, Bits 0–1 = \$1.
- Pull the AA pin high when selected, Bit 2 = 1.
- Activate the AA pin during external program space accesses, Bit 3 = 1.
- Activate the AA pin during external X data space accesses, Bit 4 = 1.
- Activate the AA pin during external Y data space accesses, Bit 5 = 0.
- Move the eight least significant bits of the address to the eight most significant bits of the external address bus, Bit 6 = 0.
- Enable the internal packing/unpacking logic during external DMA accesses, Bit 7 = 0.
- Specify the number of address bits to compare, Bits 8–11 = \$A
- Specify the most significant portion of the address to compare, Bits 12–23 = \$100

The value loaded into the AAR2 is \$100A1D.

The value loaded into AAR0 and AAR1 is \$000000.

Select the proper number of wait states for the memory configuration using the Bus Control Register (BCR). The BCR value combines the following bits for each feature:

- Address Attribute area 0 wait states, Bits 0–4 = \$0
- Address Attribute area 1 wait states, Bits 5–9 = \$0
- Address Attribute area 2 wait states, Bits 0–12] = \$1
- Address Attribute area 3 wait states, Bits 13–15 = \$1
- Default address area wait states, Bits 16–20 = \$0
- Bus state status, Bit 21 = 0
- Enable Bus Lock Hold, Bit 22 = 0
- Enable Bus Request Hold, Bit 23 = 0

The value loaded into the BCR register is \$002400

---

Configure the operating mode and external memory controls using the Operating Mode Register (OMR). The OMR value combines the following bits for each feature:

- MA–MD bits specify the DSP operating mode, Bits 0–3 = \$0.
- External Bus Disable bit disables the external bus controller for power conservation when external memory is not used, Bit 4 = \$0.
- Memory Switch Mode bit reconfigures internal memory spaces, Bit 7 = \$0.
- Transfer Acknowledge Synchronize Select bit selects the synchronization method for the Transfer Acknowledge (TA) pin, Bit 11 = \$0.
- Bus Release Timing bit selects between a fast and slow bus release of the  $\overline{BB}$  pin, Bit 12 = \$0.
- Address Attribute Priority Disable bit allows the Address Attribute pins, AA0–AA3, to be used in any combination, Bit 14. = \$1.
- All other OMR bits are selected for their defaults of \$000000.

The value loaded into the OMR is \$004000.

Configure the memory mode of the DSP using the Status Register (SR). The SR value combines the following bits for each feature:

- Sixteen-Bit Compatibility mode enables full compatibility to object code written for the DSP56000 family of DSPs, Bit 13 = \$0.
- Instruction Cache Enable bit enables the instruction cache controller and changes the last 1K of internal program memory into cache memory, Bit 19 = \$1.
- All other Status Register bits are selected for their defaults of \$000000.

The value loaded into the SR is \$080000, which is the value loaded during reset.

---

**Example 3-3. 16K 'P'/'X' and 16K 'Y' Space Fast SRAM Memory Exercise Program**

Motorola DSP56300 Assembler Version 6.0.1.6 97-01-25 08:05:22 asram3.asm

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
```

```

                page      132,60,3,3,
;
;   ASRAM3.ASM - Simple program to test 16Kx24-bits of Program/X-Data
;               and 16Kx24-bits of Y-Data memory using a DSP56303
;
;   The program uses Internal P:RAM to test External P/X & Y:RAM
;               from $100000 - $103FFF @ 1w/s
;
PMemStart      equ      $100000
PMemEnd       equ      $104000
PMemSize      equ      PMemEnd-PMemStart

;--- Program Specific Storage Locations (X DATA SPACE)
P_MEM_FAIL_ADDRESS
000000        equ      $000000
P_MEM_FAIL_WROTE
000001        equ      $000001
P_MEM_FAIL_READ
000002        equ      $000002

Y_MEM_FAIL_ADDRESS
000003        equ      $000003
Y_MEM_FAIL_WROTE
000004        equ      $000004
Y_MEM_FAIL_READ
000005        equ      $000005

MEM_PASS_COUNTER
000006        equ      $000006

;--- DSP56303 Control Registers (X I/O SPACE)
FFFFFB       BCR          equ      $FFFFFB          ; Bus Control Register
FFFFFD       PCTL        equ      $FFFFFD          ; PLL Control Register
FFFFF6       AAR3        equ      $FFFFF6          ; Address Attribute Register #3
FFFFF7       AAR2        equ      $FFFFF7          ; Address Attribute Register #2

;--- PCTL value = 0x0E0013
000000       prediv      equ      0                ; Pre-Divider = 1
000000       lowdiv      equ      0                ; Low Power Divider = 1
000013       pllmul      equ      19              ; VCO Mult = 20; (19+1)*4.00MHz=80.00MHz
000000       crystal     equ      0                ; No, Crystal not less than 200kHz
000000       disXTAL     equ      0                ; No, do not disable crystal use
020000       pllstop     equ      $020000         ; Yes, PLL runs during STOP
040000       enpll       equ      $040000         ; Yes, enable PLL operation
080000       disclk      equ      $080000         ; Yes, disable CORE clock output
0E0013       PCTL_value  equ      prediv+lowdiv+pllmul+crystal+disXTAL+pllstop+enpll+disclk

;--- AAR3 value = 0x100A39
000001       acctype3    equ      1                ; External Memory access type = 0x1
000000       aahigh3     equ      0                ; Enable AA3 pin to be low when selected
000008       aap3        equ      $8                ; Yes, Enable AA3 pin on ext 'P' accesses
000010       aax3        equ      $10              ; Yes, Enable AA3 pin on ext 'X' accesses
000020       aay3        equ      $20              ; Yes, Enable AA3 pin on ext 'Y' accesses
000000       aswap3      equ      0                ; No, Enable address bus swap
000000       enpack3     equ      0                ; No, Enable packing/unpacking logic
000A00       nadd3       equ      $000A00         ; Compare 10 address bits
100000       msadd3      equ      $100000         ; Most significant portion of address,
; $100000 - $103fff, to compare.
; (0001,0000,00xx,xxxx,xxxx,xxxx)
100A39       AAR3_value  equ      acctype3+aahigh3+aap3+aax3+aay3+aswap3+enpack3+nadd3+msadd3

;--- AAR2 value = 0x100A1D
000001       acctype2    equ      1                ; External Memory access type = 0x1
000004       aahigh2     equ      $4                ; Enable AA2 pin to be high when selected
000008       aap2        equ      $8                ; Yes, Enable AA2 pin on ext 'P' accesses
000010       aax2        equ      $10              ; Yes, Enable AA2 pin on ext 'X' accesses
000000       aay2        equ      0                ; No, Enable AA2 pin on ext 'Y' accesses
000000       aswap2      equ      0                ; No, Enable address bus swap
000000       enpack2     equ      0                ; No, Enable packing/unpacking logic
000A00       nadd2       equ      $000A00         ; Compare 10 address bits
100000       msadd2      equ      $100000         ; Most significant portion of address,
; $100000 - $103fff, to compare.
; (0001,0000,00xx,xxxx,xxxx,xxxx)
```



```
75 100A1D AAR2_value equ acctype2+aaahigh2+aap2+aax2+aay2+aswap2+enpack2+nadd2+msadd2
76
77 ;--- BCR value = 0x002400
78 000000 aaa0ws equ 0 ; Address Attribute Area 0 w/s = 0
79 000000 aaalws equ 0 ; Address Attribute Area 1 w/s = 0
80 000400 aaa2ws equ $000400 ; Address Attribute Area 2 w/s = 1
81 002000 aaa3ws equ $002000 ; Address Attribute Area 3 w/s = 1
82 000000 defws equ 0 ; Default Address Area w/s = 0
83 000000 bussss equ 0 ; Bus state status = 0
84 000000 enblh equ 0 ; Enable Bus Lock Hold = 0
85 000000 enbrh equ 0 ; Enable Bus Request Hold = 0
86 002400 BCR_value equ aaa0ws+aaalws+aaa2ws+aaa3ws+defws+bussss+enblh+enbrh
87
88 ;-----
89 P:000100 org p:$100 ;Keep the program in internal RAM
90
91 memtst
92
93 ; Initialization Section
94 P:000100 08F4BD movep #PCTL_value,x:PCTL ; Set PLL Control Register
95 0E0013
96 P:000102 05F43A movec #$004000,OMR ; Disable Address Attribute Priority
97 004000
98 P:000104 05F439 movec #$080000,SR ; Enable 1K Cache
99 080000
100 P:000106 08F4BB movep #BCR_value,x:BCR ; Set external wait states
101 002400
102 P:000108 08F4B7 movep #AAR2_value,x:AAR2 ; Set Address Attribute Reg2
103 100A1D
104 P:00010A 08F4B6 movep AAR3_value,x:AAR3 ; Set Address Attribute Reg3
105 100A39
106
107 P:00010C 05F420 move #-1,m0 ; Set LINEAR addressing mode
108 FFFFFFFF
109 P:00010E 05F423 move #-1,m3
110 FFFFFFFF
111
112 P:000110 20001B clr b
113
114 P:000111 000000 nop
115 P:000112 570000 move b,x:P_MEM_FAIL_ADDRESS ; Initialize P:Failed Address -> $000000
116 P:000113 570100 move b,x:P_MEM_FAIL_WROTE ; Initialize P:Expected Data -> $000000
117 P:000114 570200 move b,x:P_MEM_FAIL_READ ; Initialize P:Data Read -> $000000
118 P:000115 570300 move b,x:Y_MEM_FAIL_ADDRESS ; Initialize Y:Failed Address -> $000000
119 P:000116 570400 move b,x:Y_MEM_FAIL_WROTE ; Initialize Y:Expected Data -> $000000
120 P:000117 570500 move b,x:Y_MEM_FAIL_READ ; Initialize Y:Data Read -> $000000
121 P:000118 570600 move b,x:MEM_PASS_COUNTER ; Initialize Pass Counter -> $000000
122
123 main
124
125 ;-----
126 ;--- fill P:memory with initial pattern ---
127
128 ;-----
129 P:000119 63F400 move #PATT,r3 ; r3 points to Test Patterns
130 00014A
131 P:00011B 000000 nop
132 P:00011C 000000 nop
133
134 P:00011D 07DB84 move p:(r3)+,x0 ; Get the Write Pattern for P/X:MEM
135 P:00011E 07DB86 move p:(r3)+,y0 ; Get the Write Pattern for Y:MEM
136
137 P:00011F 70F400 move #PMemSize,n0 ; Get memory size
138 004000
139
140 P:000121 60F400 move #PMemStart,r0 ; Get starting address for fill
141 100000
142
143 P:000123 06D820 rep n0 ; Fill P/X:RAM with first data pattern
144 P:000124 075884 move x0,p:(r0)+
145
146 P:000125 60F400 move #PMemStart,r0
147 100000
148 P:000127 06D820 rep n0 ; Fill Y:RAM with first data pattern
149 P:000128 4E5800 move y0,y:(r0)+
150
151 ;-----
152 ;--- Check for expected pattern data in each RAM location ---
153 ;--- and then replace with a new data pattern. ---
154 ;--- ...This provides an address check. Since erroneous ---
155 ;--- ...addressing will cause the data to be written into ---
156 ;--- ...incorrect locations and this will be evident in ---
```



## 32K x 8-bit Memory Based Designs

```
142 ;--- ...the next read pass. ---
143 ;-----
144 P:000129 063890 DOR #PATTN,test_Pm ; Start Pattern Test Loop
145 P:00012B 60F400 000013 move #PMemStart,r0 ; Get starting address of Test Memory
146 P:00012D 200041 100000 tfr x0,a ; Save the last P/X test pattern -> a
147 P:00012E 200059 60F400 tfr y0,b ; Save the last Y test pattern -> b
148 P:00012F 07DB84 move P:(r3)+,x0 ; Get the next P/X test pattern -> x0
149 P:000130 07DB86 move P:(r3)+,y0 ; Get the next Y test pattern -> y0
150
151 ; Test this pattern through external RAM
152 P:000131 06D810 DOR n0,next_loc ; Test all external RAM locations
153 P:000133 07E085 move P:(r0),x1 ; Read P/X:RAM location
154 P:000134 200065 cmp x1,a ; Read data = last test pattern?
155 P:000135 05240D bne <PERR ; No, error if compare fails
156 P:000136 076084 move x0,P:(r0) ; Yes, Write next test pattern ->
P/X:RAM
157
158 P:000137 4FE000 move Y:(r0),y1 ; Read Y:RAM location
159 P:000138 20007D cmp y1,b ; Read data = last test pattern?
160 P:000139 05240D bne <YERR ; No, error if compare fails
161 P:00013A 4E5800 move y0,Y:(r0)+ ; Yes, Write next test pattern ->
Y:RAM
162 P:00013B 000000 nop
163 next_loc
164
165 P:00013C 000000 nop ; Time to start next test pattern
166
167 test_Pm
168
169 ; One Pass Complete
170 ; All test patterns have been tried and passed in external RAM
171 P:00013D 518600 move x:MEM_PASS_COUNTER,b0
172 P:00013E 000009 inc b ; Update pass loop counter
173 P:00013F 000000 nop
174 P:000140 510600 move b0,x:MEM_PASS_COUNTER
175
176 P:000141 050F98 bra main ; Do it all over again
177
178 ;-----
179 ; PERR -- handles P/X:RAM error messaging with user
180 ;
181 ; Expected Data --> a
182 ; Read Data --> x1
183 ; Address of failure --> r0
184
185 PERR
186 P:000142 600000 move r0,x:P_MEM_FAIL_ADDRESS ; Save off address of failure
187 P:000143 560100 move a,x:P_MEM_FAIL_WROTE ; Save off expected data
188 P:000144 450200 move x1,x:P_MEM_FAIL_READ ; Save off data read
189
190 P:000145 050C00 bra * ; Dynamically HALT here
191
192 ;-----
193 ; YERR -- handles Y:RAM error messaging with user
194 ;
195 ; Expected Data --> b
196 ; Read Data --> y1
197 ; Address of failure --> r0
198
199 YERR
200 P:000146 600300 move r0,x:Y_MEM_FAIL_ADDRESS ; Save off address of failure
201 P:000147 570400 move b,x:Y_MEM_FAIL_WROTE ; Save off expected data
202 P:000148 470500 move y1,x:Y_MEM_FAIL_READ ; Save off data read
203
204 P:000149 050C00 bra * ; Dynamically HALT here
205
206
207
208 ; Memory Test Patterns
209
210 P:00014A PATT dc $000000,$FFFFFF,$AAAAAA,$555555,$2BAD2C
211 P:00014F dc $800000,$400000,$200000,$100000
212 P:000153 dc $080000,$040000,$020000,$010000
213 P:000157 dc $008000,$004000,$002000,$001000
214 P:00015B dc $000800,$000400,$000200,$000100
215 P:00015F dc $000080,$000040,$000020,$000010
216 P:000163 dc $000008,$000004,$000002,$000001
217 P:000167 dc $7FFFFFF,$BFFFFFF,$DFFFFFF,$EFFFFFF
```



```
218 P:00016B dc $F7FFFF,$FBFFFF,$FDFFFF,$FEFFFF
219 P:00016F dc $FF7FFF,$FFBFFF,$FFDFFF,$FFEFFF
220 P:000173 dc $FFF7FF,$FFFBFF,$FFFDFF,$FFFEFF
221 P:000177 dc $FFFF7F,$FFFFBF,$FFFFDF,$FFFFEF
222 P:00017B dc $FFFFF7,$FFFFFB,$FFFFFD,$FFFFFE
223 P:00017F dc $FEDCBA,$123456,$012345,$EDCBA9
224
225 P:000183 dc $000000,$FFFFFF,$AAAAAA,$555555,$2BAD2C
226 P:000188 dc $800000,$400000,$200000,$100000
227 P:00018C dc $080000,$040000,$020000,$010000
228 P:000190 dc $008000,$004000,$002000,$001000
229 P:000194 dc $000800,$000400,$000200,$000100
230 P:000198 dc $000080,$000040,$000020,$000010
231 P:00019C dc $000008,$000004,$000002,$000001
232 P:0001A0 dc $7FFFFFF,$BFFFFFF,$DFFFFFF,$EFFFFFF
233 P:0001A4 dc $F7FFFF,$FBFFFF,$FDFFFF,$FEFFFF
234 P:0001A8 dc $FF7FFF,$FFBFFF,$FFDFFF,$FFEFFF
235 P:0001AC dc $FFF7FF,$FFFBFF,$FFFDFF,$FFFEFF
236 P:0001B0 dc $FFFF7F,$FFFFBF,$FFFFDF,$FFFFEF
237 P:0001B4 dc $FFFFF7,$FFFFFB,$FFFFFD,$FFFFFE
238 P:0001B8 dc $FEDCBA,$123456,$012345,$EDCBA9
239
240 000038 PATN equ (( *-PATT)/2)-1
241
242 end memtst

