

# Interfacing EPROM and EEPROM Memory with Motorola's DSP56300 Family of Digital Signal Processors

by Phil Brewer

## 1 Overview

This application note describes how to interface electrically erasable and programmable read only memory (EEPROM) and erasable programmable read only memory (EPROM) with 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 parts. Taking advantage of the DSP's available control lines makes virtually glueless external memory interfaces possible, thereby reducing the cost and using fewer parts in an application's memory design.

Specifically, this application note describes implementations for EPROM and EEPROM memory 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>Overview.....</b>	<b>1-1</b>
1.1	DSP56300 Family .....	1-2
1.2	Application Note Organization.....	1-2
1.3	Non-volatile Memory Families ....	1-3
1.4	References .....	1-4
<b>2</b>	<b>Interface Overview .....</b>	<b>2-1</b>
2.1	DSP56300 External Memory Timing .....	2-1
2.2	DSP Memory Control Registers ...	2-4
<b>3</b>	<b>EEPROM Memory .....</b>	<b>3-1</b>
3.1	2K x 8-bit BOOT EEPROM Example .....	3-1
3.2	32K x 8-bit BOOT/Overlay EEPROM Example .....	3-11
<b>4</b>	<b>Erasable Programmable Read Only Memory .....</b>	<b>4-1</b>
4.1	128K x 24-bit BOOT, 'P', 'X' and 'Y' EPROM Example .....	4-1
4.2	16K x 8-bit BOOT EPROM Example .....	4-12
4.3	512K x 8-bit BOOT/Overlay EPROM Example .....	4-19
<b>5</b>	<b>Advantages .....</b>	<b>5-1</b>
5.1	Flexible Memory Configuration Capabilities .....	5-1
5.2	Disadvantages .....	5-3
5.3	Speed Selection.....	5-3

## 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 five parts:

1. Introduction
2. DSP56303 memory expansion port, Port A, timing characteristics and use
3. EEPROM memory designs
  - 2K x 8-bit Boot example using one 2K x 8-bit 5.0V memory
  - 32K x 8-bit Boot/Overlay example using one 32K x 8-bit 3.3V memory

4. EPROM Memory Designs;
  - 128K x 24-bit BOOT, 128K x 24-bit Program, 128K x 24-bit 'X' and 128K x 24-bit 'Y' Memory example using three 512K x 8-bit 5.0V memories,
  - 16K x 8-bit Boot example using one 16K x 8-bit 5.0V memory,
  - 512K x 8-bit Boot/Overlay example using one 512K x 8-bit 3.3V memory.
5. Summary of advantages and disadvantages of using EPROM and EEPROM.

This document is a condensation of the memory interface details specified in the *DSP56300 24-Bit Digital Signal Processor Family Manual*, user and technical data manuals, and AMD's EPROM and ATMEL's EEPROM/EPROM data manuals.

## 1.3 Non-volatile Memory Families

The following non-volatile memory families were considered for this application note:

- Mask ROM
- PROM
- OTPROM
- EPROM
  - 5.0V
  - 3.3V
- EEPROM
  - 5.0V
  - 3.3V
  - 2.7V
- Serial EEPROM
  - 5.0V
  - 3.3V
  - 2.7V
  - 2.5V
  - 1.8V

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 document give designers the insight necessary to implement other memory families and memory types, if needed, with DSP56300 family devices.

## 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 DSP56301 24-bit Digital Signal Processor User's Manual (DSP56301UM/AD)*, Motorola, 1995.

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

*Motorola DSP56302EVM User's Manual*, Motorola, 1996.

*Motorola DSP56303EVM User's Manual*, Motorola, 1996.

*Atmel Corporation Nonvolatile Memory Data Book*, Atmel, May 1996.

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

*Quality Semiconductor QuickSwitch® Products Databook*, Quality Semiconductor, 1995.

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

## 2 Interface Overview

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

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.

### 2.1 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.** Asynchronous Memory Speeds Based on DSP Wait States

External Clock (MHz)	DF	MF	PDF	WS	Core Clock (MHz)	TC (nS)		$t_{AA}$ - max (nS)		$t_{AW}$ -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

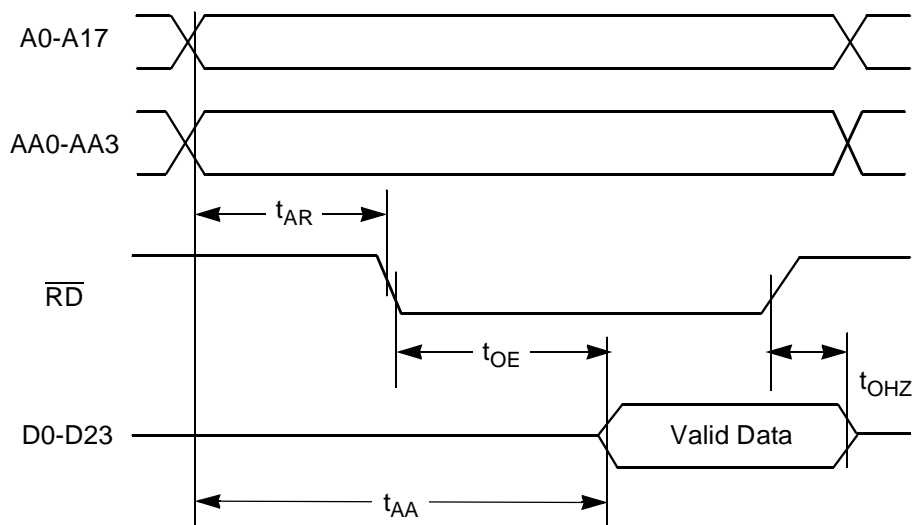
DF = PLL Division Factor  
 MF = PLL Multiplication Factor  
 PDF = PLL Pre-Division Factor  
 WS = wait states  
 TC = Clock Cycle Time  
 $t_{AA}$  = Data access time (i.e., address and AA valid to input data valid)  
 $t_{AW}$  = Data access time (i.e., address and AA valid to  $\overline{WR}$  deassertion)

### 2.1.1 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

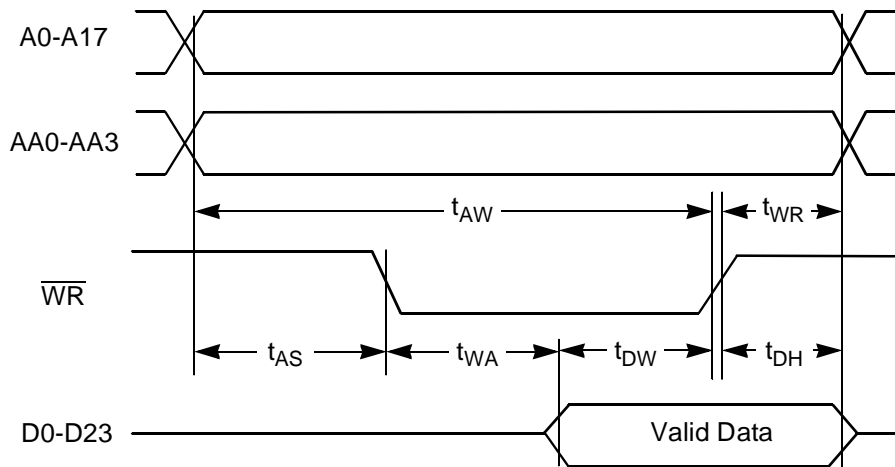
### 2.1.2 DSP56300 External Memory Bus Asynchronous Write Timing

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

## 2.2 DSP 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



## 2.2.1 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.

## 2.2.2 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

## 2.2.3 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.

## 2.2.4 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.

## 2.2.5 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 EEPROM Memory

EEPROM provides non-volatile program and data storage, which is in-circuit reprogrammable, and offers relatively fast access times. However, even the fastest EEPROM memories would require a DSP core running at 80 MHz to generate external memory wait states. With each wait state equivalent to one clock period of the DSP core, one wait state for a core running at 80 MHz is roughly 12.5nS.

When the external memory device must reside in a stable address during an entire external access, a DSP56300 family device incurs an automatic one wait state penalty. Since EEPROM devices have this address stability requirement, the DSP operates with at least one wait state when using these external memories.

The following two EEPROM memory design examples illustrate how easy and flexible it is to use an EEPROM device with the Motorola DSP56300 family. The first example, 2K x 8-bit BOOT EEPROM, shows one solution for an embedded system that loads a program from EEPROM at BOOT time into the DSP's internal RAM and then executes it. The second example, 32K x 8-bit BOOT EEPROM, shows another solution for a Bootable embedded system. In this implementation, the DSP loads internal program RAM from the EEPROM at BOOT time and then overlays new programs and data from the EEPROM when needed.

### 3.1 2K x 8-bit BOOT EEPROM Example

This section describes a 2K x 8-bit 'P' memory space, Boot-strap, BOOT, EEPROM implementation using ATMEL's AT28C16-25 (see **Figure 3-1** for the memory map layout, **Figure 3-2** for the block diagram, **Figure 3-5** for the schematic, and **Example 3-1** for the code example). Using an EEPROM in this configuration, a program placed in the EEPROM can be loaded into the DSP at Boot time and run. The running program can then save an updated version of the program back into the EEPROM for later use.

For this example, the DSP core runs at 80 MHz and the input frequency source is a 4.000 MHz crystal. For a 250nS EEPROM, 20 wait states are required. However, during the DSP's boot sequence, the DSP generates 31 wait states with the core running at 4.0 MHz, sufficient to accommodate a 7.7mS access time device. This 5.0V device is organized as 2K x 8-bits with a 250nS access time. One memory device is used to achieve the 8-bit wide BOOT bus.

Level conversion to and from 3.3V and 5V is necessary on the 8-bit data bus to accommodate the 5V memory devices. This is accomplished by using one of Quality Semiconductor's QS3245 QuickSwitch<sup>®</sup> 8-bit bus switches. This switch allows the connection of 3.3V CMOS logic (the DSP data bus) on one side and 5V TTL-compatible logic (the memory devices) on the other side, effectively providing a 3.3V-to-5V level conversion without adding any significant, 0.25nS, propagation delay.

During RESET with Mode 1 selected, the DSP BOOT code configures Address Attribute Line 1 for program accesses in the address range from \$D00000 to \$DFFFFFF. The BOOT code then loads bytes from the EEPROM, packs them into 24-bit words and stores them into Program RAM. The first word, three packed bytes, read from the EEPROM indicates the number of words to load. The second word from the EEPROM contains the starting load address for the packed data. This starting load address is also the address which gains program control after the program load is completed. The program listed in **Example 3-1** initializes the Address Attribute Register, calculates an 8-bit checksum value and programs the 8-bit checksum value in the last location of the 2K EEPROM bank.

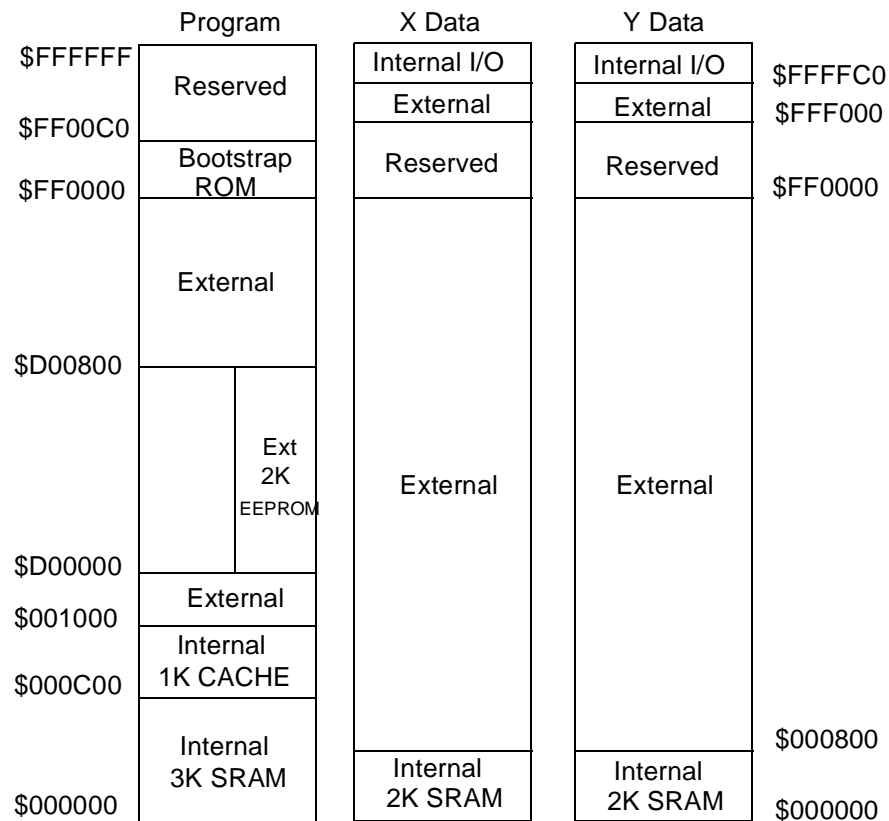


Figure 3-1. 2K x 8-bit BOOT EEPROM Memory Map

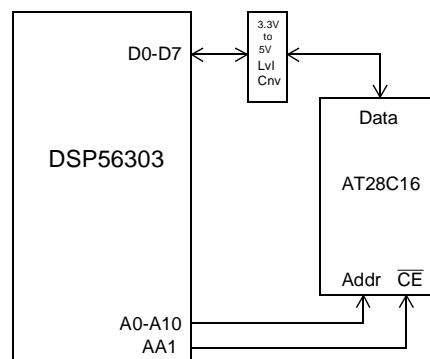


Figure 3-2. 2K x 8-bit BOOT/Overlay EEPROM Memory Example

### 3.1.1 EEPROM Timing Requirements

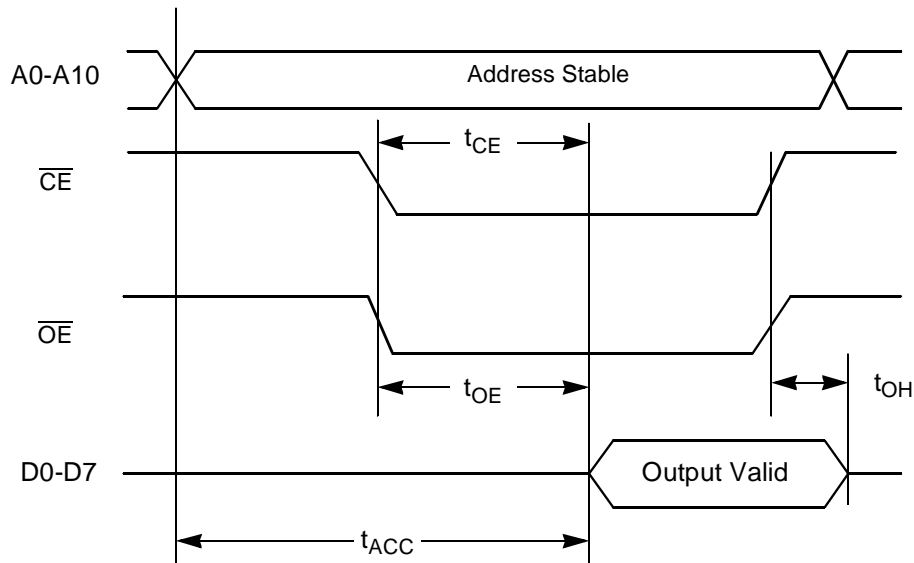
For the EEPROM device to work properly, its timing requirements must be met. Following are the timing requirements for the AT28C16-25 2K x 8-bit 250 nS EEPROM.