0 Errors
0 Warnings
```





## 4 128K × 8-bit Memory Based Designs

This section describes how to implement several different DSP memory space Fast SRAM designs using three 128K × 8-bit 3.3 V memories with a Motorola DSP56303 device.

Using one hardware memory design based on three 128K × 8-bit 3.3 V memories, the DSP56303's Memory Expansion Port allows the 128K × 24-bit memory bank to be logically configured for use in various memory space arrangements. The configuration is accomplished by programmatically changing the Memory Expansion Port's Address Attribute Control Registers.

Configuring and using one Memory Expansion Port Address Attribute Control Register, the 128K memory bank can accommodate seven different memory space arrangements.

1. 128K × 24-bit 'P' Space Fast SRAM
2. 128K × 24-bit 'X' Space Fast SRAM
3. 128K × 24-bit 'Y' Space Fast SRAM
4. 128K × 24-bit 'P'/'X' Space Fast SRAM
5. 128K × 24-bit 'P'/'Y' Space Fast SRAM
6. 128K × 24-bit 'X'/'Y' Space Fast SRAM
7. 128K × 24-bit 'P'/'X'/'Y' Space Fast SRAM

Configuring and using two Memory Expansion Port Address Attribute Control Registers, the 128K memory bank can accommodate thirteen different memory space arrangements.

1. 128K × 24-bit 'P' Space Fast SRAM
2. 128K × 24-bit 'X' Space Fast SRAM
3. 128K × 24-bit 'Y' Space Fast SRAM
4. 128K × 24-bit 'P'/'X' Space Fast SRAM
5. 128K × 24-bit 'P'/'Y' Space Fast SRAM
6. 128K × 24-bit 'X'/'Y' Space Fast SRAM
7. 128K × 24-bit 'P'/'X'/'Y' Space Fast SRAM
8. 64K × 24-bit 'P'/'X' and 64K × 24-bit 'Y' Space Fast SRAM
9. 64K × 24-bit 'P'/'Y' and 64K × 24-bit 'X' Space Fast SRAM
10. 64K × 24-bit 'P' and 64K × 24-bit 'X'/'Y' Space Fast SRAM
11. 64K × 24-bit 'P' and 64K × 24-bit 'X' Space Fast SRAM
12. 64K × 24-bit 'P' and 64K × 24-bit 'Y' Space Fast SRAM
13. 64K × 24-bit 'X' and 64K × 24-bit 'Y' Space Fast SRAM

All of these memory space configurations efficiently use the full capacity of the memory chips in the 128K × 24-bit memory bank.

---

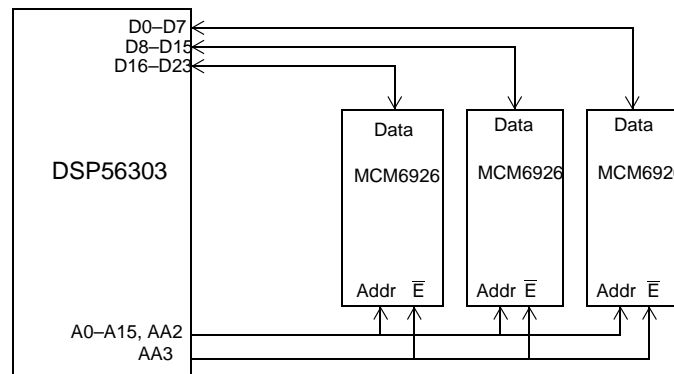
The memory configuration examples in the remainder of this chapter are based on one common hardware design that uses two Address Attribute Selectors and implements two of the most common configurations: 128K × 24-bit ‘P’/‘X’/‘Y’ Space Fast SRAM configuration and a 64K × 24-bit ‘X’ and 64K × 24-bit ‘Y’ Space Fast SRAM configuration (see **Figure 4-4** for a schematic of the hardware design).

## 4.1 128K × 24-bit Common Fast SRAM Hardware Design

This section describes an asynchronous Fast SRAM 128K × 24-bit memory bank implementation using Motorola’s MCM6926 device (see **Figure 4-1**). Memory bank designs of this size and type allow for future application expandability in both size and speed. The 3.3 V devices also require less hardware, providing a glueless memory interface with the DSP.

The memory bank design is implemented using two Address Attribute Selectors. However, for the 128K × 24-bit ‘P’/‘X’/‘Y’ Space Fast SRAM configuration a one Address Attribute Selector design can be used—A16 substituted for AA2. The two Address Attribute Selector design demonstrates the ultimate flexibility of the Address Attribute Selectors.

For this common hardware design, the DSP core runs at a maximum of 80 MHz, and the input frequency source is a 4.000 MHz crystal. The asynchronous Fast SRAM requires an access time equal to or less than 12.4 nS to satisfy the DSP’s one-wait state external memory requirements. This 3.3 V device is organized as 128K × 8-bits. Therefore, three memory devices are used to achieve the 24-bit wide bus.



**Figure 4-1** 128K × 24-bit Fast SRAM Memory Example

### 4.1.1 MCM6926-12 Memory Timing Requirements

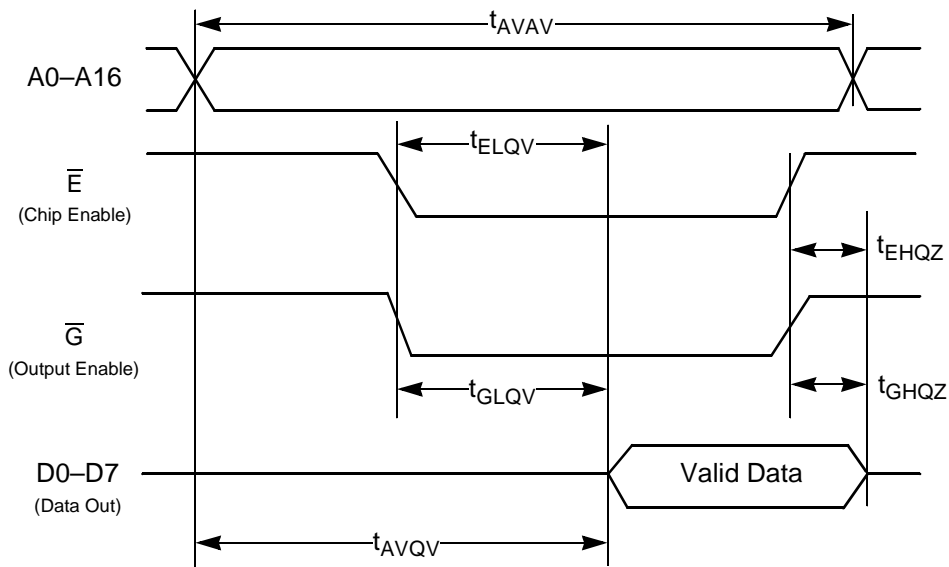
For the asynchronous Fast SRAM device to work properly, its timing requirements must be met. The sections that follow give the timing requirements for the MCM6926-12 128K × 8-bit 12 nS Fast SRAM.

#### 4.1.1.1 Read Cycle Timing

**Table 4-1** shows the memory read timing values used in the memory read cycle timing diagram, **Figure 4-2**.

**Table 4-1.** MCM6926-12 Memory Read Timing Specifications

Read Cycle Parameter	Symbol	Min	Max
Read Cycle Time	$t_{AVAV}$	12 nS	—
Address Access Time	$t_{AVQV}$	—	12 nS
Enable Access Time	$t_{ELQV}$	—	12 nS
Output Enable Access Time	$t_{GLQV}$	—	6 nS
Enable High to Output High-Z	$t_{EHQZ}$	0 nS	6 nS
Output Enable High to Output High-Z	$t_{GHQZ}$	0 nS	6 nS

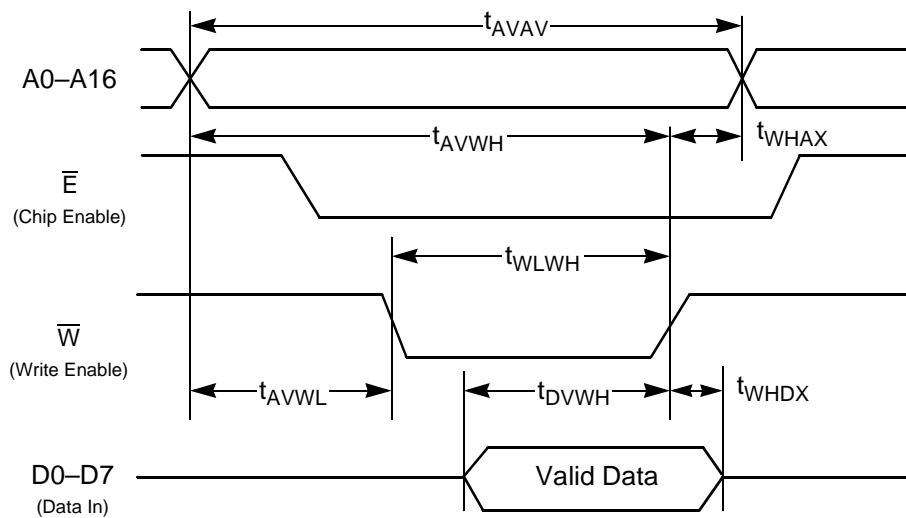

**Figure 4-2.** MCM6926 Memory Read Cycle Timing Diagram

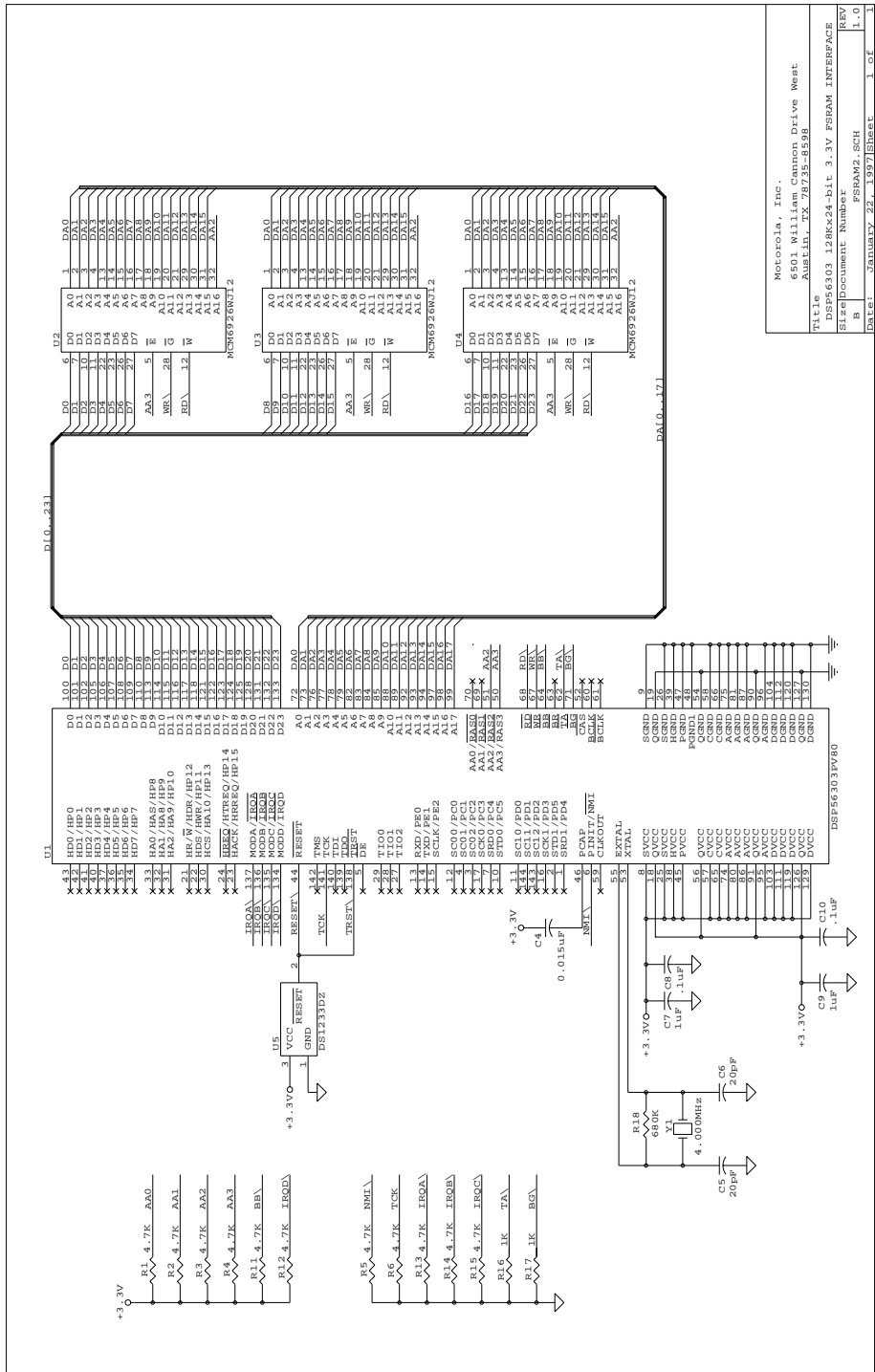
#### 4.1.1.2 Write Cycle Timing

**Table 4-2** shows the memory write timing values used in the memory write cycle timing diagram, **Figure 4-3**.

**Table 4-2.** MCM6926-12 Memory Write Timing Specifications

Write Cycle Parameter	Symbol	Min	Max
Write Cycle Time	$t_{AVAV}$	12 nS	—
Address Setup Time	$t_{AVWL}$	0 nS	—
Address Valid to End of Write	$t_{AVWH}$	10 nS	—
Write Pulse Width	$t_{WLWH}$	10 nS	—
Data Valid to End of Write	$t_{DVWH}$	6 nS	—
Data Hold Time	$t_{WHDX}$	0 nS	—


**Figure 4-3.** MCM6926 Memory Write Cycle Timing Diagram



## 4.2 128K × 24-bit ‘P’/‘X’/‘Y’ Fast SRAM Example

This section describes a 128K × 24-bit shared ‘P’/‘X’/‘Y’ memory space, asynchronous Fast SRAM implementation using Motorola’s MCM6926 device (see **Figure 4-5** for the memory map layout, **Figure 4-1** for the block diagram, and **Example 4-1** for the example code). In a shared memory space, data written to one address in one memory space can be accessed by the same address in another memory space (e.g., writing \$012345 to P:\$100000 could be read by X:\$100000, Y:\$100000 or P:\$100000).

The DSP core runs at 80 MHz, and the input frequency source is a 4.000 MHz crystal. The asynchronous Fast SRAM requires an access time equal to or less than 12.4 nS to satisfy the DSP’s one wait state external memory requirements.

	Program	X Data	Y Data	
\$FFFFFF	Reserved	Internal I/O	Internal I/O	\$FFFFC0
		External	External	\$FFF000
\$FF00C0	Bootstrap ROM	Reserved	Reserved	\$FF0000
\$FF0000				
	External	External	External	
\$120000	Shared External 128K SRAM	Shared External 128K SRAM	Shared External 128K SRAM	
\$100000				
\$001000	External	External	External	
\$000C00	Internal 1K Cache			
\$000000	Internal 3K SRAM	Internal 2K SRAM	Internal 2K SRAM	\$000800
				\$000000

**Figure 4-5.** 128K × 24-bit ‘P’/‘X’/‘Y’ Fast SRAM Memory Map

### 4.2.1 DSP56303 Port A Timing Requirements and Register Settings

For optimal use of the 128K × 24-bit ‘P’/‘X’/‘Y’ space memory configuration, set up the following DSP control registers.

Set the core speed of the DSP for optimum processor and memory performance using the DSP PLL and Clock Generation register (PCTL). For this example, the DSP core runs at 80 MHz, and the input frequency source is a 4.000 MHz crystal. The PCTL register value combines the following bits for each feature:

- Desired Core Frequency = 80 MHz
- Given the External Frequency = 4.000 MHz
- Predivider value = 1, Bits 20–23 = \$0
- Low-power Divider value = 1, Bits 12–14 = \$0
- VCO Multiplication value = 20, Bits 0–11 = \$013
- Crystal less than 200 kHz, Bit 15 = 0
- Disable XTAL drive output, Bit 16 = 0
- PLL runs during STOP, Bit 17 = 1
- Enable PLL operation, Bit 18 = 1
- Disable core clock output, Bit 19 = 1

The value loaded into the PCTL register is \$0E0013.

Address Attribute Pin 3 enables, via Fast SRAM  $\bar{E}$ , the external 128K SRAM bank accesses in the address range from \$100000 to \$11FFFF during program, X data, and Y data space requests. Configure the memory address space requirements for Address Attribute Pin 3 using Address Attribute Register 3 (AAR3). The AAR3 value combines the following bits for each feature:

- Specify the external memory access type as asynchronous SRAM, Bits 0–1 = \$1.
- Pull the AA pin high when selected, Bit 2 = 0.
- Activate the AA pin during external program space accesses, Bit 3 = 1.
- Activate the AA pin during external X data space accesses, Bit 4 = 1.
- Activate the AA pin during external Y data space accesses, Bit 5 = 1.
- Move the eight least significant bits of the address to the eight most significant bits of the external address bus, Bit 6 = 0.
- Enable the internal packing/unpacking logic during external DMA accesses, Bit 7 = 0.
- Specify the number of address bits to compare, Bits 8–11 = \$7
- Specify the most significant portion of the address to compare, Bits 12–23 = \$100

The value loaded into the AAR3 is \$100739.

---

Address Attribute Pin 2 selects, via Fast SRAM A16, address line A16 in the external 128K SRAM bank during accesses in the address range from \$110000 to \$11FFFF during program, X data, and Y data space requests. Configure the memory address space requirements for the Address Attribute Pin 2 using the Address Attribute Register 2 (AAR2). The AAR2 value combines the following bits for each feature:

- Specify the external memory access type as asynchronous SRAM, Bits 0–1 = \$1.
- Pull the AA pin high when selected, Bit 2 = 1.
- Activate the AA pin during external program space accesses, Bit 3 = 1.
- Activate the AA pin during external X data space accesses, Bit 4 = 1.
- Activate the AA pin during external Y data space accesses, Bit 5 = 1.
- Move the eight least significant bits of the address to the eight most significant bits of the external address bus, Bit 6 = 0.
- Enable the internal packing/unpacking logic during external DMA accesses, Bit 7 = 0.
- Specify the number of address bits to compare, Bits 8–11 = \$8
- Specify the most significant portion of the address to compare, Bits 12–23 = \$110

The value loaded into the AAR2 is \$11083D.

The value loaded into AAR0 and AAR1 is \$000000.

Select the proper number of wait states for the memory configuration using the Bus Control Register (BCR). The BCR value combines the following bits for each feature:

- Address attribute area 0 wait states, Bits 0–4 = \$0
- Address attribute area 1 wait states, Bits 5–9 = \$0
- Address attribute area 2 wait states, Bits 10–12 = \$1
- Address attribute area 3 wait states, Bits 13–15 = \$1
- Default address area wait states, Bits 16–20 = \$0
- Bus state status, Bit 21 = 0
- Enable Bus Lock Hold, Bit 22 = 0
- Enable Bus Request Hold, Bit 23 = 0

The value loaded into the BCR is \$002400.

---



Configure the operating mode and external memory controls using the Operating Mode Register (OMR). The OMR value combines the following bits for each feature:

- MA–MD bits specify the DSP operating mode, Bits 0–3 = \$0.
- External Bus Disable bit disables the external bus controller for power conservation when external memory is not used, Bit 4 = \$0.
- Memory Switch Mode bit, reconfigures internal memory spaces, Bit 7 = \$0.
- Transfer Acknowledge Synchronize Select bit selects the synchronization method for the Transfer Acknowledge (TA) pin, Bit 11 = \$0.
- Bus Release Timing bit selects between a fast and slow bus release of the  $\overline{BB}$  pin, Bit 12 = \$0.
- Address Attribute Priority Disable bit allows the Address Attribute pins, AA0–AA3, to be used in any combination, Bit 14. = \$1.
- All other OMR bits are selected for their defaults of \$000000.

The value loaded into the OMR is \$004000.

Configure the memory mode of the DSP using the Status Register (SR). The SR value combines the following bits for each feature:

- Sixteen-Bit Compatibility mode enables full compatibility to object code written for the DSP56000 family of DSPs, Bit 13 = \$0.
- Instruction Cache Enable bit enables the instruction cache controller and changes the last 1K of internal program memory into cache memory, Bit 19 = \$1.
- All other Status Register bits are selected for their defaults of \$000000.

The value loaded into the SR is \$080000, which is the value loaded during reset.

---

**Example 4-1. 128K × 24-bit 'P'/'X'/'Y' Space Fast SRAM Memory Exercise**

Motorola DSP56300 Assembler Version 6.0.1.6 97-01-25 08:05:25 asram4.asm

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
```

```

                                page    132,60,3,3,
;
;   ASRAM4.ASM - Simple program to test 128Kx24-bits of unified
;                   Program/X-Data/Y-Data memory using a DSP56303
;
;   The program uses Internal P:RAM to test External P/X/Y:RAM
;                   from $100000 - $11FFFF @ 1w/s
;
PMemStart equ    $100000
PMemEnd   equ    $120000
PMemSize  equ    PMemEnd-PMemStart

;--- Program Specific Storage Locations (X DATA SPACE)
MEM_FAIL_ADDRESS
000000 equ    $000000
MEM_FAIL_WROTE
000001 equ    $000001
MEM_FAIL_READ
000002 equ    $000002
MEM_PASS_COUNTER
000003 equ    $000003

;--- DSP56303 Control Registers (X I/O SPACE)
FFFFFB BCR equ    $FFFFFB ; Bus Control Register
FFFFFD PCTL equ    $FFFFFD ; PLL Control Register
FFFFF6 AAR3 equ    $FFFFF6 ; Address Attribute Register #3
FFFFF7 AAR2 equ    $FFFFF7 ; Address Attribute Register #2

;--- PCTL value = 0x0E0013
000000 prediv equ    0 ; Pre-Divider = 1
000000 lowdiv equ    0 ; Low Power Divider = 1
000013 pllmul equ    19 ; VCO Mult = 20; (19+1)*4.00MHz=80.00MHz
000000 crystal equ    0 ; No, Crystal not less than 200kHz
000000 disXTAL equ    0 ; No, do not disable crystal use
020000 pllstop equ    $020000 ; Yes, PLL runs during STOP
040000 enpll equ    $040000 ; Yes, enable PLL operation
080000 disclk equ    $080000 ; Yes, disable CORE clock output
0E0013 PCTL_value equ    prediv+lowdiv+pllmul+crystal+disXTAL+pllstop+enpll+disclk

;--- AAR3 value = 0x100739
000001 acctype3 equ    1 ; External Memory access type = 0x1
000000 aahigh3 equ    0 ; Enable AA3 pin to be low when selected
000008 aap3 equ    $8 ; Yes, Enable AA3 pin on ext 'P' accesses
000010 aax3 equ    $10 ; Yes, Enable AA3 pin on ext 'X' accesses
000020 aay3 equ    $20 ; Yes, Enable AA3 pin on ext 'Y' accesses
000000 aswap3 equ    0 ; No, Enable address bus swap
000000 enpack3 equ    0 ; No, Enable packing/unpacking logic
000700 nadd3 equ    $000700 ; Compare 7 address bits
100000 msadd3 equ    $100000 ; Most significant portion of address,
; $100000 - $11ffff, to compare.
; (0001,000x,xxxx,xxxx,xxxx,xxxx)
100739 AAR3_value equ    acctype3+aahigh3+aap3+aax3+aay3+aswap3+enpack3+nadd3+msadd3

;--- AAR2 value = 0x11083D
000001 acctype2 equ    1 ; External Memory access type = 0x1
000004 aahigh2 equ    $4 ; Enable AA2 pin to be high when selected
000008 aap2 equ    $8 ; Yes, Enable AA2 pin on ext 'P' accesses
000010 aax2 equ    $10 ; Yes, Enable AA2 pin on ext 'X' accesses
000020 aay2 equ    $20 ; Yes, Enable AA2 pin on ext 'Y' accesses
000000 aswap2 equ    0 ; No, Enable address bus swap
000000 enpack2 equ    0 ; No, Enable packing/unpacking logic
000800 nadd2 equ    $000800 ; Compare 8 address bits
110000 msadd2 equ    $110000 ; Most significant portion of address,
; $110000 - $11ffff, to compare.
; (0001,0001,xxxx,xxxx,xxxx,xxxx)
11083D AAR2_value equ    acctype2+aahigh2+aap2+aax2+aay2+aswap2+enpack2+nadd2+msadd2

;--- BCR value = 0x002400
000000 aaa0ws equ    0 ; Address Attribute Area 0 w/s = 0
000000 aaalws equ    0 ; Address Attribute Area 1 w/s = 0
000400 aaa2ws equ    $000400 ; Address Attribute Area 2 w/s = 1
002000 aaa3ws equ    $002000 ; Address Attribute Area 3 w/s = 1
000000 defws equ    0 ; Default Address Area w/s = 0
```



```
75 000000 busss equ 0 ; Bus state status = 0
76 000000 enblh equ 0 ; Enable Bus Lock Hold = 0
77 000000 enbrh equ 0 ; Enable Bus Request Hold = 0
78 002400 BCR_value equ aaa0ws+aaalws+aaa2ws+aaa3ws+defws+busss+enblh+enbrh
79
80 ;-----
81 P:000100 org p:$100 ;Keep the program in internal RAM
82
83 memtst
84
85 ; Initialization Section
86 P:000100 08F4BD movep #PCTL_value,x:PCTL ; Set PLL Control Register
87 P:000102 0E0013 05F43A movec #$004000,OMR ; Disable Address Attribute Priority
88 P:000104 004000 05F439 movec #$080000,SR ; Enable 1K Cache
89 P:000106 08F4BB movep #BCR_value,x:BCR ; Set external wait states
90 P:000108 002400 08F4B7 movep #AAR2_value,x:AAR2 ; Set Address Attribute Reg2
91 P:00010A 11083D 08F4B6 movep #AAR3_value,x:AAR3 ; Set Address Attribute Reg3
92 P:00010C 100739
93 P:00010C 05F420 move #-1,m0 ; Set LINEAR addressing mode
94 P:00010E FFFFFFFF 05F423 move #-1,m3
95 P:000110 FFFFFFFF
96 P:000110 20001B clr b
97 P:000111 000000 nop
98 P:000112 570000 move b,x:MEM_FAIL_ADDRESS ; Initialize Failed Address -> $000000
99 P:000113 570100 move b,x:MEM_FAIL_WROTE ; Initialize Expected Data -> $000000
100 P:000114 570200 move b,x:MEM_FAIL_READ ; Initialize Data Read -> $000000
101 P:000115 570300 move b,x:MEM_PASS_COUNTER ; Initialize Pass Counter -> $000000
102
103 main
104 ;-----
105 ;--- fill P:memory with initial pattern ---
106 ;-----
107 P:000116 63F400 move #PATT,r3 ; r3 points to Test Patterns
108 P:000118 000138 000000 nop
109 P:000119 000000 nop
110
111 P:00011A 07DB84 move p:(r3)+,x0 ; Get the Write Pattern for P:MEM
112
113 P:00011B 70F400 move #PMemSize,n0 ; Get memory size
114 P:00011D 020000
115 P:00011D 60F400 move #PMemStart,r0 ; Get starting address for fill
116 P:00011D 100000
117 P:00011F 06D820 rep n0 ; Fill RAM with first pattern data
118 P:000120 075884 move x0,p:(r0)+
119
120 ;-----
121 ;--- Check for expected pattern data in each RAM location ---
122 ;--- and then replace with a new data pattern. ---
123 ;--- ...This provides an address check. Since erroneous ---
124 ;--- ...addressing will cause the data to be written into ---
125 ;--- ...incorrect locations and this will be evident in ---
126 ;--- ...the next read pass. ---
127 ;-----
128 P:000121 063890 DOR #PATTN,test_Pm ; Start Pattern Test Loop
129 P:000123 00000D 60F400 move #PMemStart,r0 ; Get starting address of Test Memory
130 P:000125 100000 200041 tfr x0,a ; Save the last test pattern -> a
131 P:000126 07DB84 move P:(r3)+,x0 ; Get the next test pattern -> x0
132
133 ; Test this pattern through external RAM
134 P:000127 06D810 DOR n0,next_loc ; Test all external RAM locations
135 P:000129 000006 07E085 move P:(r0),x1 ; Read RAM location
136 P:00012A 200065 052409 cmp x1,a ; Read data = last test pattern?
137 P:00012B 075884 bne <ERR ; No, error if compare fails
138 P:00012C 000000 move x0,P:(r0)+ ; Yes, Write next test pattern -> RAM
139 P:00012D 000000 nop
140 next_loc
141
```



## 128K × 8-bit Memory Based Designs

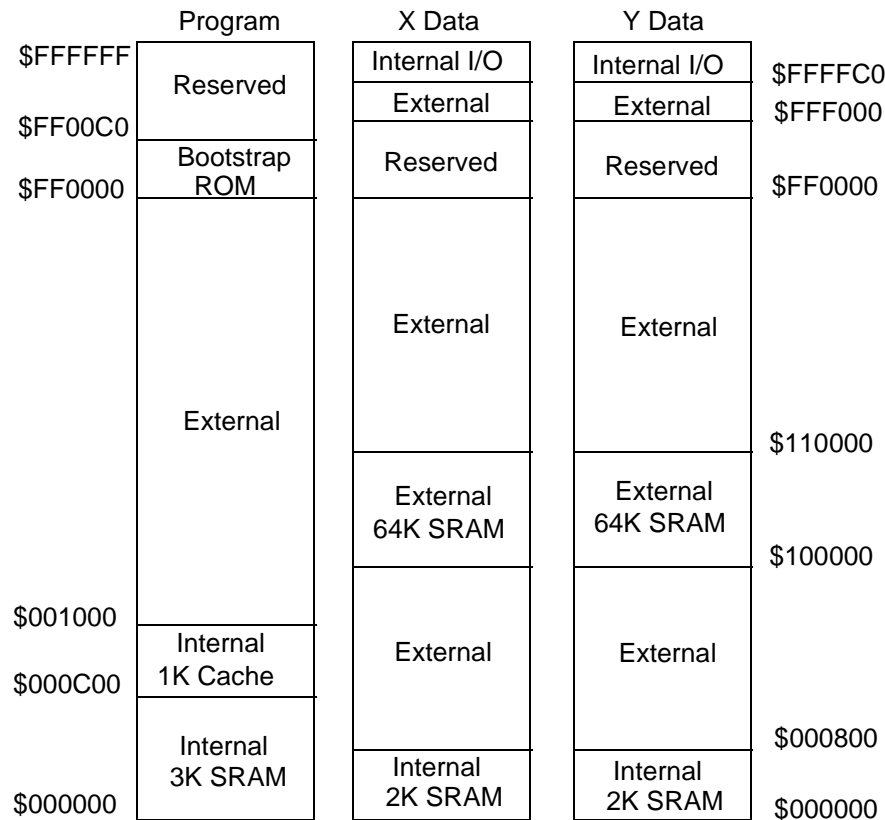
```
142 P:00012E 000000 nop ; Time to start next test pattern
143
144 test_Pm
145
146 ; One Pass Complete
147 ; All test patterns have been tried and passed in external RAM
148 P:00012F 518300 move x:MEM_PASS_COUNTER,b0
149 P:000130 000009 inc b ; Update pass loop counter
150 P:000131 000000 nop
151 P:000132 510300 move b0,x:MEM_PASS_COUNTER
152
153 P:000133 050FC3 bra main ; Do it all over again
154
155 ; -----
156 ; ERR -- handles error messaging with user
157 ;
158 ; Expected Data --> a
159 ; Read Data --> x1
160 ; Address of failure --> r0
161
162 ERR
163 P:000134 600000 move r0,x:MEM_FAIL_ADDRESS ; Save off address of failure
164 P:000135 560100 move a,x:MEM_FAIL_WROTE ; Save off expected data
165 P:000136 450200 move x1,x:MEM_FAIL_READ ; Save off data read
166
167 P:000137 050C00 bra * ; Dynamically HALT here
168
169
170 ; Memory Test Patterns
171
172 P:000138 PATT dc $000000,$FFFFFF,$AAAAAA,$555555,$2BAD2C
173 P:00013D dc $800000,$400000,$200000,$100000
174 P:000141 dc $080000,$040000,$020000,$010000
175 P:000145 dc $008000,$004000,$002000,$001000
176 P:000149 dc $000800,$000400,$000200,$000100
177 P:00014D dc $000080,$000040,$000020,$000010
178 P:000151 dc $000008,$000004,$000002,$000001
179 P:000155 dc $7FFFFFF,$BFFFFFF,$DFFFFFF,$EFFFFFF
180 P:000159 dc $F7FFFF,$FBFFFF,$FDFFFF,$FEFFFF
181 P:00015D dc $FF7FFF,$FFBFFF,$FFDFFF,$FFEFFF
182 P:000161 dc $FFF7FF,$FFFBFF,$FFFDFF,$FFF7FF
183 P:000165 dc $FFFF7F,$FFFFBF,$FFFFDF,$FFFFEF
184 P:000169 dc $FFFFF7,$FFFFFB,$FFFFFD,$FFFFFE
185 P:00016D dc $FEDCBA,$123456,$012345,$EDCBA9
186
187 000038 PATTN equ *-PATT-1
188
189 end memtst

0 Errors
0 Warnings
```

## 4.3 64K × 24-bit 'X' and 64K × 24-bit 'Y' Fast SRAM Example

This section describes a 64K × 24-bit 'X' and 64K × 24-bit 'Y' memory space, asynchronous Fast SRAM implementation using Motorola's MCM6926 device (see **Figure 4-6** for the memory map layout, **Figure 4-1** for the block diagram, and **Example 4-2** for the example code).