#### 3.1.1.1 AT28C16-25 Read Cycle Timing

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

**Table 3-1.** AT28C16-25 Memory Read Timing Specifications

Read Cycle Parameter	Symbol	Min	Max
Address to Output Delay	$t_{ACC}$	—	250nS
Chip Enable to Output Delay	$t_{CE}$	—	250nS
Output Enable to Output Delay	$t_{OE}$	10nS	100nS
Output Hold Time from Address, $\overline{CE}$ or $\overline{OE}$ . Whichever occurs first.	$t_{OH}$	0nS	—



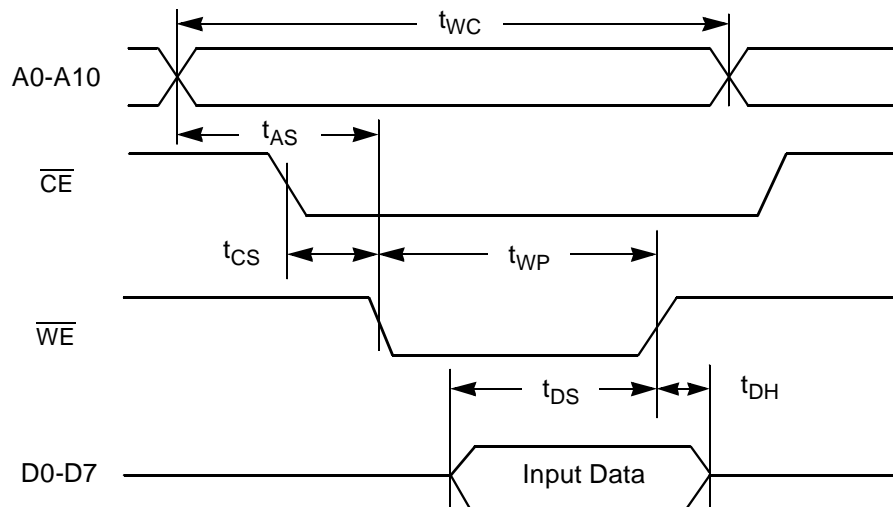
**Figure 3-3.** AT28C16 Memory Read Cycle Timing Diagram

#### 3.1.1.2 AT28C16-25 Write Cycle Timing

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

**Table 3-2.** AT28C16-25 Memory Write Timing Specifications

Write Cycle Parameter	Symbol	Min	Max
Write cycle time (to Program)	$t_{WC}$	—	1mS
Address setup time	$t_{AS}$	10nS	—
$\overline{CE}$ setup time	$t_{CS}$	0nS	—
Write pulse width	$t_{WP}$	100nS	1uS
Data setup time	$t_{DS}$	50nS	—
Data hold time	$t_{DH}$	10nS	—

**Figure 3-4.** AT8C16 Memory Write Cycle Timing Diagram.

Non-volatile memory locations in the EEPROM can be individually programmed by writing data to that location and then waiting 1mS for the operation to complete. To determine when the data has been successfully written, the memory location is read and compared with the last written value. When they compare, the write is complete.



Writing to an EEPROM memory location can be done as follows:

- Write the DATA byte to the EEPROM.
- Read the data from the address to which the DATA was written, and wait until the DATA written = DATA read.

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

For optimal use of the 2K x 8-bit BOOT EEPROM 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 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 1 is used to enable, via EEPROM  $\overline{CE}$ , external 2K EEPROM memory bank accesses in the address range from \$D00000 to \$D007FF during program space requests. Configure the memory address space requirements for the Address Attribute Pin 1 using the Address Attribute Register 1 (AAR1). The AAR1 register 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 8 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 = \$D.
- Specify the most significant portion of the address to compare, Bits 12-23 = \$D00.

The value loaded into the AAR1 register is \$D00D09. The value loaded into AAR0, AAR2 and AAR3 is \$000000.

Select the proper number of wait states for the memory configuration using the bus control register (BCR). The BCR register 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 = \$14
- Address Attribute Area 2 wait states, Bits 10-12 = \$0
- Address Attribute Area 3 wait states, Bits 13-15 = \$0
- 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 \$000280

Configure the operating mode and external memory controls using the Operating Mode Register (OMR). The OMR register 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 register 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 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 SR bits are selected for their defaults of \$000000.

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



**Example 3-1. 2K x 8-bit BOOT EEPROM Checksum Verify Program**

Motorola DSP56300 Assembler Version 6.0.1.6 97-02-27 21:53:54 eeprom1.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,
;
;   eeprom1.asm - Simple program to calculate the 8-bit Checksum for
;   a 2K x 8-bit block of EEPROM memory using a DSP56303.
;   Contains:   Initialization routine,
;               Routine to calculate 8-bit checksum,
;               Routine to write checksum.
;
;   The program runs in Internal P:RAM to calculate the checksum on
;   External P:EEPROM from $D00000 - $D003FF @ 20w/s
;
D00000 MemStart      equ    $D00000
D00400 MemEnd       equ    $D00400
000400 MemSize      equ    MemEnd-MemStart ; Last Word is stored Checksum value

;--- Program Specific Storage Locations (X DATA SPACE)
NEW_CHECKSUM
000000 OLD_CHECKSUM  equ    $000000      ; Computed Checksum Value
000001 OLD_CHECKSUM  equ    $000001      ; Old Checksum from EEPROM

;--- DSP56303 Control Registers (X I/O SPACE)
FFFFFB BCR          equ    $FFFFFB      ; Bus Control Register
FFFFFD PCTL         equ    $FFFFFD      ; PLL Control Register
FFFFF8 AAR1         equ    $FFFFF8      ; Address Attribute Register 1

;--- 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

;--- AAR1 value = 0xD00D09
000001 acctypel     equ    1             ; External Memory access type = 0x1
000000 aahighl      equ    0             ; Enable AAL pin to be low when selected
000008 aapl         equ    $8            ; Yes, Enable AAL pin on ext 'P' accesses
000000 aaxl         equ    0             ; No, Enable AAL pin on ext 'X' accesses
000000 aayl         equ    0             ; No, Enable AAL pin on ext 'Y' accesses
000000 aswapl       equ    0             ; No, Enable address bus swap
000000 enpackl      equ    0             ; No, Enable packing/unpacking logic
000D00 naddl        equ    $000D00      ; Compare 13 address bits
D00000 msaddl       equ    $D00000      ; Most significant portion of address,
; $D00000 - D007ff, to compare.
; (1101,0000,0000,0xxx,xxxx,xxxx)
D00D09 AAR1_value   equ    acctypel+aahighl+aapl+aaxl+aayl+aswapl+enpackl+naddl+msaddl

;--- BCR value = 0x000280
000000 aaa0ws       equ    0             ; Address Attribute Area 0 w/s = 0
000280 aaa1ws       equ    $280         ; Address Attribute Area 1 w/s = 20
000000 aaa2ws       equ    0             ; Address Attribute Area 2 w/s = 0
000000 aaa3ws       equ    0             ; Address Attribute Area 3 w/s = 0
000000 defws        equ    0             ; Default Address Area w/s = 0
000000 busss        equ    0             ; Bus state status = 0
000000 enblh        equ    0             ; Enable Bus Lock Hold = 0
000000 enbrh        equ    0             ; Enable Bus Request Hold = 0
000280 BCR_value    equ    aaa0ws+aaa1ws+aaa2ws+aaa3ws+defws+busss+enblh+enbrh

;-----
;   Header for DSP56303 BOOT Code
;-----
P:0000FE org         p:$100-2
P:0000FE dc         pgm_end-eeprom1 ; Number of words in program
P:0000FF dc         eeprom1         ; Starting address for program
;-----

```

```

75     P:000100                org     p:$100          ; Keep the program in internal RAM
76
77         eeprom1
78     P:000100  0D1080        bsr     init          ; Initialize DSP
              00000D
79
80     P:000102  0D1080        bsr     calc_checksum ; Calculate Checksum of EEPROM
              000014
81
82     P:000104  548100        move    x:OLD_CHECKSUM,a1 ; Get EEPROM's Old Checksum value
83     P:000105  558000        move    x:NEW_CHECKSUM,b1 ; Get Calculated Checksum value
84     P:000106  20000D        cmp     a,b          ; Old Checksum = New Checksum?
85     P:000107  0D104A        beq     _done        ; Yes, we are done
              000005
86
87     P:000109  448000        move    x:NEW_CHECKSUM,x0 ; Get Calculated Checksum value
88     P:00010A  0D1080        bsr     write_checksum ; Write EEPROM Checksum Data value
              000023
89
90         _done
91     P:00010C  050C00        bra     *            ; DONE, Do a dynamic HALT
92
93
94     ;-----
95     ; Initialization Section
96     ;-----
97         init
98     P:00010D  08F4BD        movep   #PCTL_value,x:PCTL ; Set PLL Control Register
              0E0013
99     P:00010F  05F439        movec   #$080000,SR      ; Enable 1K Cache
              080000
100    P:000111  08F4BB        movep   #BCR_value,x:BCR ; Set external wait states
              000280
101    P:000113  08F4B8        movep   #AAR1_value,x:AAR1 ; Set Address Attribute Reg1
              D00D09
102
103    P:000115  00000C        rts
104
105     ;-----
106     ; Routine to Calculate 8-bit Checksum
107     ;-----
108         calc_checksum
109    P:000116  05F420        move    #-1,m0        ; Set LINEAR addressing mode
              FFFFFFFF
110    P:000118  60F400        move    #MemStart,r0 ; Set Starting Address of EEPROM
              D00000
111    P:00011A  70F400        move    #MemSize-1,n0 ; Set to Size of Flash - Checksum
              0003FF
112
113    P:00011C  200013        clr     a
114    P:00011D  20001B        clr     b
115    P:00011E  540000        move    a1,x:NEW_CHECKSUM ; Initialize computed checksum -> $000000
116    P:00011F  540100        move    a1,x:OLD_CHECKSUM ; Initialize read checksum -> $000000
117
118     ; Compute the 8-bit Checksum
119    P:000120  44F400        move    #>$FF,x0      ; Set lower Byte Mask
              0000FF
120
121    P:000122  06D810        dor    n0,_ploop
              000004
122    P:000124  07D88C        move    p:(r0)+,a1    ; Get the EEPROM location Value
123    P:000125  200046        and    x0,a           ; Mask for lower byte
124    P:000126  200018        add    a,b           ; Compute checksum
              _ploop
125
126
127    P:000127  07E08C        move    p:(r0),a1     ; Get EEPROM's Old Checksum value
128    P:000128  20004E        and    x0,b           ; Limit calculated Checksum to lower byte
129    P:000129  200046        and    x0,a           ; Limit Old Checksum to lower byte
130    P:00012A  550000        move    b1,x:NEW_CHECKSUM ; Save the Computed Checksum value
131    P:00012B  540100        move    a1,x:OLD_CHECKSUM ; Save the Old Checksum value
132
133    P:00012C  00000C        rts
134
135     ;-----
136     ; Routine to Place EEPROM into ERASE/WRITE MODE and send it the checksum Data
137     ;-----
138         write_checksum
139    P:00012D  60F400        move    #MemEnd-1,r0
              D003FF
140    P:00012F  076084        move    x0,p:(r0)    ; Send the checksum value to the EEPROM

```

## EEPROM Memory

---

```
141                                     ; -- Now in EEPROM's Data Protect State
142                                     ; -- Wait till ERASE/WRITE Cycle is complete
143 P:000130 44F400      move    #>$FF,x0    ; Lower Byte Mask
      0000FF
144
145         _write_wait
146 P:000132 07E08D      move    p:(r0),b1    ; Get current value at EEPROM location
147 P:000133 20004E      and     x0,b          ; Mask for lower byte
148 P:000134 20000D      cmp     a,b          ; Value written = Value in EEPROM?
149 P:000135 0527DD      bne    _write_wait    ; No, wait until it is
150                                     ; Yes
151 P:000136 00000C      rts
152
153                                     ;-----
154         pgm_end
155
156                                     end    eeprom1

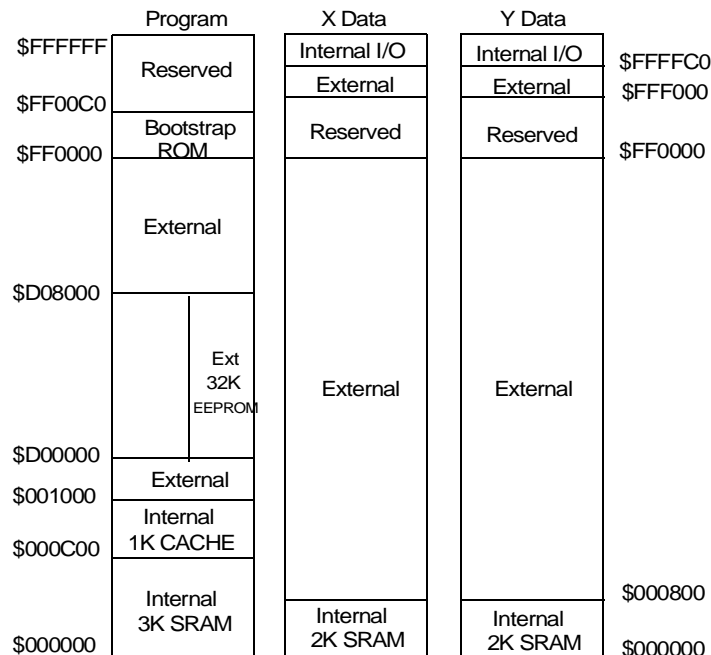
0      Errors
0      Warning
```

## 3.2 32K x 8-bit BOOT/Overlay EEPROM Example

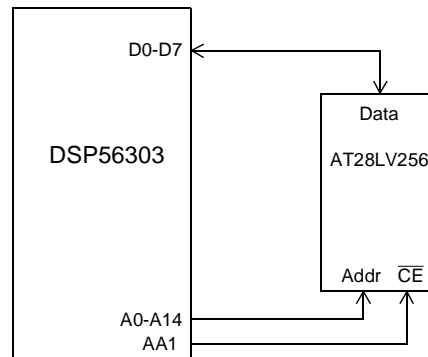
This section describes a 32K x 8-bit 'P' memory space, Boot-strap, BOOT, with program overlay EEPROM implementation using ATMEL's AT28LV256-25 device (see **Figure 3-6** for the memory map layout, **Figure 3-7** for the block diagram, **Figure 3-10** for the schematic, and **Example 3-2** for the code example). Using an EEPROM in this configuration, a program placed in the EEPROM can be loaded into the DSP at Boot time and run. The program can then load additional programs or data from the EEPROM into the DSP via overlay techniques. The boot or overlay program can then save an updated version of the program or data back into the EEPROM for later use.

For this example, the DSP core runs at 80 MHz and the input frequency source is a 4.000 MHz crystal. For a 250 nS EEPROM, 20 wait states are required. However, during the DSP's Boot sequence, the DSP generates 31 wait states with the core running at 4.0 MHz, sufficient to accommodate a 7.7 mS access time device. This 3.3 V device is organized as 32K x 8-bits with a 250 nS access time. One memory device composes the 8-bit wide BOOT bus.

During RESET with Mode 1 selected, the DSP BOOT code configures Address Attribute Line 1 for program accesses in the address range from \$D00000 to \$DFFFFFF. The BOOT code then loads bytes from the EEPROM, packs them into 24-bit words and stores them into Program RAM. The first word, three packed bytes read from the EEPROM, indicates the number of words to load. The second word from the EEPROM contains the starting load address for the packed data. This starting load address is also the address which gains program control after the program load is completed. The program listed in **Example 3-2** initializes the Address Attribute Register, calculates an 8-bit checksum value, and programs the 8-bit checksum value in the last location of the 32K EEPROM bank.



**Figure 3-6.** 32K x 8-bit BOOT EEPROM Memory Map



**Figure 3-7.** 32K x 8-bit BOOT/Overlay EEPROM Memory Example.

### 3.2.1 EEPROM Timing Requirements

For the EEPROM device to work properly, its timing requirements must be met. Following are the timing requirements for the AT28LV256-25 32K x 8-bit 250 nS EEPROM.

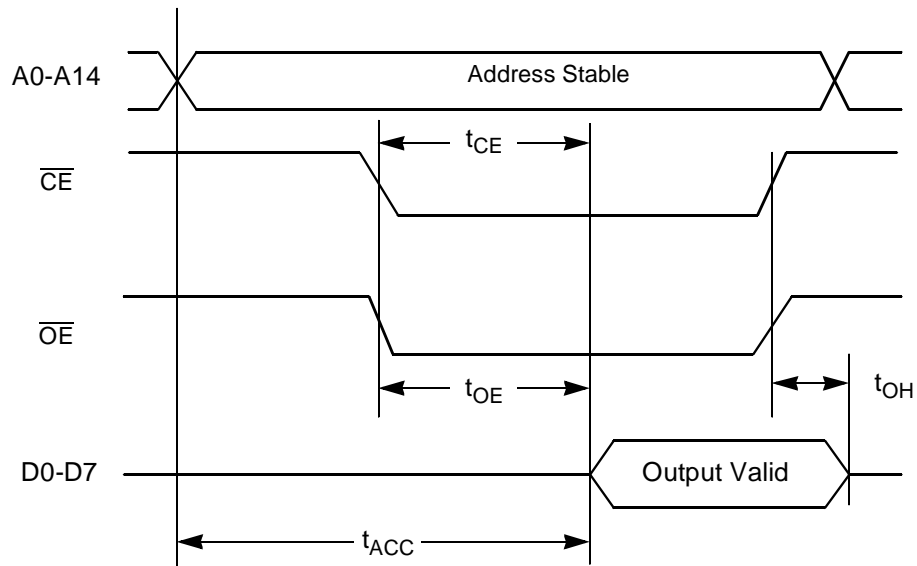
#### 3.2.1.1 AT28LV256-25 Read Cycle Timing

**Table 3-3** shows the memory read timing specification values used in the memory read cycle timing diagram, **Figure 3-8**.

**Table 3-3.** AT28LV256-25 Memory Read Timing Specifications

Read Cycle Parameter	Symbol	Min	Max
Address to output delay	$t_{ACC}$	—	250nS
Chip enable to output delay	$t_{CE}$	—	250nS
Output enable to output delay	$t_{OE}$	—	100nS
Output Hold time from address, $\overline{CE}$ or $\overline{OE}$ . Whichever occurs first.	$t_{OH}$	0nS	—





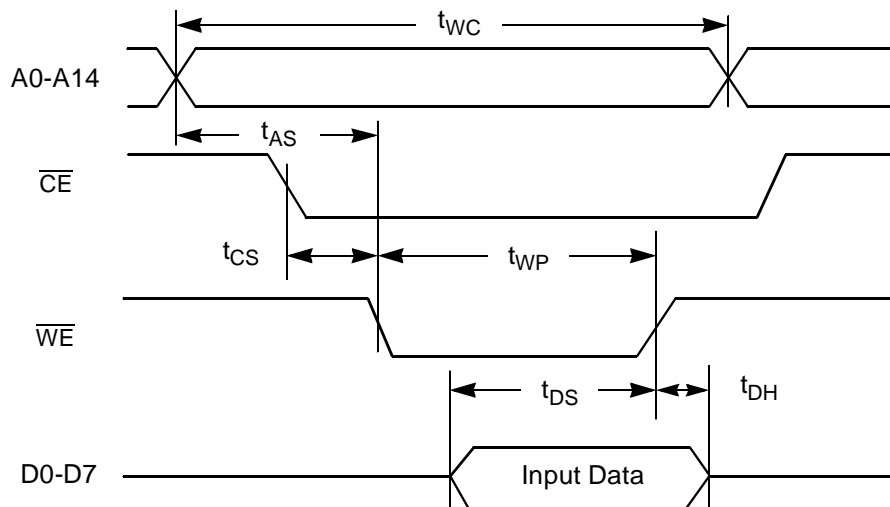
**Figure 3-8.** AT28LV256 Memory Read Cycle Timing Diagram.

### 3.2.2 AT28LV256-25 Write Cycle Timing

**Table 3-4** shows the memory write timing specification values used in the memory write cycle timing diagram, **Figure 3-9**.

**Table 3-4.** AT28LV256-25 Memory Write Timing Specifications

Write Cycle Parameter	Symbol	Min	Max
Write cycle time (to program)	$t_{WC}$	—	10mS
Address setup time	$t_{AS}$	0 nS	—
$\overline{CE}$ setup time	$t_{CS}$	0 nS	—
Write pulse width	$t_{WP}$	200 nS	—
Data setup time	$t_{DS}$	50 nS	—
Data hold time	$t_{DH}$	0 nS	—



**Figure 3-9.** AT28LV256 Memory Write Cycle Timing Diagram.

Non-volatile memory locations in the EEPROM can be individually programmed, or a sector of 64 locations can be written during one program cycle. The AT28LV256 device is organized as 128 sectors of 64 bytes each.

To write to an EEPROM memory location requires the writing of one byte, then waiting for the erase/program cycle to complete; or the data for a complete sector, 64 bytes, is written to the EEPROM before the erase/program cycle is started. To do this, the following data sequence is written to the EEPROM:

- Write \$0000AA to location \$5555 relative to the EEPROM,
- Write \$000055 to location \$2AAA relative to the EEPROM,
- Write \$0000A0 to location \$5555 relative to the EEPROM,
- Write 1 to 64 bytes of DATA to the sector in the EEPROM,
- Read the last sector Address DATA until the DATA written = DATA read.

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

For optimal use of the 32K x 8-bit BOOT EEPROM 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 1 enables, via EEPROM  $\overline{CE}$ , external 32K EEPROM memory bank accesses in the address range from \$D00000 to \$D07FFF during program space requests. Configure the memory address space requirements for the Address Attribute Pin 1 using the Address Attribute Register 1 (AAR1). The AAR1 register 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 = \$D00.

The value loaded into the AAR1 is \$D00909. The value loaded into AAR0, AAR2 and AAR3 is \$000000.

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

- Address Attribute Area Zero wait states, Bits 0-4 = \$0
- Address Attribute Area One wait states, Bits 5-9 = \$14
- Address Attribute Area Two wait states, Bits 10-12 = \$0
- Address Attribute Area Three wait states, Bits 13-15 = \$0
- 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 \$000280.

Configure the operating mode and external memory controls using the Operating Mode Register (OMR). The OMR register 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 register 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 SR bits are selected for their defaults of \$000000.

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

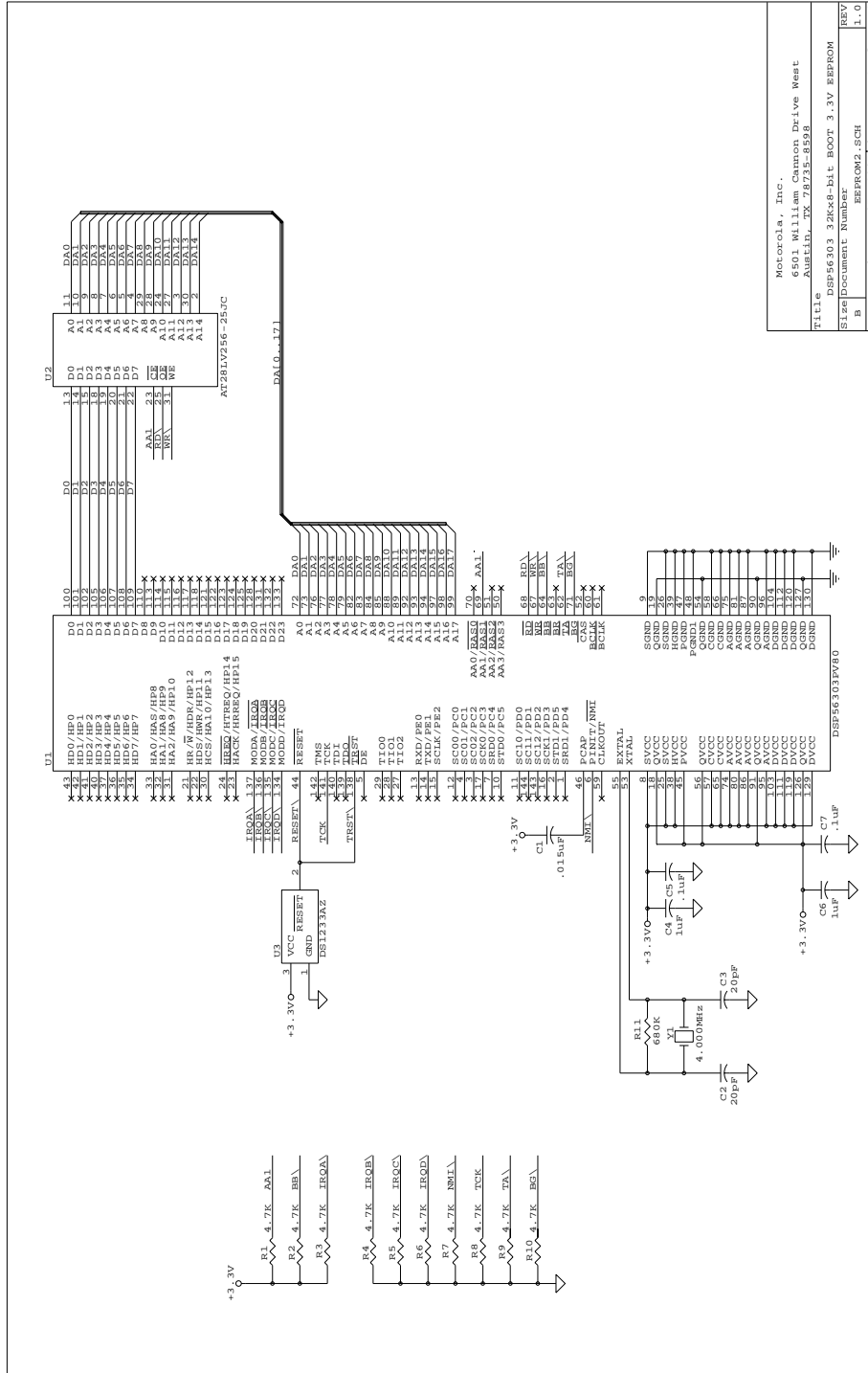


Figure 3-10. 32Kx8-bit BOOT EEPROM Schematic

Motorola, Inc.
6501 Williams Canyon Drive West
Chandler, AZ 85226-1898
Title
DSP56303 32Kx8-bit BOOT 3.3V EEPROM
Size/Document Number
B EEPROM2.SCH
REV
1.0
Date: February 27, 1997/Street
1.GF

**Example 3-2. 32K x 8-bit BOOT EEPROM Checksum Verify Program**

Motorola DSP56300 Assembler Version 6.0.1.6 97-02-26 21:21:55 eeprom2.asm