The DSP core runs at 80 MHz and the input frequency source is a 4.000 MHz crystal. The asynchronous Fast SRAM requires an access time equal to or less than 12.4 nS to satisfy the DSP's one wait state external memory requirements.



**Figure 4-6.** 64K × 24-bit 'X' and 64K × 24-bit 'Y' Memory Map

### 4.3.1 DSP56303 Port A Timing Requirements and Register Settings

For optimal use of the 64K × 24-bit 'X' and 64K × 24-bit 'Y' space memory configuration, set up the following DSP control registers.

Set the core speed of the DSP for optimum processor and memory performance using the DSP PLL and Clock Generation register (PCTL). For this example, the DSP core runs at 80 MHz and the input frequency source is a 4.000 MHz crystal. The PCTL register value combines the following bits for each feature:

- Desired Core Frequency = 80 MHz
- Given the External Frequency = 4.000 MHz
- Predivider value = 1, Bits 20–23 = \$0
- Low-power Divider value = 1, Bits 12–14 = \$0
- VCO Multiplication value = 20, Bits 0–11 = \$013
- Crystal less than 200 kHz, Bit 15 = 0
- Disable XTAL drive output, Bit 16 = 0
- PLL runs during STOP, Bit 17 = 1
- Enable PLL operation, Bit 18 = 1
- Disable core clock output, Bit 19 = 1

The value loaded into the PCTL register is \$0E0013.

Address Attribute Pin 3 enables, via Fast SRAM  $\bar{E}$ , the external 128K SRAM bank accesses in the address range from \$100000 to \$10FFFF during X data and Y data space requests. Configure the memory address space requirements for the Address Attribute Pin 3 using Address Attribute Register 3 (AAR3). The AAR3 value combines the following bits for each feature:

- Specify the external memory access type as asynchronous SRAM, Bits 0–1 = \$1.
- Pull the AA pin high when selected, Bit 2 = 0.
- Activate the AA pin during external program space accesses, Bit 3 = 0.
- Activate the AA pin during external X data space accesses, Bit 4 = 1.
- Activate the AA pin during external Y data space accesses, Bit 5 = 1.
- Move the eight least significant bits of the address to the eight most significant bits of the external address bus, Bit 6 = 0.
- Enable the internal packing/unpacking logic during external DMA accesses, Bit 7 = 0.
- Specify the number of address bits to compare, Bits 8–11 = \$8
- Specify the most significant portion of the address to compare, Bits 12–23 = \$100

The value loaded into the AAR3 is \$100831.

---

Address Attribute Pin 2 selects, via Fast SRAM A16, the external 128K SRAM bank during accesses in the address range from \$100000 to \$10FFFF to differentiate X data space requests from Y data space requests. Configure the memory address space requirements for the Address Attribute Pin 2 using the Address Attribute Register 2 (AAR2). The AAR2 value combines the following bits for each feature:

- Specify the external memory access type as asynchronous SRAM, bits 0–1 = \$1.
- Pull the AA pin high when selected, Bit 2 = 1.
- Activate the AA pin during external program space accesses, Bit 3 = 0.
- Activate the AA pin during external X data space accesses, Bit 4 = 1.
- Activate the AA pin during external Y data space accesses, Bit 5 = 0.
- Move the eight least significant bits of the address to the eight most significant bits of the external address bus, Bit 6 = 0.
- Enable the internal packing/unpacking logic during external DMA accesses, Bit 7 = 0.
- Specify the number of address bits to compare, Bits 8–11 = \$8
- Specify the most significant portion of the address to compare, Bits 12–23 = \$100

The value loaded into the AAR2 is \$100815.

The value loaded into AAR0 and AAR1 is \$000000.

Select the proper number of wait states for the memory configuration using the Bus Control Register (BCR). The BCR value combines the following bits for each feature:

- Address attribute area 0 wait states, Bits 0–4 = \$0
- Address attribute area 1 wait states, Bits 5–9 = \$0
- Address attribute area 2 wait states, Bits 10–12 = \$1
- Address attribute area 3 wait states, Bits 13–15 = \$1
- Default address area wait states, Bits 16–20 = \$0
- Bus state status, Bit 21 = 0
- Enable Bus Lock Hold, Bit 22 = 0
- Enable Bus Request Hold, Bit 23 = 0

The value loaded into the BCR is \$002400.

---

Configure the operating mode and external memory controls using the Operating Mode Register (OMR). The OMR value combines the following bits for each feature:

- MA–MD bits specify the DSP operating mode, Bits 0–3 = \$0.
- External Bus Disable bit disables the external bus controller for power conservation when external memory is not used, Bit 4 = \$0.
- Memory Switch Mode bit reconfigures internal memory spaces, Bit 7 = \$0.
- Transfer Acknowledge Synchronize Select bit selects the synchronization method for the Transfer Acknowledge (TA) pin, Bit 11 = \$0.
- Bus Release Timing bit selects between a fast and slow bus release of the  $\overline{BB}$  pin, Bit 12 = \$0.
- Address Attribute Priority Disable bit allows the Address Attribute pins, AA0–AA3, to be used in any combination, Bit 14 = \$1.
- All other OMR bits are selected for their defaults of \$000000.

The value loaded into the OMR is \$004000.

Configure the memory mode of the DSP using the Status Register (SR). The SR register value combines the following bits for each feature:

- Sixteen-Bit Compatibility mode enables full compatibility to object code written for the DSP56000 family, Bit 13 = \$0.
- Instruction Cache Enable bit enables the instruction cache controller and changes the last 1K of internal program memory into cache memory, Bit 19 = \$1.

The value loaded into the SR is \$080000, which is the value loaded during reset.

The value loaded into the SR is \$080000, which is the value loaded during reset.

---







## 128K × 8-bit Memory Based Designs

```
74                                     ; (0001,0000,xxxx,xxxx,xxxx,xxxx)
75 100815      AAR2_valu equ      acctype2+aahigh2+aap2+aax2+aay2+aswap2+enpack2+nadd2+msadd2
76
77      ;--- BCR value = 0x002400
78 000000      aaa0ws   equ      0          ; Address Attribute Area 0 w/s = 0
79 000000      aaalws   equ      0          ; Address Attribute Area 1 w/s = 0
80 000400      aaa2ws   equ      $000400   ; Address Attribute Area 2 w/s = 1
81 002000      aaa3ws   equ      $002000   ; Address Attribute Area 3 w/s = 1
82 000000      defws   equ      0          ; Default Address Area w/s = 0
83 000000      bussss   equ      0          ; Bus state status = 0
84 000000      enblh   equ      0          ; Enable Bus Lock Hold = 0
85 000000      enbrh   equ      0          ; Enable Bus Request Hold = 0
86 002400      BCR_value equ      aaa0ws+aaalws+aaa2ws+aaa3ws+defws+bussss+enblh+enbrh
87
88      ;-----
89 P:000100      org      p:$100          ;Keep the program in internal RAM
90
91      memtst
92
93      ; Initialization Section
94 P:000100      08F4BD   movewp  #PCTL_value,x:PCTL      ; Set PLL Control Register
95                                     0E0013
96 P:000102      05F43A   movewc  #$004000,OMR          ; Disable Address Attribute Priority
97                                     004000
98 P:000104      05F439   movewc  #$080000,SR          ; Enable 1K Cache
99                                     080000
100 P:000106      08F4BB   movewp  #BCR_value,x:BCR      ; Set external wait states
101                                     002400
102 P:000108      08F4B7   movewp  #AAR2_value,x:AAR2    ; Set Address Attribute Reg2
103                                     100815
104 P:00010A      08F4B6   movewp  #AAR3_value,x:AAR3    ; Set Address Attribute Reg3
105                                     100831
106
107 P:00010C      05F420   move     #-1,m0          ; Set LINEAR addressing mode
108                                     FFFFFFFF
109 P:00010E      05F423   move     #-1,m3
110                                     FFFFFFFF
111
112 P:000110      20001B   clr      b
113 P:000111      000000   nop
114 P:000112      570000   move     b,x:X_MEM_FAIL_ADDRESS ; Initialize X:Failed Address -> $000000
115 P:000113      570100   move     b,x:X_MEM_FAIL_WROTE  ; Initialize X:Expected Data -> $000000
116 P:000114      570200   move     b,x:X_MEM_FAIL_READ   ; Initialize X:Data Read -> $000000
117 P:000115      570300   move     b,x:Y_MEM_FAIL_ADDRESS ; Initialize Y:Failed Address -> $000000
118 P:000116      570400   move     b,x:Y_MEM_FAIL_WROTE  ; Initialize Y:Expected Data -> $000000
119 P:000117      570500   move     b,x:Y_MEM_FAIL_READ   ; Initialize Y:Data Read -> $000000
120 P:000118      570600   move     b,x:MEM_PASS_COUNTER  ; Initialize Pass Counter -> $000000
121
122      main
123      ;-----
124      ;--- fill external memory with initial pattern      ---
125      ;-----
126 P:000119      63F400   move     #PATT,r3          ; r3 points to Test Patterns
127                                     00014A
128 P:00011B      000000   nop
129 P:00011C      000000   nop
130
131 P:00011D      07DB84   move     p:(r3)+,x0        ; Get the Write Pattern for X:MEM
132 P:00011E      07DB86   move     p:(r3)+,y0        ; Get the Write Pattern for Y:MEM
133
134 P:00011F      70F400   move     #MemSize,n0       ; Get memory size
135                                     010000
136
137 P:000121      60F400   move     #MemStart,r0      ; Get starting address for fill
138                                     100000
139 P:000123      06D820   rep     n0                  ; Fill X:RAM with first data pattern
140 P:000124      445800   move     x0,x:(r0)+
141
142 P:000125      60F400   move     #MemStart,r0
143                                     100000
144 P:000127      06D820   rep     n0                  ; Fill Y:RAM with first data pattern
145 P:000128      4E5800   move     y0,y:(r0)+
146
147      ;-----
148      ;--- Check for expected pattern data in each RAM location ---
149      ;--- and then replace with a new data pattern. ---
150      ;--- ...This provides an address check. Since erroneous ---
151      ;--- ...addressing will cause the data to be written into ---
152      ;--- ...incorrect locations and this will be evident in ---
153      ;--- ...the next read pass. ---
```



```
142 ;-----  
143 P:000129 063890 DOR #PATTN,test_m ; Start Pattern Test Loop  
144 P:00012B 60F400 move #MemStart,r0 ; Get starting address of Test Memory  
145 P:00012D 200041 tfr x0,a ; Save the last X test pattern -> a  
146 P:00012E 200059 tfr y0,b ; Save the last Y test pattern -> b  
147 P:00012F 07DB84 move P:(r3)+,x0 ; Get the next X test pattern -> x0  
148 P:000130 07DB86 move P:(r3)+,y0 ; Get the next Y test pattern -> y0  
149  
150 ; Test this pattern through external RAM  
151 P:000131 06D810 DOR n0,next_loc ; Test all external RAM locations  
152 P:000133 45E000 move X:(r0),x1 ; Read X:RAM location  
153 P:000134 200065 cmp x1,a ; Read data = last test pattern?  
154 P:000135 05240D bne <XERR ; No, error if compare fails  
155 P:000136 446000 move x0,X:(r0) ; Yes, Write next test pattern -> X:RAM  
156  
157 P:000137 4FE000 move Y:(r0),y1 ; Read Y:RAM location  
158 P:000138 20007D cmp y1,b ; Read data = last test pattern?  
159 P:000139 05240D bne <YERR ; No, error if compare fails  
160 P:00013A 4E5800 move y0,Y:(r0)+ ; Yes, Write next test pattern -> Y:RAM  
161 P:00013B 000000 nop  
162 next_loc  
163  
164 P:00013C 000000 nop ; Time to start next test pattern  
165  
166 test_m  
167  
168 ; One Pass Complete  
169 ; All test patterns have been tried and passed in external RAM  
170 P:00013D 518600 move x:MEM_PASS_COUNTER,b0  
171 P:00013E 000009 inc b ; Update pass loop counter  
172 P:00013F 000000 nop  
173 P:000140 510600 move b0,x:MEM_PASS_COUNTER  
174  
175 P:000141 050F98 bra main ; Do it all over again  
176  
177 ;-----  
178 ; XERR -- handles X:RAM error messaging with user  
179 ;  
180 ; Expected Data --> a  
181 ; Read Data --> x1  
182 ; Address of failure --> r0  
183  
184 XERR  
185 P:000142 600000 move r0,x:X_MEM_FAIL_ADDRESS ; Save off address of failure  
186 P:000143 560100 move a,x:X_MEM_FAIL_WROTE ; Save off expected data  
187 P:000144 450200 move x1,x:X_MEM_FAIL_READ ; Save off data read  
188  
189 P:000145 050C00 bra * ; Dynamically HALT here  
190  
191 ;-----  
192 ; YERR -- handles Y:RAM error messaging with user  
193 ;  
194 ; Expected Data --> b  
195 ; Read Data --> y1  
196 ; Address of failure --> r0  
197  
198 YERR  
199 P:000146 600300 move r0,x:Y_MEM_FAIL_ADDRESS ; Save off address of failure  
200 P:000147 570400 move b,x:Y_MEM_FAIL_WROTE ; Save off expected data  
201 P:000148 470500 move y1,x:Y_MEM_FAIL_READ ; Save off data read  
202  
203 P:000149 050C00 bra * ; Dynamically HALT here  
204  
205  
206  
207 ; Memory Test Patterns  
208  
209 P:00014A PATT dc $000000,$FFFFFF,$AAAAAA,$555555,$2BAD2C  
210 P:00014F dc $800000,$400000,$200000,$100000  
211 P:000153 dc $080000,$040000,$020000,$010000  
212 P:000157 dc $008000,$004000,$002000,$001000  
213 P:00015B dc $000800,$000400,$000200,$000100  
214 P:00015F dc $000080,$000040,$000020,$000010  
215 P:000163 dc $000008,$000004,$000002,$000001  
216 P:000167 dc $7FFFFFF,$BFFFFFF,$DFFFFFF,$EFFFFFF
```



## 128K × 8-bit Memory Based Designs

---

```
217 P:00016B dc $F7FFFF,$FBFFFF,$FDFFFF,$FEFFFF
218 P:00016F dc $FF7FFF,$FFBFFF,$FFDFFF,$FFEFFF
219 P:000173 dc $FFF7FF,$FFFBFF,$FFDFFF,$FFEFFF
220 P:000177 dc $FFFF7F,$FFFFBF,$FFFFDF,$FFFFEF
221 P:00017B dc $FFFFF7,$FFFFFB,$FFFFFD,$FFFFFE
222 P:00017F dc $FEDCBA,$123456,$012345,$EDCBA9
223
224 P:000183 dc $000000,$FFFFFF,$AAAAAA,$555555,$2BAD2C
225 P:000188 dc $800000,$400000,$200000,$100000
226 P:00018C dc $080000,$040000,$020000,$010000
227 P:000190 dc $008000,$004000,$002000,$001000
228 P:000194 dc $000800,$000400,$000200,$000100
229 P:000198 dc $000080,$000040,$000020,$000010
230 P:00019C dc $000008,$000004,$000002,$000001
231 P:0001A0 dc $7FFFFFF,$BFFFFFF,$DFFFFFF,$EFFFFFF
232 P:0001A4 dc $F7FFFF,$FBFFFF,$FDFFFF,$FEFFFF
233 P:0001A8 dc $FF7FFF,$FFBFFF,$FFDFFF,$FFEFFF
234 P:0001AC dc $FFF7FF,$FFFBFF,$FFDFFF,$FFEFFF
235 P:0001B0 dc $FFFF7F,$FFFFBF,$FFFFDF,$FFFFEF
236 P:0001B4 dc $FFFFF7,$FFFFFB,$FFFFFD,$FFFFFE
237 P:0001B8 dc $FEDCBA,$123456,$012345,$EDCBA9
238
239 000038 PATTN equ (( *-PATT)/2)-1
240
241 end memtst

0 Errors
0 Warnings
```

---

## 5 64K × 24-bit Memory Based Designs

This section describes how to implement several different DSP memory space Fast SRAM configuration designs using one 64K × 24-bit 3.3 V memory with a Motorola DSP56303 device.

Using one hardware memory design based on one 64K × 24-bit 3.3 V memory, the DSP56303's Memory Expansion Port allows the memory bank to be logically configured for use in various memory space arrangements.