```

1          page      132,60,3,3,
2          ;
3          ; eeprom2.asm - Simple program to calculate the 8-bit Checksum for
4          ; a 32K x 8-bit block of EEPROM memory using a DSP56303.
5          ; Contains: Initialization routine,
6          ; Routine to calculate 8-bit checksum,
7          ; Routine to write checksum.
8          ;
9          ; The program runs in Internal P:RAM to calculate the checksum on
10         ; External P:EEPROM from $D00000 - $D07FFF @ 20w/s
11         ;
12
13         D00000      MemStart      equ      $D00000
14         D08000      MemEnd        equ      $D08000
15         008000      MemSize       equ      MemEnd-MemStart ; Last Word is stored Checksum value
16
17         ;--- Program Specific Storage Locations (X DATA SPACE)
18         NEW_CHECKSUM
19         000000      OLD_CHECKSUM   equ      $000000          ; Computed Checksum Value
20
21         000001      OLD_CHECKSUM   equ      $000001          ; Old Checksum from PEROM
22
23         ;--- DSP56303 Control Registers (X I/O SPACE)
24         FFFFFB      BCR            equ      $FFFFFFB         ; Bus Control Register
25         FFFFFFFD     PCTL          equ      $FFFFFFD         ; PLL Control Register
26         FFFFF8      AAR1          equ      $FFFFFF8         ; Address Attribute Register 1
27
28         ;--- PCTL value = 0x0E0013
29         000000      prediv         equ      0                ; Pre-Divider = 1
30         000000      lowdiv        equ      0                ; Low Power Divider = 1
31         000013      pllmul         equ      19              ; VCO Mult = 20; (19+1)*4.00MHz=80.00MHz
32         000000      crystal        equ      0                ; No, Crystal not less than 200kHz
33         000000      disXTAL        equ      0                ; No, do not disable crystal use
34         020000      pllstop        equ      $020000         ; Yes, PLL runs during STOP
35         040000      enpll          equ      $040000         ; Yes, enable PLL operation
36         080000      disclk         equ      $080000         ; Yes, disable CORE clock output
37         0E0013      PCTL_value     equ      rediv+lowdiv+pllmul+crystal+disXTAL+pllstop+enpll+disclk
38
39         ;--- AAR1 value = 0xD00909
40         000001      acctypel       equ      1                ; External Memory access type = 0x1
41         000000      aahighl        equ      0                ; Enable AAL pin to be low when selected
42         000008      aapl           equ      $8                ; Yes, Enable AAL pin on ext 'P' accesses
43         000000      aaxl           equ      0                ; No, Enable AAL pin on ext 'X' accesses
44         000000      aayl           equ      0                ; No, Enable AAL pin on ext 'Y' accesses
45         000000      aswapl         equ      0                ; No, Enable address bus swap
46         000000      enpackl        equ      0                ; No, Enable packing/unpacking logic
47         000900      naddl          equ      $000900         ; Compare 9 address bits
48         D00000      msaddl         equ      $D00000         ; Most significant portion of address,
49         ; $D00000 - D07fff, to compare.
50         ; (1101,0000,0xxx,xxxx,xxxx,xxxx)
51         D00909      AAR1_value     equ      acctypel+aahighl+aapl+aaxl+aayl+aswapl+enpackl+naddl+msaddl
52
53
54         ;--- BCR value = 0x000280
55         000000      aaa0ws         equ      0                ; Address Attribute Area 0 w/s = 0
56         000280      aaa1ws         equ      $280            ; Address Attribute Area 1 w/s = 20
57         000000      aaa2ws         equ      0                ; Address Attribute Area 2 w/s = 0
58         000000      aaa3ws         equ      0                ; Address Attribute Area 3 w/s = 0
59         000000      defws          equ      0                ; Default Address Area w/s = 0
60         000000      busss          equ      0                ; Bus state status = 0
61         000000      enblh          equ      0                ; Enable Bus Lock Hold = 0
62         000000      enbrh          equ      0                ; Enable Bus Request Hold = 0
63         000280      BCR_value      equ      aaa0ws+aaa1ws+aaa2ws+aaa3ws+defws+busss+enblh+enbrh
64
65         ;-----
66         ; Header for DSP56303 BOOT Code
67         ;-----
68         P:0000FE      org          p:$100-2
69
70         P:0000FE      dc            pgm_end-eeprom2          ; Number of words in program
71         P:0000FF      dc            eeprom2                  ; Starting address for program
72
73
74         ;-----

```

```

75     P:000100                org     p:$100           ; Keep the program in internal RAM
76
77         eeprom2
78     P:000100  0D1080        bsr     init           ; Initialize DSP
              00000D
79
80     P:000102  0D1080        bsr     calc_checksum  ; Calculate Checksum of EEPROM
              000014
81
82     P:000104  548100        move    x:OLD_CHECKSUM,a1 ; Get EEPROM's Old Checksum value
83     P:000105  558000        move    x:NEW_CHECKSUM,b1 ; Get Calculated Checksum value
84     P:000106  20000D        cmp     a,b           ; Old Checksum = New Checksum?
85     P:000107  0D104A        beq     _done         ; Yes, we are done
              000005
86
87     P:000109  448000        move    x:NEW_CHECKSUM,x0 ; Get Calculated Checksum value
88     P:00010A  0D1080        bsr     write_checksum  ; Write EEPROM Checksum Data value
              000023
89
90         _done
91     P:00010C  050C00        bra     *             ; DONE, Do a dynamic HALT
92
93
94     ;-----
95     ; Initialization Section
96     ;-----
97         init
98     P:00010D  08F4BD        movep   #PCTL_value,x:PCTL ; Set PLL Control Register
              0E0013
99     P:00010F  05F439        movec   #$080000,SR      ; Enable 1K Cache
              080000
100    P:000111  08F4BB        movep   #BCR_value,x:BCR ; Set external wait states
              000280
101    P:000113  08F4B8        movep   #AAR1_value,x:AAR1 ; Set Address Attribute Reg1
              D00909
102
103    P:000115  00000C        rts
104
105     ;-----
106     ; Routine to Calculate 8-bit Checksum
107     ;-----
108         calc_checksum
109    P:000116  05F420        move    #-1,m0         ; Set LINEAR addressing mode
              FFFFFFFF
110    P:000118  60F400        move    #MemStart,r0   ; Set Starting Address of EEPROM
              D00000
111    P:00011A  70F400        move    #MemSize-1,n0  ; Set to Size of Flash - Checksum
              007FFF
112
113    P:00011C  200013        clr     a
114    P:00011D  20001B        clr     b
115    P:00011E  540000        move    a1,x:NEW_CHECKSUM ; Initialize computed checksum -> $000000
116    P:00011F  540100        move    a1,x:OLD_CHECKSUM ; Initialize read checksum -> $000000
117
118     ; Compute the 8-bit Checksum
119    P:000120  44F400        move    #>$FF,x0       ; Set lower Byte Mask
              0000FF
120
121    P:000122  06D810        dor     n0,_ploop
              000004
122    P:000124  07D88C        move    p:(r0)+,a1     ; Get the EEPROM location Value
123    P:000125  200046        and     x0,a           ; Mask for lower byte
124    P:000126  200018        add     a,b           ; Compute checksum
              _ploop
125
126
127    P:000127  07E08C        move    p:(r0),a1     ; Get EEPROM's Old Checksum value
128    P:000128  20004E        and     x0,b           ; Limit calculated Checksum to lower byte
129    P:000129  200046        and     x0,a           ; Limit Old Checksum to lower byte
130    P:00012A  550000        move    b1,x:NEW_CHECKSUM ; Save the Computed Checksum value
131    P:00012B  540100        move    a1,x:OLD_CHECKSUM ; Save the Old Checksum value
132
133    P:00012C  00000C        rts
134
135     ;-----
136     ; Routine to Place EEPROM into ERASE/WRITE MODE and send it the checksum data
137     ;-----
138         write_checksum
139    P:00012D  60F400        move    #MemEnd-1,r0
              D07FFF
140    P:00012F  076084        move    x0,p:(r0)     ; Send the checksum value to the EEPROM

```

## EEPROM Memory

---

```
141                                     ; -- Now in EEPROM's Data Protect State
142                                     ;-- Wait till ERASE/WRITE Cycle is complete
143 P:000130 44F400      move    #>$FF,x0      ; Lower Byte Mask
144                                     0000FF
145                                     _write_wait
146 P:000132 07E08D      move    p:(r0),b1      ; Get current value at EEPROM location
147 P:000133 20004E      and     x0,b          ; Mask for lower byte
148 P:000134 20000D      cmp     a,b          ; Value written = Value in EEPROM?
149 P:000135 0527DD      bne    _write_wait    ; No, wait until it is
150                                     ; Yes
151 P:000136 00000C      rts
152
153                                     ;-----
154                                     pgm_end
155
156                                     end     eeprom2

0      Errors
0      Warning
```



## 4 Erasable Programmable Read Only Memory

EPROM, PROM and ROM memory provide non-volatile program and data storage with relatively fast access times. However, even the fastest EPROM memories require a DSP core running at 80 MHz to generate external memory wait states. With each wait state equivalent to one clock period of the DSP core, one wait state for a core running at 80 MHz is roughly 12.5 nS.

When the external memory device must reside in a stable address during the entire external access, a DSP56300 family device incurs an automatic one wait state penalty. Since EPROM devices have this address stability requirement, the DSP operates with at least one wait state when using these external memories.

The following three EPROM design examples illustrate how easy and flexible it is to use EPROM devices with the DSP56300 family. The first example, 128K x 24-bit BOOT, 128K x 24-bit 'P', 128K x 24-bit 'X' and 128K x 24-bit 'Y', shows a solution for an embedded system using four unique banks of EPROM memory. This allows for one bank to be used for booting the DSP, another bank for external program code execution, and the remaining two banks for external coefficient data use. The second example, 16K x 8-bit BOOT, shows a solution for a Bootable embedded system. In this implementation, the DSP loads internal program RAM from the EPROM at BOOT time and then executes it. The third example, 512K x 8-bit BOOT/Overlay, shows another solution for a Bootable embedded system. This implementation loads internal DSP program RAM from the EPROM at BOOT time and then overlays new programs and data from the EPROM when needed.

### 4.1 128K x 24-bit BOOT, 'P', 'X' and 'Y' EPROM Example

This section describes a 128K x 24-bit BOOT, 128K x 24-bit 'P', 128K x 24-bit 'X' and 128K x 24-bit 'Y' memory space, EPROM implementation using AMD's AM27C040 device (see **Figure 4-1** for BOOT memory map layout, **Figure 4-2** for user memory map layout, and **Figure 4-3** for the block diagram, and **Figure 4-1** for the address attribute associations for BOOT, Program, X-Data and Y-Data spaces). Since the EPROM is present in the DSP's program space, program control can be turned over to the program in the EPROM at Boot time. This allows an embedded application to boot and run from EPROM, permitting the DSP's internal 1K Cache to help speed repeated program functions. Also, since 'X' and 'Y' data space is available in the EPROM, program and data variables can be referenced after bootup.

For this example, the DSP core runs at 80 MHz and the input frequency source is a 4.000 MHz crystal. For a 90 nS EPROM, eight wait states are required. This 5 V device is organized as 512K x 8-bits with a 90 nS access time. Three memory devices compose the 24-bit wide 'P', 'X' and 'Y' buses.

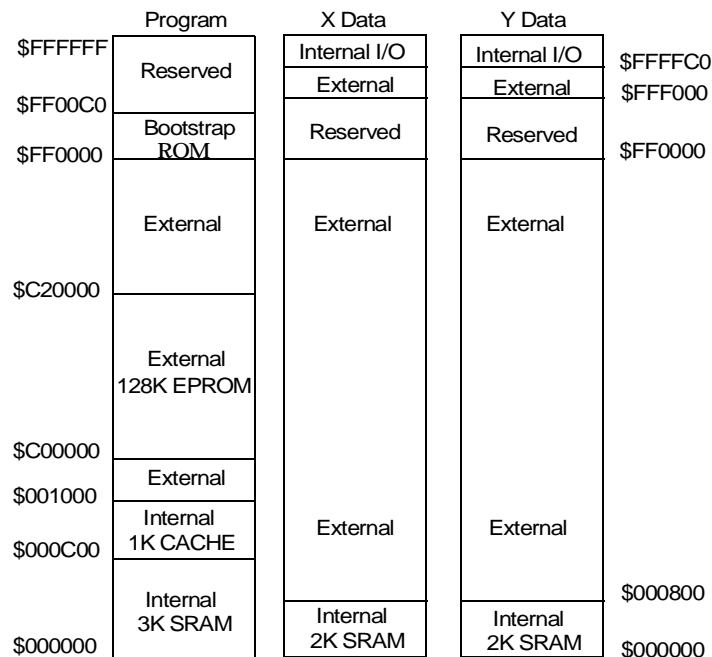
Level conversion to and from 3.3 V and 5 V is necessary on the 24-bit data bus to accommodate the 5 V memory devices. This is accomplished by using three Quality Semiconductor's QS3245 QuickSwitch<sup>®</sup> 8-bit Bus Switches. These switches allow the connection of 3.3 V CMOS logic (the DSP data bus) on one side and 5 V TTL-compatible logic (the 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.

During DSP RESET with Mode 0 selected, the DSP starts fetching instructions from external memory location P:\$C00000. Since the DSP56303 device only uses 18 address lines, A0 - A17, to select external memory, address P:\$C00000 appears as P:\$000000 on the external address bus. Therefore, the EPROM is configured to respond to all external memory requests. Because of this, data aliasing occurs at every 256K boundary if no additional address decoding is provided. Since the 512K EPROM

requires 19 address lines to select all of its 512K memory locations, EPROM address line A18 is controlled by Address Attribute Line 1 (AA1) (see the memory space selection chart in **Figure 4-1**, the schematic in **Figure 4-5** and the program listed in **Example 4-1**). The EEPROM address line A17 which is controlled by Address Attribute Line 0 (AA0), selects the EEPROM used during X-Data or Y-Data accesses. Using these two (2) address attribute lines, the 512K x 24-bit EPROM memory bank can be segmented into four (4) regions of 128K x 24-bits each (see **Figure 4-1**). The program listed in **Example 4-1** initializes the address attribute registers and calculates a 24-bit checksum value for each of the four 128K EPROM memory spaces to compare with the checksum values stored in the last location of each 128K EPROM bank.

**Table 4-1.** Memory Space Selections Selected by Address Attributes

	<b>Address Attribute Line EPROM Address Line</b>	<b>AA1 A18</b>	<b>AA0 A17</b>
<b>Y-Data Space</b>	Y:\$100000 - \$1FFFFFF	0	0
<b>X-Data Space</b>	X:\$100000 - \$1FFFFFF	0	1
<b>Program Space</b>	P:\$100000 - \$1FFFFFF	1	0
<b>Program BOOT Space</b>	P:\$C00000 - \$C1FFFF Aliased from \$0 - \$FFFFFF at every fourth 128K page boundary while AA0 and AA1 are unconfigured.	1	1



**Figure 4-1.** 128K x 24-bit BOOT EPROM Memory Map

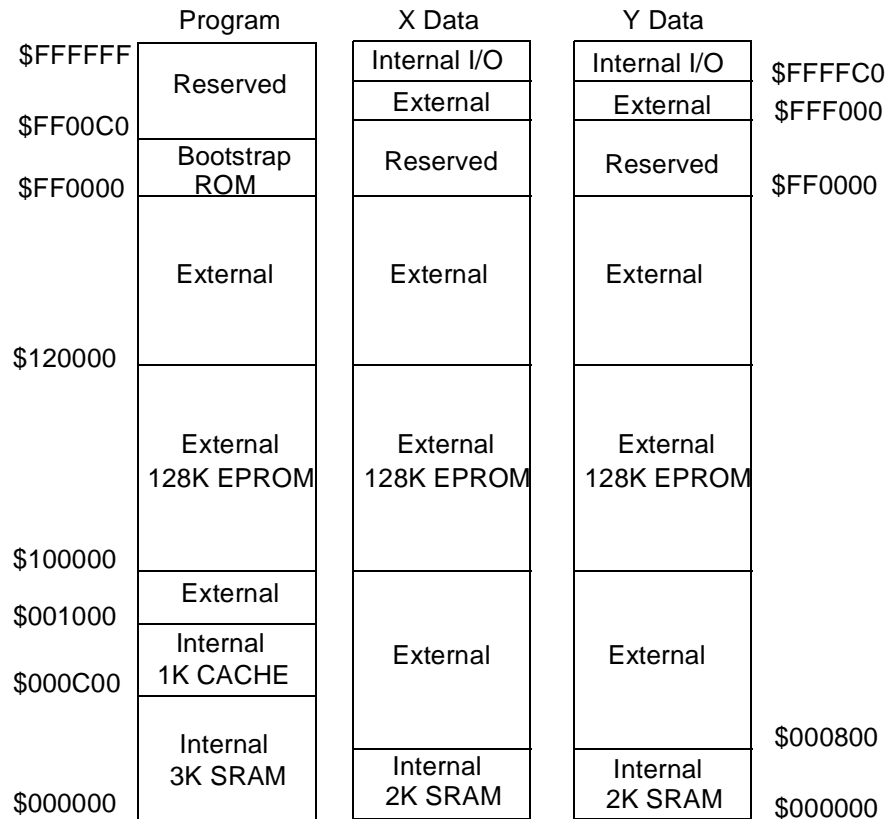


Figure 4-2. 128K x 24-bit 'P', 'X' and 'Y' EPROM Memory Map

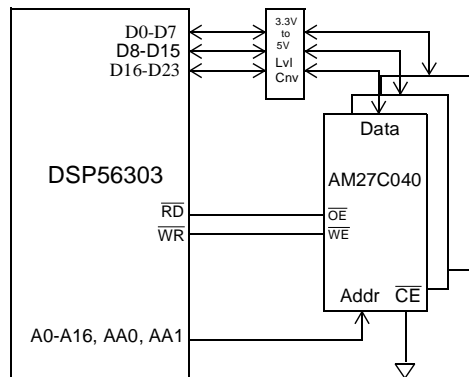


Figure 4-3. 512K x 24-bit EPROM Memory Example.

### 4.1.1 EPROM Timing Requirements

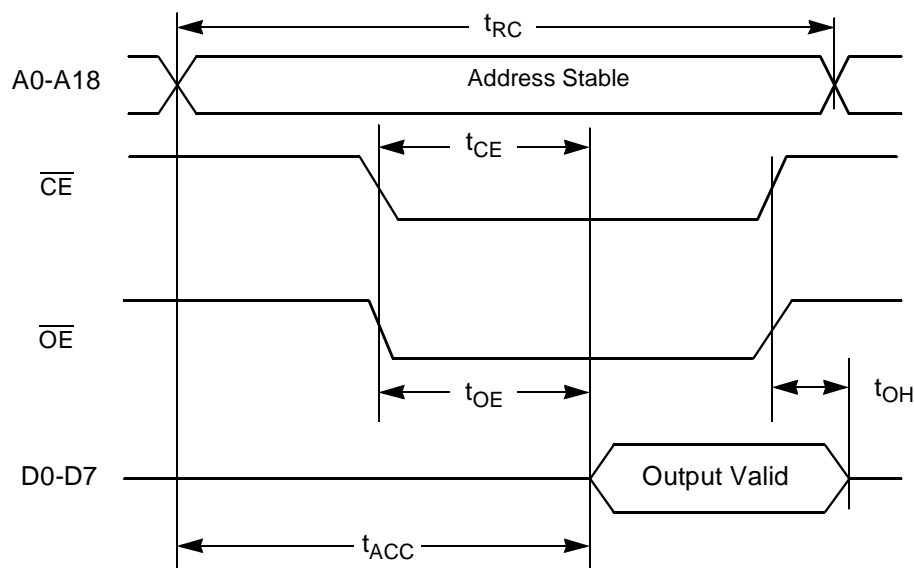
For the EPROM device to work properly, its timing requirements must be met. Following are the timing requirements for the AM27C040-90 512K x 8-bit 90 nS EPROM.

### 4.1.1.1 AM27C040-90 Read Cycle Timing

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

**Table 4-2.** AM27C040-90 Memory Read Timing Specifications

Read Cycle Parameter	Symbol	Min	Max
Read cycle time	$t_{RC}$	90 nS	—
Address to output delay	$t_{ACC}$	—	90 nS
Chip enable to output delay	$t_{CE}$	—	90 nS
Output enable to output delay	$t_{OE}$	—	40 nS
Output hold time from address, $\overline{CE}$ or $\overline{OE}$ . Whichever occurs first.	$t_{OH}$	0 nS	—



**Figure 4-4.** AM27C040 Memory Read Cycle Timing Diagram.

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

For optimal use of the 512K x 24-bit 'P' 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 = 0x0
- Low-power Divider value = 1, Bits 12-14 = 0x0
- VCO Multiplication value = 20, Bits 0-11 = 0x013
- 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 0x0E0013.

Address Attribute Pin 1 selects, via EPROM A18, the external 128K EPROM memory bank during accesses in the address range from \$100000 to \$11FFFF between program space requests and X-Data/Y-Data space requests. Configure the memory address space requirements for the Address Attribute Pin 1 using the Address Attribute Register 1 (AAR1). The AAR1 register value combines the following bits for each feature:

- Specify the external memory access type as asynchronous SRAM, Bits 0-1 = 0x1.
- 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 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 = 0x7.
- Specify the most significant portion of the address to compare, Bits 12-23 = 0x100.

The value loaded into the AAR1 register is 0x10070D.

Address Attribute Pin 0 selects, via EPROM A17, the external 128K EPROM memory bank accesses in the address range from \$100000 to \$11FFFF between X data space requests 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 register value combines the following bits for each feature:

- Specify the external memory access type as asynchronous SRAM, Bits 0-1 = 0x1.
- 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 = 0x7.
- Specify the most significant portion of the address to compare, Bits 12-23 = 0x100.

The value loaded into the AAR0 register is 0x100715. The value loaded into AAR2 and AAR3 is 0x000000.

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

- Address attribute area zero wait states, Bits 0-4 = 0x8
- Address attribute area one wait states, Bits 5-9 = 0x8
- Address attribute area two wait states, Bits 10-12 = 0x0
- Address attribute area three wait states, Bits 13-15 = 0x0
- Default address area wait states, Bits 16-20 = 0x8
- 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 0x080108.

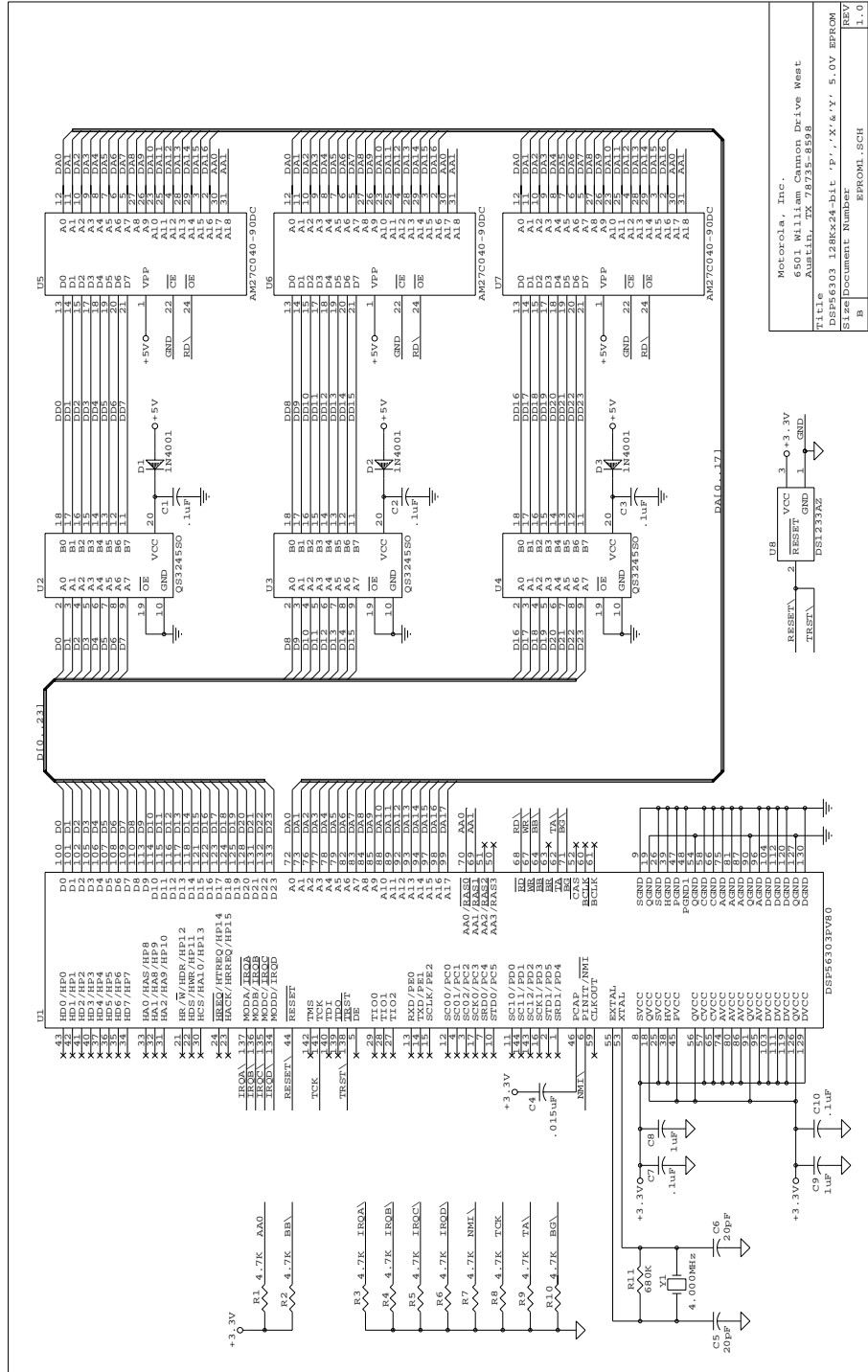


Figure 4-5. 128Kx24 BOOT, 'P', 'X' and 'Y' Space EPROM Schematic

Motorola, Inc.  
6501 William Cannon Drive West  
Austin, TX 78731-9398  
Title: DSP56303 128Kx24-bit 'P', 'X' & 'Y' 5.0V EPROM  
B Size Document Number: EPR0M1.SCH  
REV 1.0  
Date: February 28, 1997 Sheet 1 of 1

### Example 4-1. 128K x 24-bit 'P', 128K x 24-bit 'X' and 128K x 24-bit 'Y' Space EPROM Checksum Program

Motorola DSP56300 Assembler Version 6.0.1.6 97-02-27 21:54:56 eprom1.asm