Configuring and using one Memory Expansion Port Address Attribute Control Register, the memory bank can accommodate seven different memory space arrangements:

1. 64K × 24-bit 'P' Space Fast SRAM
2. 64K × 24-bit 'X' Space Fast SRAM
3. 64K × 24-bit 'Y' Space Fast SRAM
4. 64K × 24-bit 'P'/'X' Space Fast SRAM
5. 64K × 24-bit 'P'/'Y' Space Fast SRAM
6. 64K × 24-bit 'X'/'Y' Space Fast SRAM
7. 64K × 24-bit 'P'/'X'/'Y' Space Fast SRAM

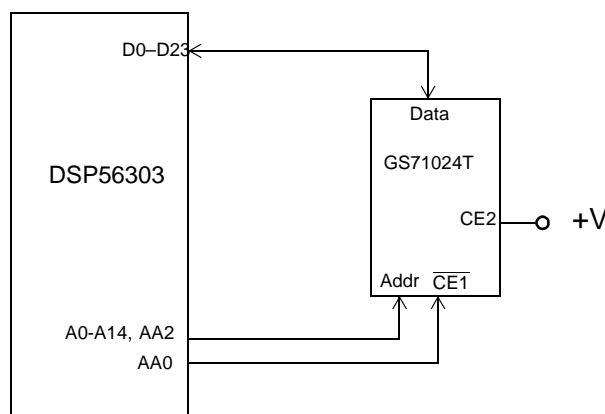
Configuring and using two Memory Expansion Port Address Attribute Control Registers, the memory bank can accommodate thirteen different memory space arrangements:

1. 64K × 24-bit 'P' Space Fast SRAM
2. 64K × 24-bit 'X' Space Fast SRAM
3. 64K × 24-bit 'Y' Space Fast SRAM
4. 64K × 24-bit 'P'/'X' Space Fast SRAM
5. 64K × 24-bit 'P'/'Y' Space Fast SRAM
6. 64K × 24-bit 'X'/'Y' Space Fast SRAM
7. 64K × 24-bit 'P'/'X'/'Y' Space Fast SRAM
8. 32K × 24-bit 'P'/'X' and 32K × 24-bit 'Y' Space Fast SRAM
9. 32K × 24-bit 'P'/'Y' and 32K × 24-bit 'X' Space Fast SRAM
10. 32K × 24-bit 'P' and 32K × 24-bit 'X'/'Y' Space Fast SRAM
11. 32K × 24-bit 'P' and 32K × 24-bit 'X' Space Fast SRAM
12. 32K × 24-bit 'P' and 32K × 24-bit 'Y' Space Fast SRAM
13. 32K × 24-bit 'X' and 32K × 24-bit 'Y' Space Fast SRAM

The memory configuration examples presented in the remainder of this section are based on one common hardware design that uses two Address Attributes and implements two of the most common configurations: 32K × 24-bit 'P'/'X' and 32K × 24-bit 'Y' Space Fast SRAM configuration and a 32K × 24-bit 'X' and 32K × 24-bit 'Y' Space Fast SRAM configuration (see **Figure 5-4** for a schematic of the hardware design).

## 5.1 64K × 24-bit Common Fast SRAM Hardware Design

This section describes an asynchronous Fast SRAM 64K × 24-bit memory bank implementation using GSI Technology's GS71024T device (see **Figure 5-1**). Memory bank designs using single memory devices allow for compact embedded designs.



**Figure 5-1.** 64K × 24-bit Fast SRAM Memory Example

The memory bank design is implemented using two Address Attribute Selectors, AA2 and AA0. AA2 splits the 64K address space of the memory bank into two 32K address spaces. AA0 enables the memory bank.

For this common hardware design, the DSP core runs at a maximum of 80 MHz, and the input frequency source is a 4.000 MHz crystal. The asynchronous Fast SRAM requires an access time equal to or less than 12.4 nS to satisfy the DSP's one wait state external memory requirements. This device is organized as 64K × 24-bits. Therefore, one memory device satisfies the 24-bit wide general-purpose DSP bus.

Since the memory device operates at 3.3 V, no voltage level conversion is required.

### 5.1.1 GS71024T-12 Memory Timing Requirements

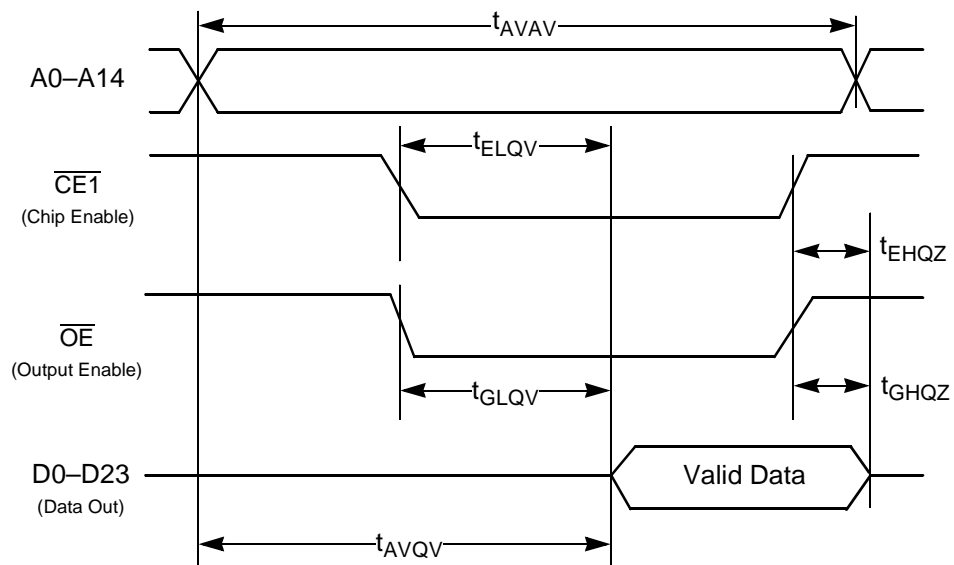
For the asynchronous Fast SRAM device to work properly, its timing requirements must be met. Following are the timing requirements for the GS71024T-12 64K × 24-bit 12 nS Fast SRAM.

### 5.1.1.1 Read Cycle Timing

**Table 5-1** contains the memory read timing specification values used in the memory read cycle timing diagram, **Figure 5-2**.

**Table 5-1.** GS71024T-12 Memory Read Timing Specifications

Read Cycle Parameter	Symbol	Min	Max
Read Cycle Time	$t_{AVAV}$	12 nS	—
Address Access Time	$t_{AVQV}$	—	12 nS
Enable Access Time	$t_{ELQV}$	—	12 nS
Output Enable Access Time	$t_{GLQV}$	—	6 nS
Enable High to Output High-Z	$t_{EHQZ}$	—	6 nS
Output Enable High to Output High-Z	$t_{GHQZ}$	—	6 nS



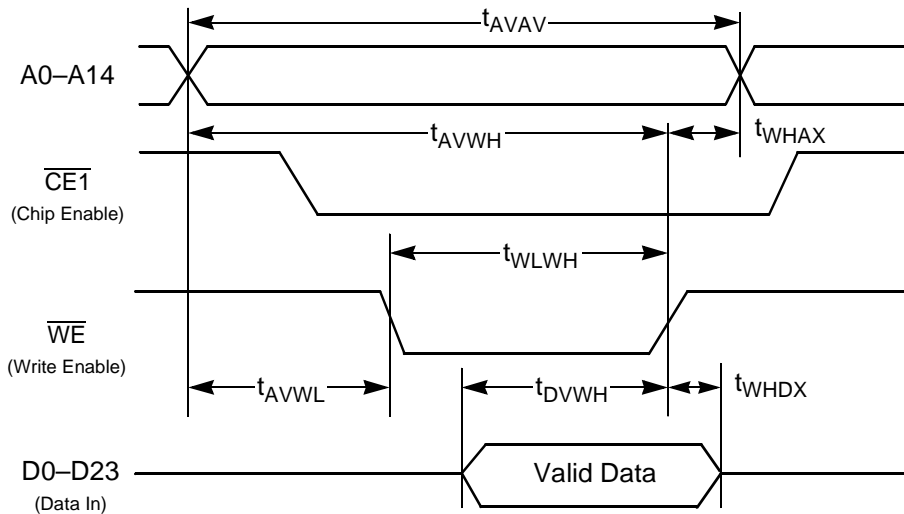
**Figure 5-2.** GS71024T-12 Memory Read Cycle Timing Diagram

### 5.1.1.2 Write Cycle Timing

**Table 5-2** shows the memory write timing values in the memory write cycle timing diagram, **Figure 5-3**.

**Table 5-2.** GS71024T-12 Memory Write Timing Specifications

Write Cycle Parameter	Symbol	Min	Max
Write Cycle Time	$t_{AVAV}$	12 nS	—
Address Setup Time	$t_{AVWL}$	0 nS	—
Address Valid to End off Write	$t_{AVWH}$	8 nS	—
Write Pulse Width	$t_{WLWH}$	8 nS	—
Data Valid to End of Write	$t_{DVWH}$	3 nS	—
Data Hold Time	$t_{WHDX}$	0 nS	—



**Figure 5-3.** GS71024T-12 Memory Write Cycle Timing Diagram



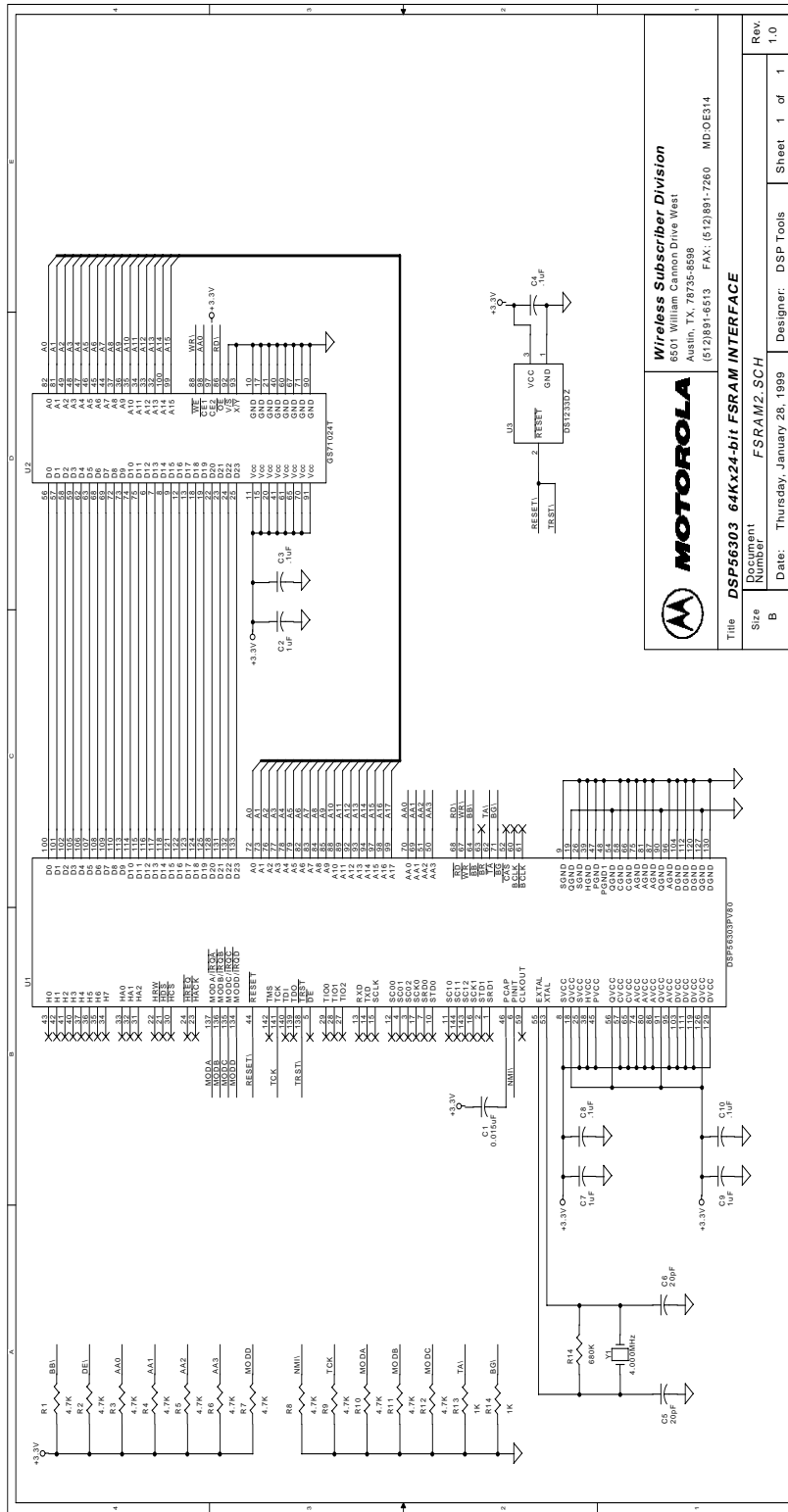


Figure 5-4. 128K × 32-bit Fast SRAM Schematic

**MOTOROLA**

*Wireless Subscriber Division*  
6501 William Cannon Drive West  
Austin, TX, 78735-8998  
(512)891-6513 FAX: (512)891-7260 MD.OE314

---

Title: **DSF6003 64Kx24-bit FSRAM INTERFACE**

Size: **B**    Part Number: **FSRAM2.SCH**    Designer: **DSP Tools**    Sheet: **1 of 1**

Date: **Thursday, January 28, 1999**    Rev: **1.0**

## 5.2 32K × 24-bit ‘P’/‘X’ and 32K × 24-bit ‘Y’ Fast SRAM Example

This section describes a 64K × 24-bit shared ‘P’/‘X’ and 32K × 24-bit ‘Y’ memory space, asynchronous Fast SRAM implementation using GSI’s GS71024T device (see **Figure 5-5** for memory map layout, **Figure 5-1** for the block diagram, and **Example 5-1** for the example code). In a shared memory space, data written to one address in one memory space can be accessed by the same address in another memory space (e.g., writing \$012345 to P:\$100000 could be read back by X:\$100000 or P:\$100000).

The DSP core runs at 80 MHz, and the input frequency source is a 4.000 MHz crystal. The asynchronous Fast SRAM requires an access time equal to or less than 12.4 nS to satisfy the DSP’s one wait state external memory requirements.

	Program	X Data	Y Data	
\$FFFFFF	Reserved	Internal I/O	Internal I/O	\$FFFFC0
		External	External	\$FFF000
\$FF00C0	Bootstrap ROM	Reserved	Reserved	\$FF0000
\$FF0000				
	External	External	External	
\$108000	Shared External 32K SRAM	Shared External 32K SRAM	External 32K SRAM	\$108000
\$100000				
\$001000	External	External	External	\$100000
\$000C00	Internal 1K Cache			
	Internal 3K SRAM	Internal 2K SRAM	Internal 2K SRAM	\$000800
\$000000				\$000000

**Figure 5-5.** 32K × 24-bit ‘P’/‘X’ and 32K × 24-bit ‘Y’ Fast SRAM Memory Map

### 5.2.1 DSP56303 Port A Timing Requirements and Register Settings

For optimal use of the 32K × 24-bit ‘P’/‘X’ and 32K × 24-bit ‘Y’ space memory configuration, set up the following DSP control registers.

Set the core speed of the DSP for optimum processor and memory performance using the DSP PLL and Clock Generation register (PCTL). For this example, the DSP core runs at 80 MHz and the input frequency source is a 4.000 MHz crystal. The PCTL register value combines the following bits for each feature:

- Desired Core Frequency = 80 MHz
- Given the External Frequency = 4.000 MHz
- Predivider value = 1, bits 20–23 = \$0
- Low-power Divider value = 1, Bits 12–14 = \$0
- VCO Multiplication value = 20, Bits 0–11 = \$013
- Crystal less than 200 kHz, Bit 15 = 0
- Disable XTAL drive output, Bit 16 = 0
- PLL runs during STOP, Bit 17 = 1
- Enable PLL operation, Bit 18 = 1
- Disable core clock output, Bit 19 = 1

The value loaded into the PCTL register is \$0E0013.

Address Attribute Pin 0 enables, via Fast SRAM  $\overline{CE1}$ , external 64K SRAM bank accesses in the address range from \$100000 to \$107FFF during Program/X data, and Y data space requests. Configure the memory address space requirements for the Address Attribute Pin 0 using Address Attribute Register 0 (AAR0). The AAR0 value combines the following bits for each feature:

- Specify the external memory access type as asynchronous SRAM, bits 0–1 = \$1
- Pull the AA pin high when selected, Bit 2 = 0
- Activate the AA pin during external program space accesses, Bit 3 = 1
- Activate the AA pin during external X data space accesses, Bit 4 = 1
- Activate the AA pin during external Y data space accesses, Bit 5 = 1
- Move the eight least significant bits of the address to the eight most significant bits of the external address bus, Bit 6 = 0
- Enable the internal packing/unpacking logic during external DMA accesses, Bit 7 = 0
- Specify the number of address bits to compare, Bits 8–11 = \$9
- Specify the most significant portion of the address to compare, Bits 12–23 = \$100

The value loaded into the AAR0 is \$100939.