```

1           page      132,60,3,3,
2           ;
3           ;      eprom1.asm - Simple program to calculate the 24-bit Checksum
4           ;      for a 128K x 24-bit block of BOOT, 128K x 24-bit block
5           ;      of Program, 128K x 24-bit block of X-Data and
6           ;      a 128K x 24-bit block of Y-Data memory using a DSP56303.
7           ;
8           ;
9           ;      The program runs in Internal P:RAM to calculate the checksum;
10          ;      on External BOOT:EPROM from $C00000 - $C1FFFF @ 8w/s,
11          ;      on External P:EPROM from $100000 - $11FFFF @ 8w/s,
12          ;      on External X:EPROM from $100000 - $11FFFF @ 8w/s and
13          ;      on External Y:EPROM from $100000 - $11FFFF @ 8w/s.
14          ;
15
16          C00000      BootStart   equ      $C00000
17          C20000      BootEnd     equ      $C20000
18          020000      BootSize    equ      BootEnd-BootStart ; Last Word is stored Checksum value
19
20          100000      PMemStart   equ      $100000
21          120000      PMemEnd    equ      $120000
22          020000      PMemSize   equ      PMemEnd-PMemStart ; Last Word is stored Checksum value
23
24          100000      XMemStart   equ      $100000
25          120000      XMemEnd    equ      $120000
26          020000      XMemSize   equ      XMemEnd-XMemStart ; Last Word is stored Checksum value
27
28          100000      YMemStart   equ      $100000
29          120000      YMemEnd    equ      $120000
30          020000      YMemSize   equ      YMemEnd-YMemStart ; Last Word is stored Checksum value
31
32          ;--- Program Specific Storage Locations (X DATA SPACE)
33          CKSUM_CALC_P      equ      $000000 ; P Space Computed Checksum Value
34          000000
35          CKSUM_READ_P      equ      $000001 ; P Space Checksum in Last Memory Location
36          000001
37          CKSUM_CALC_X      equ      $000002 ; X Space Computed Checksum Value
38          000002
39          CKSUM_READ_X      equ      $000003 ; X Space Checksum in Last Memory Location
40          000003
41          CKSUM_CALC_Y      equ      $000004 ; Y Space Computed Checksum Value
42          000004
43          CKSUM_READ_Y      equ      $000005 ; Y Space Checksum in Last Memory Location
44          000005
45          CKSUM_CALC_B      equ      $000006 ; BOOT Space Computed Checksum Value
46          000006
47          CKSUM_READ_B      equ      $000007 ; BOOT Space Checksum in Last Memory Location
48          000007
49
50          ;--- DSP56303 Control Registers (X I/O SPACE)
51          FFFFFB      BCR        equ      $FFFFFB ; Bus Control Register
52          FFFFFD      PCTL       equ      $FFFFFD ; PLL Control Register
53          FFFF9       AAR0       equ      $FFF9 ; Address Attribute Register 0
54          FFFF8       AAR1       equ      $FFF8 ; Address Attribute Register 1
55
56          ;--- PCTL value = 0x0E0013
57          000000      prediv     equ      0 ; Pre-Divider = 1
58          000000      lowdiv     equ      0 ; Low Power Divider = 1
59          000013      pllmul     equ      19 ; VCO Mult = 20; (19+1)*4.00 MHz=80.00 MHz
60          000000      crystal    equ      0 ; No, Crystal not less than 200 kHz
61          000000      disXTAL    equ      0 ; No, do not disable crystal use
62          020000      pllstop    equ      $020000 ; Yes, PLL runs during STOP
63          040000      enpll      equ      $040000 ; Yes, enable PLL operation
64          080000      disclk     equ      $080000 ; Yes, disable CORE clock output
65          0E0013      PCTL_value equ      prediv+lowdiv+pllmul+crystal+disXTAL+pllstop+enpll+disclk
66
67          ;--- AAR1 value = 0x10070D
68          000001      acctypel   equ      1 ; External Memory access type = 0x1
69          000001      aahighl     equ      1 ; Enable AAL pin to be high when selected
70          000008      aapl        equ      $8 ; Yes, Enable AAL pin on ext 'P' accesses
71          000000      aaxl        equ      0 ; No, Enable AAL pin on ext 'X' accesses
72          000000      aayl        equ      0 ; No, Enable AAL pin on ext 'Y' accesses
73          000000      aswapl      equ      0 ; No, Enable address bus swap

```



```

74      000000      enpack1      equ      0          ; No, Enable packing/unpacking logic
75      000700      nadd1       equ      $000700    ; Compare 7 address bits
76      100000      msadd1      equ      $100000    ; Most significant portion of address,
77                                     ; $100000 - 11ffff, to compare.
78                                     ; (0001,000x,xxxx,xxxx,xxxx,xxxx)
79      10070A      AAR1_value  equ      acctype1+aahigh1+aapl1+aa1+aay1+aswap1+enpack1+nadd1+msadd1
80
81      ;--- AAR0 value = 0x100715
82      000001      acctype     equ      1          ; External Memory access type = 0x1
83      000004      aahigh      equ      $4         ; Enable AA0 pin to be High when selected
84      000000      aap         equ      0          ; No, Enable AA0 pin on ext 'P' accesses
85      000010      aax         equ      $10        ; Yes, Enable AA0 pin on ext 'X' accesses
86      000000      aay         equ      0          ; No, Enable AA0 pin on ext 'Y' accesses
87      000000      aswap       equ      0          ; No, Enable address bus swap
88      000000      enpack     equ      0          ; No, Enable packing/unpacking logic
89      000700      nadd        equ      $000700    ; Compare 7 address bits
90      100000      msadd       equ      $100000    ; Most significant portion of address,
91                                     ; $100000 - 11ffff, to compare.
92                                     ; (0001,000x,xxxx,xxxx,xxxx,xxxx)
93      100715      AAR0_value  equ      acctype+aahigh+aap+aax+aay+aswap+enpack+nadd+msadd
94
95      ;--- BCR value = 0x080108
96      000008      aaa0ws     equ      $8         ; Address Attribute Area 0 w/s = 8
97      000100      aaalws     equ      $100        ; Address Attribute Area 1 w/s = 8
98      000000      aaa2ws     equ      0          ; Address Attribute Area 2 w/s = 0
99      000000      aaa3ws     equ      0          ; Address Attribute Area 3 w/s = 0
100     080000      defws      equ      $080000    ; Default Address Area w/s = 8
101     000000      busss      equ      0          ; Bus state status = 0
102     000000      enblh     equ      0          ; Enable Bus Lock Hold = 0
103     000000      enbrh     equ      0          ; Enable Bus Request Hold = 0
104     080108      BCR_value  equ      aaa0ws+aaalws+aaa2ws+aaa3ws+defws+busss+enblh+enbrh
105
106      ;-----
107      P:000100      org      p:$100        ; Keep the program in internal RAM
108
109      eprom1
110
111      ;----- Initialization Section -----
112      P:000100      08F4BD      movep     #PCTL_value,x:PCTL ; Set PLL Control Register
113      P:000102      05F43A      movec     #$004000,OMR      ; Disable Address Attribute Priority
114      P:000104      05F439      movec     #$080000,SR      ; Enable 1K Cache
115      P:000106      08F4BB      movep     #BCR_value,x:BCR ; Set external wait states
116      080108
117      ;-----
118      ;-----
119      do_b_checksum
120      P:000108      08F4B9      movep     #>$0,x:AAR0      ; Clear Address Attribute Reg0
121      P:00010A      08F4B8      movep     #>$0,x:AAR1      ; Clear Address Attribute Reg1
122      000000
123      P:00010C      05F420      move     #-1,m0           ; Set LINEAR addressing mode
124      P:00010E      60F400      move     #BootStart,r0    ; Set Starting Address of BOOT:EPROM
125      P:000110      70F400      move     #BootSize,n0     ; Set to Size of BOOT:EPROM
126      020000
127      P:000112      200013      clr      a
128      P:000113      20001B      clr      b
129      P:000114      560600      move     a,x:CKSUM_CALC_B ; Initialize computed checksum -> $000000
130      P:000115      560700      move     a,x:CKSUM_READ_B ; Initialize read checksum -> $000000
131
132      ;----- Compute the 24-bit BOOT: Space Checksum -----
133      P:000116      06D810      dor      n0,b_loop        ;
134      P:000118      07D88E      move     p:(r0)+,a        ; Get the BOOT:EPROM location Value
135      P:000119      200018      add      a,b              ; Compute checksum
136      b_loop
137
138      P:00011A      07E08E      move     p:(r0),a         ; Get Checksum from BOOT:EPROM
139      P:00011B      570600      move     b,x:CKSUM_CALC_B ; Save the Computed Checksum value
140      P:00011C      560700      move     a,x:CKSUM_READ_B ; Save the Old Checksum value
141
142      ;-----
143      ;-----
144      do_p_checksum

```

## Erasable Programmable Read Only Memory

```

145 P:00011D 08F4B9 100715  movep  #AAR0_value,x:AAR0 ; Set Address Attribute Reg0
146 P:00011F 08F4B8 10070A  movep  #AAR1_value,x:AAR1 ; Set Address Attribute Reg1
147
148 P:000121 05F420  FFFFFFFF  move   #-1,m0 ; Set LINEAR addressing mode
149 P:000123 60F400  100000  move   #PMemStart,r0 ; Set Starting Address of P:EPROM
150 P:000125 70F400  020000  move   #PMemSize,n0 ; Set to Size of P:EPROM
151
152 P:000127 200013  clr     a
153 P:000128 20001B  clr     b
154 P:000129 560000  move   a,x:CKSUM_CALC_P ; Initialize computed checksum -> $000000
155 P:00012A 560100  move   a,x:CKSUM_READ_P ; Initialize read checksum -> $000000
156
157 ;----- Compute the 24-bit P: Space Checksum -----
158 P:00012B 06D810  000003  dor    n0,p_loop ;
159 P:00012D 07D88E  move    p:(r0)+,a ; Get the P:EPROM location Value
160 P:00012E 200018  add     a,b ; Compute checksum
161 p_loop
162
163 P:00012F 07E08E  move    p:(r0),a ; Get Checksum from P:EPROM
164 P:000130 570000  move    b,x:CKSUM_CALC_P ; Save the Computed Checksum value
165 P:000131 560100  move    a,x:CKSUM_READ_P ; Save the Old Checksum value
166
167 ;*****
168 ;*****
169 do_x_checksum
170 P:000132 05F420  FFFFFFFF  move   #-1,m0 ; Set LINEAR addressing mode
171 P:000134 60F400  100000  move   #XMemStart,r0 ; Set Starting Address of X:EPROM
172 P:000136 70F400  020000  move   #XMemSize,n0 ; Set to Size of X:EPROM
173
174 P:000138 200013  clr     a
175 P:000139 20001B  clr     b
176 P:00013A 560200  move   a,x:CKSUM_CALC_X ; Initialize computed checksum -> $000000
177 P:00013B 560300  move   a,x:CKSUM_READ_X ; Initialize read checksum -> $000000
178
179 ;----- Compute the 24-bit X: Space Checksum -----
180 P:00013C 06D810  000003  dor    n0,x_loop ;
181 P:00013E 56D800  move    x:(r0)+,a ; Get the X:EPROM location Value
182 P:00013F 200018  add     a,b ; Compute checksum
183 x_loop
184
185 P:000140 56E000  move    x:(r0),a ; Get Checksum from X:EPROM
186 P:000141 570200  move    b,x:CKSUM_CALC_X ; Save the Computed Checksum value
187 P:000142 560300  move    a,x:CKSUM_READ_X ; Save the Old Checksum value
188
189 ;*****
190 ;*****
191 do_y_checksum
192 P:000143 05F420  FFFFFFFF  move   #-1,m0 ; Set LINEAR addressing mode
193 P:000145 60F400  100000  move   #YMemStart,r0 ; Set Starting Address of Y:EPROM
194 P:000147 70F400  020000  move   #YMemSize,n0 ; Set to Size of Y:EPROM
195
196 P:000149 200013  clr     a
197 P:00014A 20001B  clr     b
198 P:00014B 560400  move   a,x:CKSUM_CALC_Y ; Initialize computed checksum -> $000000
199 P:00014C 560500  move   a,x:CKSUM_READ_Y ; Initialize read checksum -> $000000
200
201 ;----- Compute the 24-bit Y: Space Checksum -----
202 P:00014D 06D810  000003  dor    n0,y_loop ;
203 P:00014F 5ED800  move    y:(r0)+,a ; Get the Y:EPROM location Value
204 P:000150 200018  add     a,b ; Compute checksum
205 y_loop
206
207 P:000151 5EE000  move    y:(r0),a ; Get Checksum from Y:EPROM
208 P:000152 570400  move    b,x:CKSUM_CALC_Y ; Save the Computed Checksum value
209 P:000153 560500  move    a,x:CKSUM_READ_Y ; Save the Read Checksum value
210
211 ;*****

```

```
212
213   P:000154 050C00      bra *                ; Done, Do a dynamic Halt
214
215                        end      eprom1