---

Address Attribute Pin 2 selects, via Fast SRAM A15, the external 64K SRAM bank during accesses in the address range from \$100000 to \$107FFF between program/X data space requests and Y data space requests. Configure the memory address space requirements for the Address Attribute Pin 2 using Address Attribute Register 2 (AAR2). The AAR2 value combines the following bits for each feature:

- Specify the external memory access type as asynchronous SRAM, Bits 0–1 = \$1
- Pull the AA pin high when selected, Bit 2 = 1
- Activate the AA pin during external program space accesses, Bit 3 = 0
- Activate the AA pin during external X data space accesses, Bit 4 = 0
- Activate the AA pin during external Y data space accesses, Bit 5 = 1
- Move the eight least significant bits of the address to eight most significant bits of the external address bus, Bit 6 = 0
- Enable the internal packing/unpacking logic during external DMA accesses, Bit 7 = 0
- Number of address bits to compare, Bits 8–11 = \$9
- Most significant portion of address to compare, Bits 12–23 = \$100

The value loaded into the AAR2 register is \$100925.

Select the proper number of wait states must for the memory configuration using the Bus Control Register (BCR). The BCR value combines the following bits for each feature:

- Address attribute area 0 wait states, Bits 0–4 = \$1
- Address attribute area 1 wait states, Bits 5–9 = \$1
- Address attribute area 2 wait states, Bits 10–12 = \$1
- Address attribute area 3 wait states, Bits 13–15 = \$1
- Default address area wait states, Bits 16–20 = \$0
- Bus state status, Bit 21 = 0
- Enable Bus Lock Hold, Bit 22 = 0
- Enable Bus Request Hold, Bit 23 = 0

The value loaded into the BCR is \$002421.

---

Configure the operating mode and external memory controls using the Operating Mode Register (OMR). The OMR value combines the following bits for each feature:

- MA–MD bits specify the DSP operating mode, Bits 0–3 = \$0.
- External Bus Disable bit disables the external bus controller for power conservation when external memory is not used, Bit 4 = \$0.
- Memory Switch Mode bit reconfigures internal memory spaces, Bit 7 = \$0.
- Transfer Acknowledge Synchronize Select bit selects the synchronization method for the Transfer Acknowledge (TA) pin, Bit 11 = \$0.
- Bus Release Timing bit selects between a fast and slow bus release of the  $\overline{BB}$  pin, Bit 12 = \$0.
- Address Attribute Priority Disable bit allows the Address Attribute pins, AA0–AA3, to be used in any combination, Bit 14 = \$1.
- All other OMR bits are selected for their defaults of \$000000.

The value loaded into the OMR is \$004000.

Configure the memory mode of the DSP using the Status Register (SR). The SR value combines the following bits for each feature:

- Sixteen-Bit Compatibility mode enables full compatibility to object code written for the DSP56000 family of DSPs, Bit 13 = \$0.
- Instruction Cache Enable bit enables the instruction cache controller and changes the last 1K of internal program memory into cache memory, Bit 19 = \$1.
- All other Status Register bits are selected for their defaults of \$000000.

The value loaded into the SR is \$080000, which is the value loaded during reset.

---



**Example 5-1. 32K x 24-bit 'P'/X' and 32K x 24-bit 'Y Space Fast SRAM Memory Exercise**

Motorola DSP56300 Assembler Version 6.0.1.6 97-01-25 08:05:33 asram6.asm

```
1
2           page      132,60,3,3,
3           ;
4           ;   ASRAM6.ASM - Simple program to test 32Kx24-bit of Program/X-Data and
5           ;   32Kx24-bits of Y-Data memory
6           ;   using a DSP56303
7           ;
8           ;   The program uses Internal P:RAM to test:
9           ;   24-bit External P:/X: & Y:RAM from $100000 - $107FFF @ 1w/s
10
11
12          100000      PMemStart      equ      $100000
13          108000      PMemEnd       equ      $108000
14          008000      PMemSize      equ      PMemEnd-PMemStart
15
16          100000      YMemStart      equ      $100000
17          108000      YMemEnd       equ      $108000
18          008000      YMemSize      equ      YMemEnd-YMemStart
19
20          ;--- Program Specific Storage Locations (X DATA SPACE)
21          P_MEM_FAIL_ADDRESS
22          000000      equ      $000000
23          P_MEM_FAIL_WROTE
24          000001      equ      $000001
25          P_MEM_FAIL_READ
26          000002      equ      $000002
27
28          Y_MEM_FAIL_ADDRESS
29          000003      equ      $000003
30          Y_MEM_FAIL_WROTE
31          000004      equ      $000004
32          Y_MEM_FAIL_READ
33          000005      equ      $000005
34
35          MEM_PASS_COUNTER
36          000009      equ      $000009
37
38          old_p_pattern
39          00000A      equ      $00000A      ; Last written pattern to P:/X:RAM
40          old_y_pattern
41          00000B      equ      $00000B      ; Last written pattern to Y:RAM
42
43          ;--- DSP56303 Control Registers (X I/O SPACE)
44          BCR         equ      $FFFFFB      ; Bus Control Register
45          PCTL        equ      $FFFFFD      ; PLL Control Register
46          AAR3        equ      $FFFFF6      ; Address Attribute Register #3
47          AAR2        equ      $FFFFF7      ; Address Attribute Register #2
48          AAR1        equ      $FFFFF8      ; Address Attribute Register #1
49          AAR0        equ      $FFFFF9      ; Address Attribute Register #0
50
51          ;--- PCTL value = 0x0E0013
52          000000      prediv equ      0      ; Pre-Divider = 1
53          000000      lowdiv equ      0      ; Low Power Divider = 1
54          000013      pllmul equ      19     ; VCO Mult = 20; (19+1)*4.00MHz=80.00MHz
55          000000      crystal equ      0     ; No, Crystal not less than 200kHz
56          000000      disXTAL equ      0     ; No, do not disable crystal use
57          020000      pllstop equ      $020000 ; Yes, PLL runs during STOP
58          040000      enpll equ      $040000 ; Yes, enable PLL operation
59          080000      disclk equ      $080000 ; Yes, disable CORE clock output
60          0E0013      PCTL_value equ      prediv+lowdiv+pllmul+crystal+disXTAL+pllstop+enpll+disclk
61
62          ;--- AAR2 value = 0x100925
63          000001      acctype2 equ      1     ; External Memory access type = 0x1
64          000004      aahigh2 equ      $4     ; Enable AA2 pin to be high when selected
65          000000      aap2 equ      0        ; No, Enable AA2 pin on ext 'P' accesses
66          000000      aax2 equ      0        ; No, Enable AA2 pin on ext 'X' accesses
67          000020      aay2 equ      $20     ; Yes, Enable AA2 pin on ext 'Y' accesses
68          000000      aswap2 equ      0      ; No, Enable address bus swap
69          000000      enpack2 equ      0      ; No, Enable packing/unpacking logic
70          000800      nadd2 equ      $000900 ; Compare 9 address bits
71          100000      msadd2 equ      $100000 ; Most significant portion of address,
72          ; $100000 - $107fff, to compare.
73          ; (0001,0000,0xxx,xxxx,xxxx,xxxx)
74          100925      AAR2_value equ      acctype2+aahigh2+aap2+aax2+aay2+aswap2+enpack2+nadd2+msadd2
```

```

75
76      ;--- AAR0 value = 0x100939
77      000001      acctype0      equ      1          ; External Memory access type = 0x1
78      000004      aahigh0       equ      $0          ; Enable AA0 pin to be high when selected
79      000008      aap0          equ      $8          ; Yes, Enable AA0 pin on ext 'P' accesses
80      000010      aax0          equ      $10         ; Yes, Enable AA0 pin on ext 'X' accesses
81      000020      aay0          equ      $20         ; Yes, Enable AA0 pin on ext 'Y' accesses
82      000000      aswap0        equ      0          ; No, Enable address bus swap
83      000000      enpack0       equ      0          ; No, Enable packing/unpacking logic
84      000900      nadd0         equ      $000900      ; Compare 9 address bits
85      108000      msadd0        equ      $100000      ; Most significant portion of address,
86                                          ; $100000 - $107fff, to compare.
87                                          ; (0001,0000,0xxx,xxxx,xxxx,xxxx)
88      100939      AAR0_value    equ      acctype0+aahigh0+aap0+aax0+aay0+aswap0+enpack0+nadd0+msadd0
89
90      ;--- BCR value = 0x002421
91      000001      aaa0ws        equ      $1          ; Address Attribute Area 0 w/s = 1
92      000020      aaalws        equ      $20         ; Address Attribute Area 1 w/s = 1
93      000400      aaa2ws        equ      $000400      ; Address Attribute Area 2 w/s = 1
94      002000      aaa3ws        equ      $002000      ; Address Attribute Area 3 w/s = 1
95      000000      defws         equ      0          ; Default Address Area w/s = 0
96      000000      busss         equ      0          ; Bus state status = 0
97      000000      enblh         equ      0          ; Enable Bus Lock Hold = 0
98      000000      enbrh         equ      0          ; Enable Bus Request Hold = 0
99      002421      BCR_value     equ      aaa0ws+aaalws+aaa2ws+aaa3ws+defws+busss+enblh+enbrh
100
101      ;-----
102      P:000100      org          p:$100 ;Keep the program in internal RAM
103
104      memtst
105
106      ; Initialization Section
107      P:000100      08F4BD        movep   #PCTL_value,x:PCTL      ; Set PLL Control Register
108      P:000102      0E0013        movec   #$004000,OMR          ; Disable Address Attribute Priority
109      P:000104      05F43A        movec   #$004000,OMR          ; Disable Address Attribute Priority
110      P:000104      05F439        movec   #$080000,SR          ; Enable 1K Cache
111      P:000106      08F4BB        movep   #BCR_value,x:BCR      ; Set external memory wait states
112      P:000108      08F4B9        movep   #AAR0_value,x:AAR0    ; Set Address Attribute Reg0
113      P:000108      10893D        movep   #AAR0_value,x:AAR0    ; Set Address Attribute Reg0
114      P:00010C      08F4B7        movep   #AAR2_value,x:AAR2    ; Set Address Attribute Reg2
115      P:00010C      100825        movep   #AAR2_value,x:AAR2    ; Set Address Attribute Reg2
116
117      P:000110      05F420        move    #-1,m0              ; Set LINEAR addressing mode
118      P:000112      FFFFFFFF        move    #-1,m2
119      P:000114      05F423        move    #-1,m3
120      P:000116      FFFFFFFF        move    #-1,m4
121      P:000118      05F425        move    #-1,m5
122      P:000118      FFFFFFFF
123
124      P:00011A      20001B        clr    b
125      P:00011B      000000        nop
126      P:00011C      570000        move   b,x:P_MEM_FAIL_ADDRESS ; Initialize P/X:Failed Address -> $000000
127      P:00011D      570100        move   b,x:P_MEM_FAIL_WROTE   ; Initialize P/X:Expected Data -> $000000
128      P:00011E      570200        move   b,x:P_MEM_FAIL_READ    ; Initialize P/X:Data Read -> $000000
129
130      P:00011F      570300        move   b,x:Y_MEM_FAIL_ADDRESS  ; Initialize Y:Failed Address -> $000000
131      P:000120      570400        move   b,x:Y_MEM_FAIL_WROTE   ; Initialize Y:Expected Data -> $000000
132      P:000121      570500        move   b,x:Y_MEM_FAIL_READ    ; Initialize Y:Data Read -> $000000
133
134      P:000125      570900        move   b,x:MEM_PASS_COUNTER    ; Initialize Pass Counter -> $000000
135
136      main
137      ;-----
138      ; Fill memory spaces with Test Patterns
139      ;-----
140      P:000126      65F400        move   #PATT,r5              ; r5 points to Test Patterns
141      P:000128      00018C        nop
142      P:000129      000000        nop
143
144      ; Fill P/X:RAM
145      P:00012A      07DD84        move   p:(r5)+,x0           ; Get the Write Pattern
146      P:00012B      70F400        move   #PMemSize,n0         ; Get memory size
147      P:00012B      008000

```



## 64K × 24-bit Memory Based Designs

```
143 P:00012D 60F400 move #PMemStart,r0 ; Get starting address for fill
      100000
144
145 P:00012F 06D820 rep n0 ; Fill P:RAM with first data pattern
146 P:000130 075884 move x0,p:(r0)+
147
148 P:000131 070A04 move x0,p:old_p_pattern ; Save the written pattern
149
150 ; Fill Y:RAM
151 P:000132 07DD84 move p:(r5)+,x0 ; Get the Next Write Pattern
152 P:000133 72F400 move #YMemSize,n2 ; Get memory size
      008000
153 P:000135 62F400 move #YMemStart,r2 ; Get starting address for fill
      100000
154
155 P:000137 06DA20 rep n2 ; Fill Y:RAM with data pattern
156 P:000138 4C5A00 move x0,y:(r2)+
157
158 P:000139 070B04 move x0,p:old_y_pattern ; Save the written pattern
159
160 ;-----
161 ;--- Cycle through each memory space with a pattern ---
162 ;-----
163 ;-----
164 ;--- Check for expected pattern data in each RAM location ---
165 ;--- and then replace with a new data pattern. ---
166 ;--- ...This provides an address check. Since erroneous ---
167 ;--- ...addressing will cause the data to be written into ---
168 ;--- ..incorrect locations and this will be evident in ---
169 ;--- ...the next read pass. ---
170 ;-----
171 P:000149 063890 DOR #PATTN,test_mem ; Start Pattern Test Loop
      000031
172 ; Do a pass through P/X:RAM
173 P:00014B 60F400 move #PMemStart,r0 ; Get starting address of Test P:Memory
      100000
174 P:00014D 078A0E move p:old_p_pattern,a ; Get the last P test pattern -> a
175 P:00014E 07DD84 move p:(r5)+,x0 ; Get the next P test pattern -> x0
176
177 ; Test this pattern through external P/X:RAM
178 P:00014F 06D810 DOR n0,next_p_loc ; Test all external P/X:RAM locations
      000007
179 P:000151 07E085 move p:(r0),x1 ; Read P:RAM location
180 P:000152 200065 cmp x1,a ; Read data = last test pattern?
181 P:000153 05244D bne <PERR ; No, error if compare fails
182 P:000154 075884 move x0,p:(r0)+ ; Yes, Write next test pattern -> P:RAM
183
184 P:000155 070A04 move x0,p:old_p_pattern ; Save the newly written test pattern
185 P:000156 000000 nop
      next_p_loc
186
187 ; Do a pass through Y:RAM
188
189 P:000157 62F400 move #YMemStart,r2 ; Get starting address of Test Y:Memory
      100000
190 P:000159 078B0E move p:old_y_pattern,a ; Get the last Y test pattern -> a
191 P:00015A 07DD84 move p:(r5)+,x0 ; Get the next Y test pattern -> x0
192
193 ; Test this pattern through external Y:RAM
194 P:00015B 06DA10 DOR n2,next_y_loc ; Test all external Y:RAM locations
      000007
195 P:00015D 4DE200 move y:(r2),x1 ; Read Y:RAM location
196 P:00015E 200065 cmp x1,a ; Read data = last test pattern?
197 P:00015F 052445 bne <YERR ; No, error if compare fails
198 P:000160 4C5A00 move x0,y:(r2)+ ; Yes, Write next test pattern -> Y:RAM
199
200 P:000161 070B04 move x0,p:old_y_pattern ; Save the newly written test pattern
201 P:000162 000000 nop
      next_y_loc
202
203 P:00017A 000000 nop ; Time to start next test pattern
204 test_mem
205
206 ; One Pass Complete
207 ; All test patterns have been tried and passed in external RAM
208 P:00017B 518900 move x:MEM_PASS_COUNTER,b0
209 P:00017C 000009 inc b ; Update pass loop counter
210 P:00017D 000000 nop
211 P:00017E 510900 move b0,x:MEM_PASS_COUNTER
212
213 P:00017F 050F47 bra main ; Do it all over again
214
```





```
215 ; -----
216 ; PERR -- handles P/X:RAM error messaging with user
217 ;
218 ; Expected Data --> a
219 ; Read Data --> x1
220 ; Address of failure --> r0
221
222 PERR
223 P:000180 600000 move r0,x:P_MEM_FAIL_ADDRESS ; Save off address of failure
224 P:000181 560100 move a,x:P_MEM_FAIL_WROTE ; Save off expected data
225 P:000182 450200 move x1,x:P_MEM_FAIL_READ ; Save off data read
226
227 P:000183 050C00 bra * ; Dynamically HALT here
228
229 ; -----
230 ; YERR -- handles Y:RAM error messaging with user
231 ;
232 ; Expected Data --> b
233 ; Read Data --> x1
234 ; Address of failure --> r2
235
236 YERR
237 P:000184 620300 move r2,x:Y_MEM_FAIL_ADDRESS ; Save off address of failure
238 P:000185 570400 move b,x:Y_MEM_FAIL_WROTE ; Save off expected data
239 P:000186 450500 move x1,x:Y_MEM_FAIL_READ ; Save off data read
240
241 P:000187 050C00 bra * ; Dynamically HALT here
242
243 ; Memory Test Patterns
244
245 P:00018C PATT dc $000000,$FFFFFF,$AAAAAA,$555555,$2BAD2C
246 P:000191 dc $800000,$400000,$200000,$100000
247 P:000195 dc $080000,$040000,$020000,$010000
248 P:000199 dc $008000,$004000,$002000,$001000
249 P:00019D dc $000800,$000400,$000200,$000100
250 P:0001A1 dc $000080,$000040,$000020,$000010
251 P:0001A5 dc $000008,$000004,$000002,$000001
252 P:0001A9 dc $7FFFFFF,$BFFFFFF,$DFFFFFF,$EFFFFFF
253 P:0001AD dc $F7FFFF,$FBFFFF,$FDFFFF,$FEFFFF
254 P:0001B1 dc $FF7FFF,$FFBFFF,$FFDFFF,$FFEFFF
255 P:0001B5 dc $FFF7FF,$FFFBFF,$FFFDFF,$FFF7FF
256 P:0001B9 dc $FFFF7F,$FFFFBF,$FFFFDF,$FFFFEF
257 P:0001BD dc $FFFFFF7,$FFFFFFB,$FFFFFFD,$FFFFFFE
258 P:0001C1 dc $FEDCBA,$123456,$012345,$EDCBA9
259
260 P:0001C5 dc $000000,$FFFFFF,$AAAAAA,$555555,$2BAD2C
261 P:0001CA dc $800000,$400000,$200000,$100000
262 P:0001CE dc $080000,$040000,$020000,$010000
263 P:0001D2 dc $008000,$004000,$002000,$001000
264 P:0001D6 dc $000800,$000400,$000200,$000100
265 P:0001DA dc $000080,$000040,$000020,$000010
266 P:0001DE dc $000008,$000004,$000002,$000001
267 P:0001E2 dc $7FFFFFF,$BFFFFFF,$DFFFFFF,$EFFFFFF
268 P:0001E6 dc $F7FFFF,$FBFFFF,$FDFFFF,$FEFFFF
269 P:0001EA dc $FF7FFF,$FFBFFF,$FFDFFF,$FFEFFF
270 P:0001EE dc $FFF7FF,$FFFBFF,$FFFDFF,$FFF7FF
271 P:0001F2 dc $FFFF7F,$FFFFBF,$FFFFDF,$FFFFEF
272 P:0001F6 dc $FFFFFF7,$FFFFFFB,$FFFFFFD,$FFFFFFE
273 P:0001FA dc $FEDCBA,$123456,$012345,$EDCBA9
274
275 P:0001FE dc $000000,$FFFFFF,$AAAAAA,$555555,$2BAD2C
276 P:000203 dc $800000,$400000,$200000,$100000
277 P:000207 dc $080000,$040000,$020000,$010000
278 P:00020B dc $008000,$004000,$002000,$001000
279 P:00020F dc $000800,$000400,$000200,$000100
280 P:000213 dc $000080,$000040,$000020,$000010
281 P:000217 dc $000008,$000004,$000002,$000001
282 P:00021B dc $7FFFFFF,$BFFFFFF,$DFFFFFF,$EFFFFFF
283 P:00021F dc $F7FFFF,$FBFFFF,$FDFFFF,$FEFFFF
284 P:000223 dc $FF7FFF,$FFBFFF,$FFDFFF,$FFEFFF
285 P:000227 dc $FFF7FF,$FFFBFF,$FFFDFF,$FFF7FF
286 P:00022B dc $FFFF7F,$FFFFBF,$FFFFDF,$FFFFEF
287 P:00022F dc $FFFFFF7,$FFFFFFB,$FFFFFFD,$FFFFFFE
288 P:000233 dc $FEDCBA,$123456,$012345,$EDCBA9
289
290 000038 PATTN equ ((*-PATT)/3)-1
291
292 end memtst
0 Errors
0 Warnings
```