0  Errors
0  Warnings
```

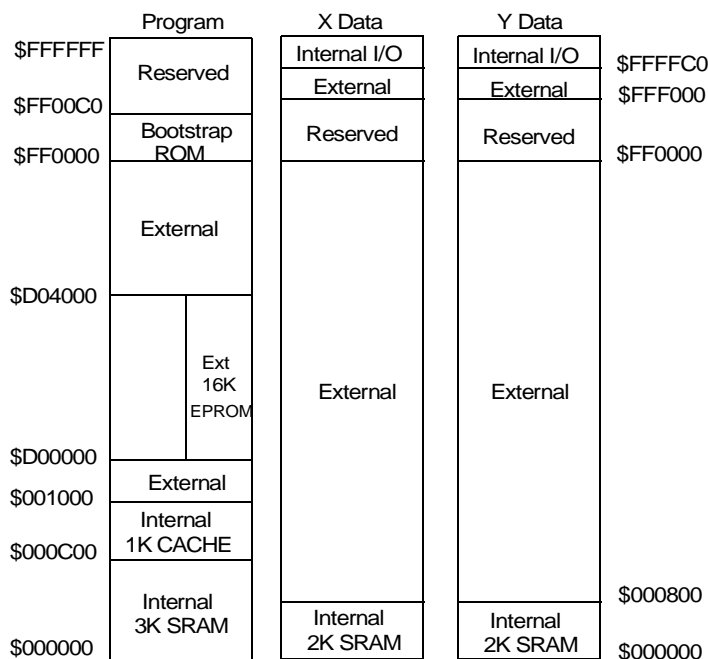
## 4.2 16K x 8-bit BOOT EPROM Example

This section describes a 16K x 8-bit 'P' memory space Boot-strap, BOOT, EPROM implementation using AMD's AM27C128 device (see **Figure 4-6** for the memory map layout, **Figure 4-7** for the block diagram and **Figure 4-9** for the schematic). Using an EPROM in this configuration, a program placed in the EPROM can be loaded into the DSP at Boot time and run.

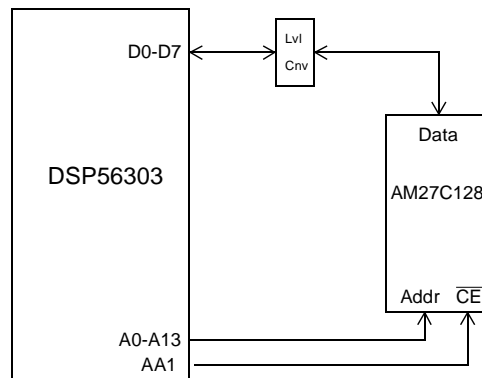
For this example, the DSP core runs at 80 MHz and the input frequency source is a 4.000 MHz crystal. For a 250 nS EPROM, 20 wait states are required. This 5 V device is organized as 16K x 8-bits with a 250 nS access time. One memory device composes the 8-bit wide 'P' bus.

Level conversion to and from 3.3 V and 5 V is necessary on the 8-bit data bus to accommodate the 5 V memory device. This is accomplished by using one of Quality Semiconductor's QS3245 QuickSwitch<sup>®</sup> 8-bit bus switches. This switch allows the connection of 3.3 V CMOS logic (the DSP data bus) on one side and 5 V TTL-compatible logic (the 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.

During RESET with Mode 1 selected, the DSP BOOT code configures Address Attribute Line 1 for Program accesses in the address range from \$D00000 to \$DFFFFFF. The BOOT code then loads bytes from the EPROM, packs them into 24-bit words and stores them into Program RAM. The first word, three packed bytes, read from the EPROM indicates the number of words to load. The second word from the EPROM contains the starting load address for the packed data. This starting load address is also the address which gains program control after the program load is completed. The program listed in **Example 4-2** initializes the Address Attribute Register and calculates an 8-bit checksum value used to compare with the checksum value stored in the last location of the 16K EPROM bank.



**Figure 4-6.** 16K x 8-bit Boot EPROM Memory Map



**Figure 4-7.** 16K x 8-bit EPROM Memory Example.

## 4.2.1 EPROM Timing Requirements

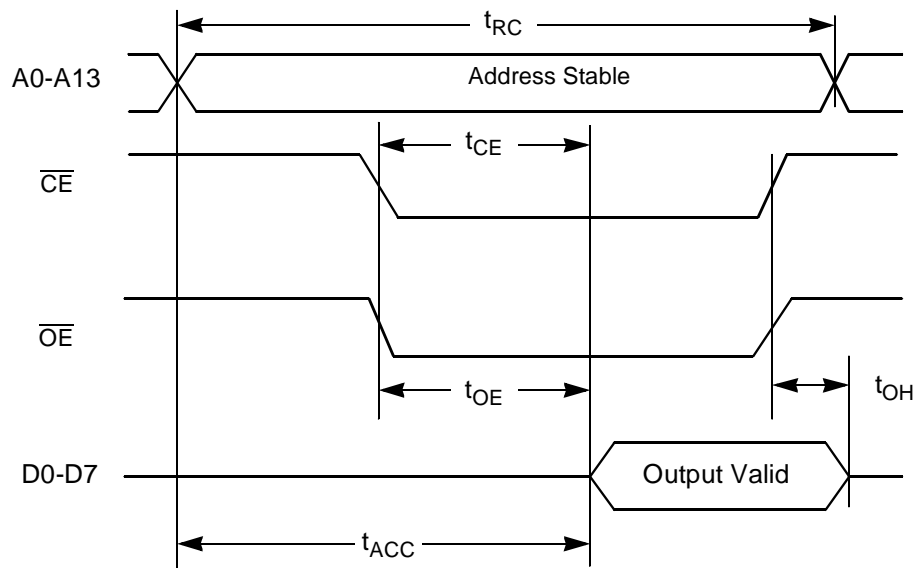
For the EPROM device to work properly, its timing requirements must be met. Following are the timing requirements for the AM27C128-250 16K x 8-bit 250 nS EPROM.

### 4.2.1.1 AM27C128-250 Read Cycle Timing

**Table 4-3** shows the memory read timing specification values used in the memory read cycle timing diagram, **Figure 4-8**.

**Table 4-3.** AM27C128-250 Memory Read Timing Specifications

Read Cycle Parameter	Symbol	Min	Max
Read Cycle Time	$t_{RC}$	250 nS	-
Address to Output Delay	$t_{ACC}$	-	250 nS
Chip Enable to Output Delay	$t_{CE}$	-	250 nS
Output Enable to Output Delay	$t_{OE}$	-	100 nS
Output Hold Time from Address, $\overline{CE}$ or $\overline{OE}$ . Whichever occurs first.	$t_{OH}$	0 nS	-



**Figure 4-8.** AM27C128 Memory Read Cycle Timing Diagram

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

For optimal use of the 64K x8-bit BOOT EPROM 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 = 0x0
- Low-power Divider value = 1, Bits 12-14 = 0x0
- VCO Multiplication value = 20, Bits 0-11 = 0x013
- 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 0x0E0013.

Address Attribute Pin 1 enables, via EPROM  $\overline{CE}$ , external 16K EPROM memory bank accesses in the address range from \$D00000 to \$D03FFF during program space requests. Configure the memory address space requirements for the Address Attribute Pin 1 using the Address Attribute Register 1(AAR1). The AAR1 register value combines the following bits for each feature:

- Specify the external memory access type as asynchronous SRAM, Bits 0-1 = 0x1.
- 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 = 0xA;
- Specify the most significant portion of the address to compare, Bits 12-23 = 0x100.

The value loaded into the AAR1 register is 0xD00A09. The value loaded into AAR0, AAR2 and AAR3 is 0x000000.

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

- Address attribute area zero wait states, Bits 0-4 = 0x0
- Address attribute area one wait state, Bits 5-9 = 0x14
- Address attribute area two wait states, Bits 10-12 = 0x0
- Address attribute area three wait states, Bits 13-15 = 0x0
- Default address area wait states, Bits 16-20 = 0x0
- 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 0x000280.

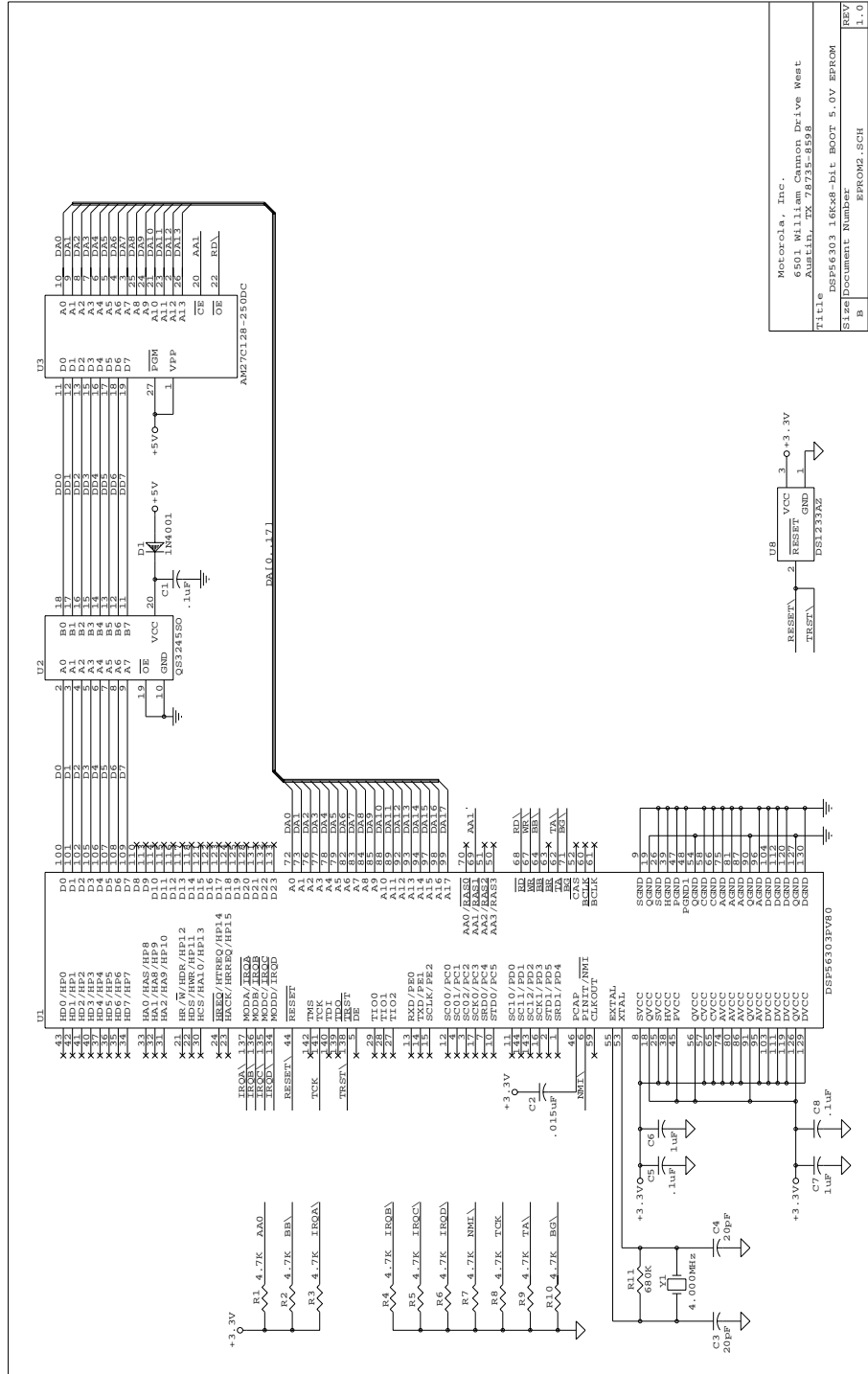


Figure 4-9. 16Kx8 BOOT EPROM Schematic



**Example 4-2. 16K x 8-bit BOOT EPROM Checksum Program**

Motorola DSP56300 Assembler Version 6.0.1.6 97-02-26 21:35:23 eprom2.asm

```

1          page      132,60,3,3,
2          ;
3          ;      eprom2.asm - Simple program to calculate the 8-bit Checksum for
4          ;      a 16K x 8-bit block of EPROM memory using a DSP56303.
5          ;      Contains: Initialization routine,
6          ;      Routine to calculate 8-bit checksum.
7          ;
8          ;      The program runs in Internal P:RAM to calculate the checksum on
9          ;      External P:EPROM from $D00000 - $D03FFF @ 20w/s
10         ;
11
12         D00000      MemStart   equ      $D00000
13         D04000      MemEnd     equ      $D04000
14         004000      MemSize    equ      MemEnd-MemStart      ; Last Word is stored Checksum value
15
16         ;--- Program Specific Storage Locations (X DATA SPACE)
17         NEW_CHECKSUM
18         000000      equ      $000000      ; Computed Checksum Value
19         OLD_CHECKSUM
20         000001      equ      $000001      ; Old Checksum from PEROM
21
22         ;--- DSP56303 Control Registers (X I/O SPACE)
23         BCR         equ      $FFFFFB      ; Bus Control Register
24         PCTL        equ      $FFFFFD      ; PLL Control Register
25         AAR1        equ      $FFFFF8      ; Address Attribute Register 1
26
27         ;--- PCTL value = 0x0E0013
28         000000      prediv     equ      0          ; Pre-Divider = 1
29         000000      lowdiv     equ      0          ; Low Power Divider = 1
30         000013      pllmul     equ      19         ; VCO Mult = 20; (19+1)*4.00 MHz=80.00 MHz
31         000000      crystal    equ      0          ; No, Crystal not less than 200 kHz
32         000000      disXTAL    equ      0          ; No, do not disable crystal use
33         020000      pllstop     equ      $020000    ; Yes, PLL runs during STOP
34         040000      enpll      equ      $040000    ; Yes, enable PLL operation
35         080000      disclk     equ      $080000    ; Yes, disable CORE clock output
36         0E0013      PCTL_value equ      prediv+lowdiv+pllmul+crystal+disXTAL+pllstop+enpll+disclk
37
38         ;--- AAR1 value = 0xD00A09
39         000001      acctype1    equ      1          ; External Memory access type = 0x1
40         000000      aahigh1     equ      0          ; Enable AAI pin to be low when selected
41         000008      aap1       equ      $8          ; Yes, Enable AAI pin on ext 'P' accesses
42         000000      aax1       equ      0          ; No, Enable AAI pin on ext 'X' accesses
43         000000      aay1       equ      0          ; No, Enable AAI pin on ext 'Y' accesses
44         000000      aswap1     equ      0          ; No, Enable address bus swap
45         000000      enpack1    equ      0          ; No, Enable packing/unpacking logic
46         000A00      nadd1      equ      $000A00    ; Compare 10 address bits
47         D00000      msadd1     equ      $D00000    ; Most significant portion of address,
48         ; $D00000 - D03fff, to compare.
49         ; (1101,0000,00xx,xxxx,xxxx,xxxx)
50         D00A09      AAR1_value equ      acctype1+aahigh1+aap1+aax1+aay1+aswap1+enpack1+nadd1+msadd1
51
52
53         ;--- BCR value = 0x000280
54         000000      aaa0ws     equ      0          ; Address Attribute Area 0 w/s = 0
55         000280      aaalws     equ      $280        ; Address Attribute Area 1 w/s = 20
56         000000      aaa2ws     equ      0          ; Address Attribute Area 2 w/s = 0
57         000000      aaa3ws     equ      0          ; Address Attribute Area 3 w/s = 0
58         000000      defws      equ      0          ; Default Address Area w/s = 0
59         000000      busss      equ      0          ; Bus state status = 0
60         000000      enblh      equ      0          ; Enable Bus Lock Hold = 0
61         000000      enbrh      equ      0          ; Enable Bus Request Hold = 0
62         000280      BCR_value  equ      aaa0ws+aaalws+aaa2ws+aaa3ws+defws+busss+enblh+enbrh
63
64         ;-----
65         ;      Header for DSP56303 BOOT Code
66         ;-----
67         P:0000FE      org      p:$100-2
68
69         P:0000FE      dc      pgm_end-eprom2      ; Number of words in program
70         P:0000FF      dc      eprom2              ; Starting address for program
71
72         ;-----
73
74         P:000100      org      p:$100              ; Keep the program in internal RAM

```

## Erasable Programmable Read Only Memory