### 5.3 32K × 24-bit ‘X’ and 32K × 24-bit ‘Y’ Fast SRAM Example

This section describes a 32K × 24-bit ‘X’ and 32K × 24-bit ‘Y’ memory space, asynchronous Fast SRAM implementation using GSI’s GS71024T device (see **Figure 5-6** for memory map layout, **Figure 5-1** for the block diagram, and **Example 5-2** for the example code).

	Program	X Data	Y Data	
\$FFFFFFF	Reserved	Internal I/O	Internal I/O	\$FFFFFFC0
\$FF00C0		External	External	\$FFF000
\$FF0000	Bootstrap ROM	Reserved	Reserved	\$FF0000
	External	External	External	\$108000
				External 32K SRAM
\$001000	External	External	External	
\$000C00	Internal 1K Cache			
	Internal 3K SRAM	Internal 2K SRAM	Internal 2K SRAM	\$000800
\$000000				

**Figure 5-6.** 32K × 24-bit ‘X’ and 32K × 24-bit ‘Y’ Memory Map

The DSP core runs at 80 MHz and the input frequency source is a 4.000 MHz crystal. The asynchronous Fast SRAM requires an access time equal to or less than 12.4 nS to satisfy the DSP’s one wait state external memory requirements.

#### 5.3.1 DSP56303 Port A Timing Requirements and Register Settings

For optimal use of the 32K × 24-bit ‘X’ and 32K × 24-bit ‘Y’ space memory configuration, set up the following DSP control registers.

Set the core speed of the DSP for optimum processor and memory performance using the DSP PLL and Clock Generation register (PCTL). In this example, the DSP core runs at 80 MHz and the input frequency source is a 4.000 MHz crystal. The PCTL register value combines the following bits for each feature:

- Desired Core Frequency = 80 MHz
- Given the External Frequency = 4.000 MHz
- Predivider value = 1, Bits 20–23 = \$0
- Low-power Divider value = 1, Bits 12–14 = \$0
- VCO Multiplication value = 20, Bits 0–11 = \$013
- Crystal less than 200 kHz, Bit 15 = 0
- Disable XTAL drive output, Bit 16 = 0
- PLL runs during STOP, Bit 17 = 1
- Enable PLL operation, Bit 18 = 1
- Disable core clock output, Bit 19 = 1

The value loaded into the PCTL register is be \$0E0013.

Address Attribute Pin 0 enables, via Fast SRAM  $\overline{CE1}$ , the external 64K SRAM bank accesses in the address range \$100000 to \$107FFF during X data and Y data space requests. Configure the memory address space requirements for the Address Attribute Pin 0 using Address Attribute Register 0 (AAR0). The AAR0 value combines the following bits for each feature:

- Specify the external memory access type as asynchronous SRAM, Bits 0–1 = \$1.
- Pull the AA pin high when selected, Bit 2 = 0.
- Activate the AA pin during external program space accesses, Bit 3 = 0.
- Activate the AA pin during external X data space accesses, Bit 4 = 1.
- Activate the AA pin to during external Y data space accesses, Bit 5 = 1.
- Move the eight least significant bits of the address to the eight most significant bits of the external address bus, Bit 6 = 0.
- Enable the internal packing/unpacking logic during external DMA accesses, Bit 7 = 0.
- Specify the number of address bits to compare, Bits 8–11 = \$9
- Specify the most significant portion of the address to compare, Bits 12–23 = \$100

The value loaded into the AAR0 is \$100931.

---

Address Attribute Pin 2 selects, via Fast SRAM A15, the external 64K SRAM bank during accesses in the address range \$100000 to \$107FFF to differentiate X data requests from Y data space requests. Configure the memory address space requirements for the Address Attribute Pin 2 using Address Attribute Register 2 (AAR2). The AAR2 value combines the following bits for each feature:

- Specify the external memory access type as asynchronous SRAM, Bits 0–1 = \$1.
- Pull the AA pin high when selected, Bit 2 = 1.
- Activate the AA pin during external program space accesses, Bit 3 = 0.
- Activate the AA pin during external X data space accesses, Bit 4 = 0.
- Activate the AA pin during external Y data space accesses, Bit 5 = 1.
- Move the eight least significant bits of the address to the eight most significant bits of the external address bus, Bit 6 = 0.
- Enable the internal packing/unpacking logic during external DMA accesses, Bit 7 = 0.
- Specify the number of address bits to compare, Bits 8–11 = \$9
- Specify the most significant portion of the address to compare, Bits 12–23 = \$100

The value loaded into the AAR2 is \$100925.

Select the proper number of wait states for the memory configuration using the Bus Control Register (BCR). The BCR value combines the following bits for each feature:

- Address attribute area 0 wait states, Bits 0–4 = \$1
- Address attribute area 1 wait states, Bits 5–9 = \$1
- Address attribute area 2 wait states, Bits 10–12 = \$1
- Address attribute area 3 wait states, Bits 13–15 = \$1
- Default address area wait states, Bits 16–20 = \$0
- Bus state status, Bit 21 = 0
- Enable Bus Lock Hold, Bit 22 = 0
- Enable Bus Request Hold, Bit 23 = 0

The value loaded into the BCR is \$002421.

Configure the operating mode and external memory controls using the Operating Mode Register (OMR). The OMR value combines the following bits for each feature:

---

- MA–MD bits specify the DSP operating mode, Bits 0–3 = \$0.
- External Bus Disable Bit disables the external bus controller for power conservation when external memory is not used, Bit 4 = \$0.
- Memory Switch Mode bit reconfigures internal memory spaces, Bit 7 = \$0.
- Transfer Acknowledge Synchronize Select bit selects the synchronization method for the Transfer Acknowledge (TA) pin, Bit 11 = \$0.
- Bus Release Timing bit selects between a fast and slow bus release of the  $\overline{BB}$  pin, Bit 12 = \$0.
- Address Attribute Priority Disable bit allows the Address Attribute pins, AA0–AA3, to be used in any combination, Bit 14 = \$1.
- All other OMR bits are selected for their defaults of \$000000.

The value loaded into the OMR is \$004000.

Configure the memory mode of the DSP using the Status Register (SR). The SR value combines the following bits for each feature:

- Sixteen-Bit Compatibility mode enables full compatibility to object code written for the DSP56000 family of DSPs, Bit 13 = \$0.
- Instruction Cache Enable bit enables the instruction cache controller and changes the last 1K of internal program memory into cache memory, Bit 19 = \$1.
- All other Status Register bits are selected for their defaults of \$000000.

The value loaded into the SR is \$080000, which is the value loaded during reset.

---



**Example 5-2. 32K x 24-bit 'X' and 32K x 24-bit 'Y' Space Fast SRAM Memory Exercise**

Motorola DSP56300 Assembler Version 6.0.1.6 97-01-25 09:27:34 asram7.asm

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
page 132,60,3,3,
;
; ASRAM7.ASM - Simple program to test
; 32Kx24-bits of X-Data and 32Kx24-bits of Y-Data
; memory using a DSP56303
;
; The program uses Internal P:RAM to test:
; 24-bit External X: & Y:RAM from $100000 - $107FFF @ 1w/s
XMemStart equ $100000
XMemEnd equ $108000
XMemSize equ XMemEnd-XMemStart
YMemStart equ $100000
YMemEnd equ $108000
YMemSize equ YMemEnd-YMemStart
;--- Program Specific Storage Locations (X DATA SPACE)
X_MEM_FAIL_ADDRESS equ $000003
X_MEM_FAIL_WROTE equ $000004
X_MEM_FAIL_READ equ $000005
Y_MEM_FAIL_ADDRESS equ $000006
Y_MEM_FAIL_WROTE equ $000007
Y_MEM_FAIL_READ equ $000008
MEM_PASS_COUNTER equ $00000C
old_x_pattern equ $00000E ; Last written pattern to X:RAM
old_y_pattern equ $00000F ; Last written pattern to Y:RAM
;--- DSP56303 Control Registers (X I/O SPACE)
BCR equ $FFFFFFB ; Bus Control Register
PCTL equ $FFFFFFD ; PLL Control Register
AAR3 equ $FFFFFF6 ; Address Attribute Register #3
AAR2 equ $FFFFFF7 ; Address Attribute Register #2
AAR1 equ $FFFFFF8 ; Address Attribute Register #1
AAR0 equ $FFFFFF9 ; Address Attribute Register #0
;--- PCTL value = 0x0E0013
prediv equ 0 ; Pre-Divider = 1
lowdiv equ 0 ; Low Power Divider = 1
pllmul equ 19 ; VCO Mult = 20; (19+1)*4.00MHz = 80.00 MHz
crystal equ 0 ; No, Crystal not less than 200kHz
disXTAL equ 0 ; No, do not disable crystal use
pllstop equ $020000 ; Yes, PLL runs during STOP
enpll equ $040000 ; Yes, enable PLL operation
disclk equ $080000 ; Yes, disable CORE clock output
PCTL_value equ prediv+lowdiv+pllmul+crystal+disXTAL+pllstop+enpll+disclk
;--- AAR2 value = 0x100925
acctype2 equ 1 ; External Memory access type = 0x1
aahigh2 equ $4 ; Enable AA2 pin to be high when selected
aap2 equ 0 ; No, Enable AA2 pin on ext 'P' accesses
aax2 equ 0 ; No, Enable AA2 pin on ext 'X' accesses
aay2 equ $20 ; Yes, Enable AA2 pin on ext 'Y' accesses
aswap2 equ 0 ; No, Enable address bus swap
enpack2 equ 0 ; No, Enable packing/unpacking logic
nadd2 equ $000900 ; Compare 9 address bits
msadd2 equ $100000 ; Most significant portion of address,
; $100000 - $107fff, to compare.
; (0001, 0000, 0xxx, xxxx, xxxx, xxxx)
AAR2_value equ acctype2+aahigh2+aap2+aax2+aay2+aswap2+enpack2+nadd2+msadd2
```

```

74          ;--- AAR0 value = 0x100931
75 000001  acctype0      equ    1          ; External Memory access type = 0x1
76 000004  aahigh0       equ    0          ; Enable AA0 pin to be low when selected
77 000000  aap0          equ    0          ; No, Enable AA0 pin on ext 'P' accesses
78 000010  aax0         equ    $10         ; Yes, Enable AA0 pin on ext 'X' accesses
79 000000  aay0         equ    $20         ; Yes, Enable AA0 pin on ext 'Y' accesses
80 000000  aswap0       equ    0          ; No, Enable address bus swap
81 000000  enpack0      equ    0          ; No, Enable packing/unpacking logic
82 000900  nadd0        equ    $000900    ; Compare 9 address bits
83 100000  msadd0       equ    $100000    ; Most significant portion of address,
84                                     ; $100000 - $107fff, to compare.
85                                     ; (0001,0000,0xxx,xxxx,xxxx,xxxx)
86 100931   AAR0_value equ    acctype0+aahigh0+aap0+aax0+aay0+aswap0+enpack0+nadd0+msadd0
87
88          ;--- BCR value = 0x002421
89 000001  aaa0ws       equ    1          ; Address Attribute Area 0 w/s = 1
90 000020  aaalws       equ    $20         ; Address Attribute Area 1 w/s = 1
91 000400  aaa2ws       equ    $000400    ; Address Attribute Area 2 w/s = 1
92 002000  aaa3ws       equ    $002000    ; Address Attribute Area 3 w/s = 1
93 000000  defws        equ    0          ; Default Address Area w/s = 0
94 000000  busss        equ    0          ; Bus state status = 0
95 000000  enblh        equ    0          ; Enable Bus Lock Hold = 0
96 000000  enbrh        equ    0          ; Enable Bus Request Hold = 0
97 002421  BCR_value    equ    aaa0ws+aaalws+aaa2ws+aaa3ws+defws+busss+enblh+enbrh
98
99          ;-----
100 P:000100          org      p:$100 ; Keep the program in internal RAM
101
102          memtst
103
104          ; Initialization Section
105 P:000100 08F4BD      movep   #PCTL_value,x:PCTL ; Set PLL Control Register
106          0E0013
106 P:000102 05F43A      movec   #004000,OMR ; Disable Address Attribute Priority
107          004000
107 P:000104 05F439      movec   #080000,SR ; Enable 1K Cache
108          080000
108 P:000106 08F4BB      movep   #BCR_value,x:BCR ; Set external memory wait states
109          002421
109 P:000108 08F4B9      movep   #AAR0_value,x:AAR0 ; Set Address Attribute Reg0
110          100915
111 P:00010C 08F4B7      movep   #AAR2_value,x:AAR2 ; Set Address Attribute Reg2
112          100925
113 P:000110 05F420      move    #-1,m0 ; Set LINEAR addressing mode
114          FFFFFFFF
115 P:000112 05F421      move    #-1,m1
116          FFFFFFFF
116 P:000114 05F422      move    #-1,m2
117          FFFFFFFF
117 P:000116 05F423      move    #-1,m3
118          FFFFFFFF
118 P:000118 05F424      move    #-1,m4
119          FFFFFFFF
119 P:00011A 05F425      move    #-1,m5
120          FFFFFFFF
121 P:00011C 20001B      clr     b
122 P:00011D 000000      nop
123
124 P:000121 570300      move    b,x:X_MEM_FAIL_ADDRESS ;Initialize X:Failed Address -> $000000
125 P:000122 570400      move    b,x:X_MEM_FAIL_WROTE ;Initialize X:Expected Data -> $000000
126 P:000123 570500      move    b,x:X_MEM_FAIL_READ ;Initialize X:Data Read -> $000000
127
128 P:000124 570600      move    b,x:Y_MEM_FAIL_ADDRESS ;Initialize Y:Failed Address -> $000000
129 P:000125 570700      move    b,x:Y_MEM_FAIL_WROTE ;Initialize Y:Expected Data -> $000000
130 P:000126 570800      move    b,x:Y_MEM_FAIL_READ ;Initialize Y:Data Read -> $000000
131
132 P:00012A 570C00      move    b,x:MEM_PASS_COUNTER ;Initialize Pass Counter -> $000000
133
134          main
135          ;-----
136          ; Fill memory spaces with Test Patterns
137          ;-----
138 P:00012B 65F400      move    #PATT,r5 ; r5 points to Test Patterns
139          0001A9
139 P:00012D 000000      nop
140 P:00012E 000000      nop
141
    
```



## 64K × 24-bit Memory Based Designs

```
142             ; Fill X:RAM
143 P:000137 07DD84      move    p:(r5)+,x0      ; Get the Next Write Pattern
144 P:000138 71F400      move    #XMemSize,n1    ; Get memory size
145             008000
145 P:00013A 61F400      move    #XMemStart,r1   ; Get starting address for fill
146             100000
146
147 P:00013C 06D920      rep     n1              ; Fill X:RAM with data pattern
148 P:00013D 445900      move    x0,x:(r1)+
149
150 P:00013E 070E04      move    x0,p:old_x_pattern ; Save the written pattern
151
152             ; Fill Y:RAM
153 P:00013F 07DD84      move    p:(r5)+,x0      ; Get the Next Write Pattern
154 P:000140 72F400      move    #YMemSize,n2    ; Get memory size
155             008000
155 P:000142 62F400      move    #YMemStart,r2   ; Get starting address for fill
156             100000
156
157 P:000144 06DA20      rep     n2              ; Fill Y:RAM with data pattern
158 P:000145 4C5A00      move    x0,y:(r2)+
159
160 P:000146 070F04      move    x0,p:old_y_pattern ; Save the written pattern
161
162             ; -----
163             ; --- Cycle through each memory space with a pattern ---
164             ; -----
165             ; -----
166             ; --- Check for expected pattern data in each RAM location ---
167             ; --- and then replace with a new data pattern. ---
168             ; --- ...This provides an address check. Since erroneous ---
169             ; --- ...addressing will cause the data to be written into ---
170             ; --- ...incorrect locations and this will be evident in ---
171             ; --- ...the next read pass. ---
172             ; -----
173
174             ; Do a pass through X:RAM
175 P:000164 61F400      move    #XMemStart,r1   ; Get starting address of Test X:Memory
176             100000
176 P:000166 078E0E      move    p:old_x_pattern,a ; Get the last X test pattern -> a
177 P:000167 07DD84      move    p:(r5)+,x0      ; Get the next X test pattern -> x0
178
179             ; Test this pattern through external X:RAM
180 P:000168 06D910      DOR    n1,next_x_loc    ; Test all external X:RAM locations
181             000007
181 P:00016A 45E100      move    x:(r1),x1       ; Read X:RAM location
182 P:00016B 200065      cmp     x1,a            ; Read data = last test pattern?
183 P:00016C 052451      bne    <XERR           ; No, error if compare fails
184 P:00016D 445900      move    x0,x:(r1)+      ; Yes, Write next test pattern -> X:RAM
185
186 P:00016E 070E04      move    x0,p:old_x_pattern ; Save the newly written test pattern
187 P:00016F 000000      nop
188             next_x_loc
189
190             ; Do a pass through Y:RAM
191 P:000170 62F400      move    #YMemStart,r2   ; Get starting address of Test Y:Memory
192             100000
192 P:000172 078F0E      move    p:old_y_pattern,a ; Get the last Y test pattern -> a
193 P:000173 07DD84      move    p:(r5)+,x0      ; Get the next Y test pattern -> x0
194
195             ; Test this pattern through external Y:RAM
196 P:000174 06DA10      DOR    n2,next_y_loc    ; Test all external Y:RAM locations
197             000007
197 P:000176 4DE200      move    y:(r2),x1       ; Read Y:RAM location
198 P:000177 200065      cmp     x1,a            ; Read data = last test pattern?
199 P:000178 052449      bne    <YERR           ; No, error if compare fails
200 P:000179 4C5A00      move    x0,y:(r2)+      ; Yes, Write next test pattern -> Y:RAM
201
202 P:00017A 070F04      move    x0,p:old_y_pattern ; Save the newly written test pattern
203 P:00017B 000000      nop
204             next_y_loc
205
206 P:000193 000000      nop                    ; Time to start next test pattern
207
208             test_mem
209
210             ;
211             ; One Pass Complete
212             ; All test patterns have been tried and passed in external RAM
212 P:000194 518C00      move    x:MEM_PASS_COUNTER,b0
213 P:000195 000009      inc     b                ; Update pass loop counter
214 P:000196 000000      nop
```



```

215 P:000197 510C00      move  b0,x:MEM_PASS_COUNTER
216
217 P:000198 050F13      bra   main                ; Do it all over again
218
219
220
221 ; -----
222 ; XERR -- handles X:RAM error messaging with user
223 ;
224 ; Expected Data --> a
225 ; Read Data --> x1
226 ; Address of failure --> r1
227
228 XERR
229 P:00019D 610300      move  r1,x:X_MEM_FAIL_ADDRESS ; Save off address of failure
230 P:00019E 560400      move  a,x:X_MEM_FAIL_WROTE    ; Save off expected data
231 P:00019F 450500      move  x1,x:X_MEM_FAIL_READ    ; Save off data read
232
233 P:0001A0 050C00      bra   *                    ; Dynamically HALT here
234
235 ; -----
236 ; YERR -- handles Y:RAM error messaging with user
237 ;
238 ; Expected Data --> b
239 ; Read Data --> x1
240 ; Address of failure --> r2
241
242 YERR
243 P:0001A1 620600      move  r2,x:Y_MEM_FAIL_ADDRESS ; Save off address of failure
244 P:0001A2 570700      move  b,x:Y_MEM_FAIL_WROTE    ; Save off expected data
245 P:0001A3 450800      move  x1,x:Y_MEM_FAIL_READ    ; Save off data read
246
247 P:0001A4 050C00      bra   *                    ; Dynamically HALT here
248
249
250
251 ; Memory Test Patterns
252
253 P:0001A9 PATT  dc   $000000,$FFFFFF,$AAAAAA,$555555,$2BAD2C
254 P:0001AE          dc   $800000,$400000,$200000,$100000
255 P:0001B2          dc   $080000,$040000,$020000,$010000
256 P:0001B6          dc   $008000,$004000,$002000,$001000
257 P:0001BA          dc   $000800,$000400,$000200,$000100
258 P:0001BE          dc   $000080,$000040,$000020,$000010
259 P:0001C2          dc   $000008,$000004,$000002,$000001
260 P:0001C6          dc   $7FFFFFF,$BFFFFFF,$DFFFFFF,$EFFFFFF
261 P:0001CA          dc   $F7FFFF,$FBFFFF,$FDFFFF,$FEFFFF
262 P:0001CE          dc   $FF7FFF,$FFBFFF,$FFDFFF,$FFEFFF
263 P:0001D2          dc   $FFF7FF,$FFFBFF,$FFFDFF,$FFF7FF
264 P:0001D6          dc   $FFFF7F,$FFFFBF,$FFFFDF,$FFFFEF
265 P:0001DA          dc   $FFFFFF7,$FFFFFFB,$FFFFFFD,$FFFFFFE
266 P:0001DE          dc   $FEDCBA,$123456,$012345,$EDCBA9
267
268 P:0001E2          dc   $000000,$FFFFFF,$AAAAAA,$555555,$2BAD2C
269 P:0001E7          dc   $800000,$400000,$200000,$100000
270 P:0001EB          dc   $080000,$040000,$020000,$010000
271 P:0001EF          dc   $008000,$004000,$002000,$001000
272 P:0001F3          dc   $000800,$000400,$000200,$000100
273 P:0001F7          dc   $000080,$000040,$000020,$000010
274 P:0001FB          dc   $000008,$000004,$000002,$000001
275 P:0001FF          dc   $7FFFFFF,$BFFFFFF,$DFFFFFF,$EFFFFFF
276 P:000203          dc   $F7FFFF,$FBFFFF,$FDFFFF,$FEFFFF
277 P:000207          dc   $FF7FFF,$FFBFFF,$FFDFFF,$FFEFFF
278 P:00020B          dc   $FFF7FF,$FFFBFF,$FFFDFF,$FFF7FF
279 P:00020F          dc   $FFFF7F,$FFFFBF,$FFFFDF,$FFFFEF
280 P:000213          dc   $FFFFFF7,$FFFFFFB,$FFFFFFD,$FFFFFFE
281 P:000217          dc   $FEDCBA,$123456,$012345,$EDCBA9
282
283 P:00021B          dc   $000000,$FFFFFF,$AAAAAA,$555555,$2BAD2C
284 P:000220          dc   $800000,$400000,$200000,$100000
285 P:000224          dc   $080000,$040000,$020000,$010000
286 P:000228          dc   $008000,$004000,$002000,$001000
287 P:00022C          dc   $000800,$000400,$000200,$000100
288 P:000230          dc   $000080,$000040,$000020,$000010
289 P:000234          dc   $000008,$000004,$000002,$000001
290 P:000238          dc   $7FFFFFF,$BFFFFFF,$DFFFFFF,$EFFFFFF
291 P:00023C          dc   $F7FFFF,$FBFFFF,$FDFFFF,$FEFFFF
292 P:000240          dc   $FF7FFF,$FFBFFF,$FFDFFF,$FFEFFF
293 P:000244          dc   $FFF7FF,$FFFBFF,$FFFDFF,$FFF7FF
294 P:000248          dc   $FFFF7F,$FFFFBF,$FFFFDF,$FFFFEF
295 P:00024C          dc   $FFFFFF7,$FFFFFFB,$FFFFFFD,$FFFFFFE
    
```



## 64K × 24-bit Memory Based Designs

---

```
296 P:000250 dc $FEDCBA,$123456,$012345,$EDCBA9
297
298 P:000254 dc $000000,$FFFFFF,$AAAAAA,$555555,$2BAD2C
299 P:000259 dc $800000,$400000,$200000,$100000
300 P:00025D dc $080000,$040000,$020000,$010000
301 P:000261 dc $008000,$004000,$002000,$001000
302 P:000265 dc $000800,$000400,$000200,$000100
303 P:000269 dc $000080,$000040,$000020,$000010
304 P:00026D dc $000008,$000004,$000002,$000001
305 P:000271 dc $7FFFFFF,$BFFFFFF,$DFFFFFF,$EFFFFFF
306 P:000275 dc $F7FFFF,$FBFFFF,$FDFFFF,$FEFFFF
307 P:000279 dc $FF7FFF,$FFBFFF,$FFDFFF,$FFEFFF
308 P:00027D dc $FFF7FF,$FFFBFF,$FFFDFF,$FFF7FF
309 P:000281 dc $FFFF7F,$FFFFBF,$FFFFDF,$FFFFEF
310 P:000285 dc $FFFFF7,$FFFFFB,$FFFFFD,$FFFFFE
311 P:000289 dc $FEDCBA,$123456,$012345,$EDCBA9
312
313 000038 PATTN equ ((*-PATT)/4)-1
314
315 end memtst

0 Errors
0 Warnings
```

---

## 6 DSP Memory Space Configurations

This section summarizes the memory design implementations presented in the previous sections.

As illustrated in Sections 3, 4 and 5, a common hardware memory design using the DSP56303's Memory Expansion Port allows the memory bank to be logically configured for use in various memory space arrangements by simply setting up the Memory Expansion Port's Address Attribute Control Registers. Configuring the Memory Expansion Port's Address Attribute Control Registers, the memory bank can accommodate thirteen different memory space arrangements if two Address Attribute control lines are used:

1. Memory Bank 'P' Space Fast SRAM
2. Memory Bank 'X' Space Fast SRAM
3. Memory Bank 'Y' Space Fast SRAM
4. Memory Bank 'P'/'X' Space Fast SRAM
5. Memory Bank 'P'/'Y' Space Fast SRAM
6. Memory Bank 'X'/'Y' Space Fast SRAM
7. Memory Bank 'P'/'X'/'Y' Space Fast SRAM
8. Memory Bank/2 'P'/'X' and Memory Bank/2 'Y' Space Fast SRAM
9. Memory Bank/2 'P'/'Y' and Memory Bank/2 'X' Space Fast SRAM
10. Memory Bank/2 'P' and Memory Bank/2 'X'/'Y' Space Fast SRAM
11. Memory Bank/2 'P' and Memory Bank/2 'X' Space Fast SRAM
12. Memory Bank/2 'P' and Memory Bank/2 'Y' Space Fast SRAM
13. Memory Bank/2 'X' and Memory Bank/2 'Y' Space Fast SRAM

### 6.1 Fast SRAM Advantages

Fast SRAM provides the designer with a high performance memory implementation allowing flexible memory configuration capabilities for the DSP56300 family. Fast SRAM allows the DSP to access external memory at its maximum speed with the minimum number of wait states (i.e., one wait state). For Fast SRAM speed selection, the critical timing specification used is typically based on the Fast SRAM's data access time,  $t_{AA}$ . However, all timing specifications must be met and should always be reviewed for compliance.

Fast SRAM allows the designer a high degree of memory configuration flexibility. Using the DSP's Address Attribute lines, a 24-bit Fast SRAM memory bank can be configured to satisfy various memory system implementations:

- 'P' memory space
  - 'X' memory space
  - 'Y' memory space
  - Unified 'P' & 'X' memory space
-

- Unified 'P' & 'Y' memory space
  - Unified 'X' & 'Y' memory space
  - Unified 'P' & 'X' with separate 'Y' memory spaces
  - Unified 'P' & 'Y' with separate 'X' memory spaces
  - Separate 'P' with unified 'X' and 'Y' memory spaces
  - Unified 'P', 'X' and 'Y' memory spaces
  - Separate 'P' and 'X' memory spaces
  - Separate 'P' and 'Y' memory spaces
  - Separate 'X' and 'Y' memory spaces
  - Separate 'P', 'X' and 'Y' memory spaces
-









## **How to Reach Us:**

### **Home Page:**

[www.freescale.com](http://www.freescale.com)

### **E-mail:**

[support@freescale.com](mailto:support@freescale.com)

### **USA/Europe or Locations Not Listed:**

Freescale Semiconductor  
Technical Information Center, CH370  
1300 N. Alma School Road  
Chandler, Arizona 85224  
+1-800-521-6274 or +1-480-768-2130  
[support@freescale.com](mailto:support@freescale.com)

### **Europe, Middle East, and Africa:**

Freescale Halbleiter Deutschland GmbH  
Technical Information Center  
Schatzbogen 7  
81829 Muenchen, Germany  
+44 1296 380 456 (English)  
+46 8 52200080 (English)  
+49 89 92103 559 (German)  
+33 1 69 35 48 48 (French)  
[support@freescale.com](mailto:support@freescale.com)

### **Japan:**

Freescale Semiconductor Japan Ltd.  
Headquarters  
ARCO Tower 15F  
1-8-1, Shimo-Meguro, Meguro-ku,  
Tokyo 153-0064  
Japan  
0120 191014 or +81 3 5437 9125  
[support.japan@freescale.com](mailto:support.japan@freescale.com)

### **Asia/Pacific:**

Freescale Semiconductor Hong Kong Ltd.  
Technical Information Center  
2 Dai King Street  
Tai Po Industrial Estate  
Tai Po, N.T., Hong Kong  
+800 2666 8080  
[support.asia@freescale.com](mailto:support.asia@freescale.com)

### **For Literature Requests Only:**

Freescale Semiconductor Literature Distribution Center  
P.O. Box 5405  
Denver, Colorado 80217  
1-800-441-2447 or 303-675-2140  
Fax: 303-675-2150  
[LDCForFreescaleSemiconductor@hibbertgroup.com](mailto:LDCForFreescaleSemiconductor@hibbertgroup.com)

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document. Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