```

75
76
77     P:000100 0D1080      bsr      eprom2      init      ; Initialize DSP
              000005
78
79     P:000102 0D1080      bsr      calc_checksum ; Calculate Checksum of EPROM
              00000C
80
81     _done
82     P:000104 050C00      bra      *           ; DONE, Do a dynamic HALT
83
84
85     ;-----
86     ; Initialization Section
87     ;-----
88     init
89     P:000105 08F4BD      movep   #PCTL_value,x:PCTL      ; Set PLL Control Register
              0E0013
90     P:000107 05F439      movec   #080000,SR             ; Enable 1K Cache
              080000
91     P:000109 08F4BB      movep   #BCR_value,x:BCR       ; Set external wait states
              000280
92     P:00010B 08F4B8      movep   #AAR1_value,x:AAR1     ; Set Address Attribute Regl
              D00A09
93
94     P:00010D 00000C      rts
95
96     ;-----
97     ; Routine to Calculate 8-bit Checksum
98     ;-----
99     calc_checksum
100    P:00010E 05F420      move    #-1,m0                ; Set LINEAR addressing mode
              FFFFFFFF
101    P:000110 60F400      move    #MemStart,r0          ; Set Starting Address of EPROM
              D00000
102    P:000112 70F400      move    #MemSize-1,n0         ; Set to Size of Flash - Checksum
              003FFF
103
104    P:000114 200013      clr     a
105    P:000115 20001B      clr     b
106    P:000116 540000      move    a1,x:NEW_CHECKSUM     ; Initialize computed checksum -> $000000
107    P:000117 540100      move    a1,x:OLD_CHECKSUM     ; Initialize read checksum -> $000000
108
109    ; Compute the 8-bit Checksum
110    P:000118 44F400      move    #>$FF,x0             ; Set lower Byte Mask
              0000FF
111
112    P:00011A 06D810      dor     n0,_ploop
              000004
113    P:00011C 07D88C      move    p:(r0)+,a1           ; Get the EPROM location Value
114    P:00011D 200046      and     x0,a                  ; Mask for lower byte
115    P:00011E 200018      add     a,b                   ; Compute checksum
              _ploop
116
117
118    P:00011F 07E08C      move    p:(r0),a1            ; Get EPROM's Old Checksum value
119    P:000120 20004E      and     x0,b                  ; Limit calculated Checksum to lower byte
120    P:000121 200046      and     x0,a                  ; Limit Old Checksum to lower byte
121    P:000122 550000      move    b1,x:NEW_CHECKSUM     ; Save the Computed Checksum value
122    P:000123 540100      move    a1,x:OLD_CHECKSUM     ; Save the Old Checksum value
123
124    P:000124 00000C      rts
125
126    ;-----
127    pgm_end
128
129    end      eprom2

```

```

0 Errors
0 Warnings

```

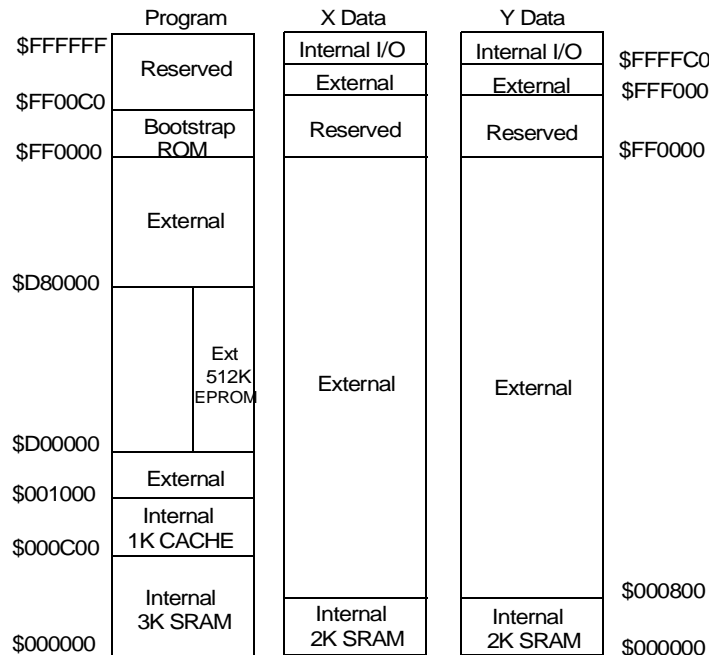
### 4.3 512K x 8-bit BOOT/Overlay EPROM Example

This section describes a 512K x 8-bit BOOT with X data Space program overlay EPROM implementation using Atmel's AT27LV040-15 device (see **Figure 4-10** for the BOOT memory map layout, **Figure 4-11** for the overlay memory map layout, **Figure 4-12** for the block diagram, and **Figure 4-14** for the schematic). By using an EPROM in this configuration, a program placed in the EPROM can be loaded into the DSP at Boot time and run. The booted program can then load additional programs and data from the EPROM into the DSP, via overlay techniques.

For this example, the DSP core runs at 80 MHz and the input frequency source is a 4.000 MHz crystal. For a 150 nS EPROM, 12 wait states are required. This 3.3 V device is organized as 512K x 8-bits with a 150 nS access time. One memory device composes the 8-bit wide BOOT bus.

During RESET with Mode 1 selected, the DSP BOOT code configures Address Attribute Line 1 for program accesses in the address range from \$D00000 to \$DFFFFFF (see **Figure 4-10**). The BOOT code then loads bytes from the EPROM, packs them into 24-bit words and stores them into Program RAM. The first word, three packed bytes, read from the EPROM indicates the number of words to load. The second word from the EPROM contains the starting load address for the packed data. This starting load address is also the address which gains program control after the program load is completed.

Under user program control, the Address Attribute Line 1 can then be configured for X data space accesses in the address range from \$100000 to \$17FFFF (see **Figure 4-12**). Address Attribute Line 0 can then be configured to select the upper half of the 512K EPROM by enabling EPROM A18 to be a one when X:\$14FFFF to X:\$17FFFF is selected. The program listed in **Example 4-3** initializes the Address Attribute Register and calculates an 8-bit checksum value used to compare with the checksum value stored in the last location of the 512K EPROM bank.



**Figure 4-10.** 512K x 8-bit BOOT EPROM Memory Map

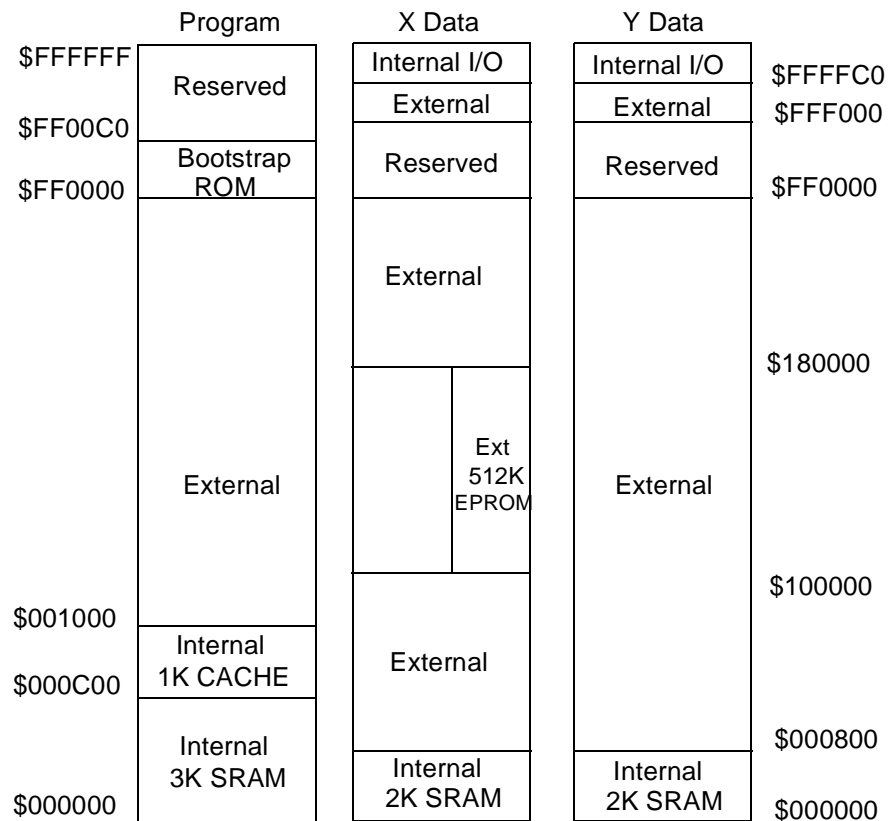


Figure 4-11. 512K x 8-bit Overlay EPROM Memory Map

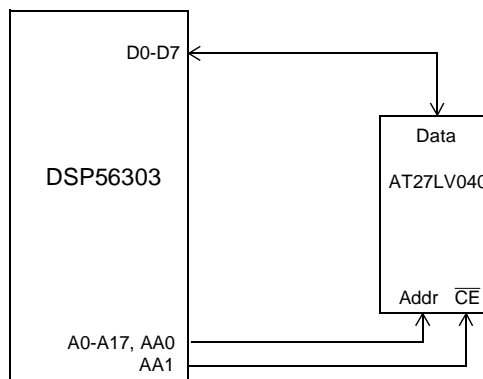


Figure 4-12. 512K x 8-bit EPROM Memory Example.

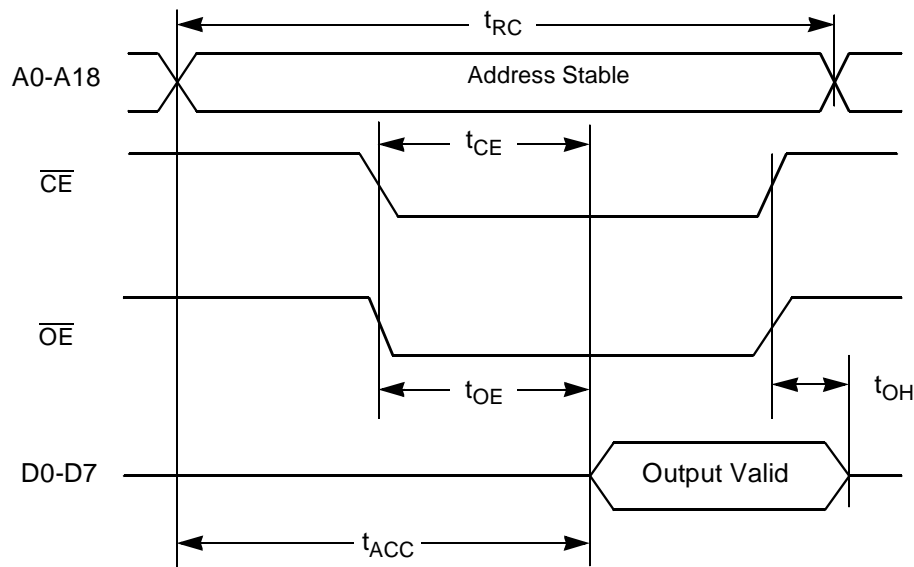
### 4.3.1 EPROM Timing Requirements

For the EPROM device to work properly, its timing requirements must be met. Following are the timing requirements for the AT27LV040-15 512K x 8-bit 150 nS EPROM.

**Table 4-4** contains the memory read timing specification values used in the memory read cycle timing diagram, **Figure 4-13**.

**Table 4-4.** AT27LV040-15 Memory Read Timing Specifications

Read Cycle Parameter	Symbol	Min	Max
Read Cycle Time	$t_{RC}$	150 nS	-
Address to Output Delay	$t_{ACC}$	-	150 nS
Chip Enable to Output Delay	$t_{CE}$	-	150 nS
Output Enable to Output Delay	$t_{OE}$	-	60 nS
Output Hold Time from Address, $\overline{CE}$ or $\overline{OE}$ . Whichever occurs first.	$t_{OH}$	0 nS	-



**Figure 4-13.** AT27LV040 Memory Read Cycle Timing Diagram.

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

For optimal use of the 512K x 8-bit BOOT/Overlay EPROM 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 = 0x0
- Low-power Divider value = 1, Bits 12-14 = 0x0
- VCO Multiplication value = 20, Bits 0-11 = 0x013
- 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 0x0E0013.

Address Attribute Pin 1 enables, via EPROM  $\overline{CE}$ , external 512K EPROM bank accesses in the address range from \$100000 to \$17FFFF during X data space requests. Configure the memory address space requirements for the Address Attribute Pin 1 using the Address Attribute Register 1 (AAR1). The AAR1 register value combines the following bits for each feature:

- Specify the external memory access type as asynchronous SRAM, Bits 0-1 = 0x1.
- 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 = 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 = 1.
- Specify the number of address bits to compare, Bits 8-11 = 0x5.
- Specify the most significant portion of the address to compare, Bits 12-23 = 0x100.

The value loaded into the AAR1 register is 0x100591.

Address Attribute Pin 0 selects, via EPROM A18, the upper address of the external 512K EPROM memory bank accesses in the address range from \$140000 to \$17FFFF during X data space requests. Configure the memory address space requirements for the Address Attribute Pin 0 using the Address Attribute Register 0 (AAR0). The AAR0 register value combines the following bits for each feature:

- Specify the external memory access type as asynchronous SRAM, Bits 0-1 = 0x1.
- 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 = 1.
- Specify the number of address bits to compare, Bits 8-11 = 0x6.
- Specify the most significant portion of address to compare, Bits 12-23 = 0x140.

The value loaded into the AAR0 register is 0x140695. The value loaded into AAR2 and AAR3 is 0x000000.

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

- Address attribute area zero wait states, Bits 0-4 = 0xC
- Address attribute area one wait states, Bits 5-9 = 0xC
- Address attribute area two wait states, Bits 10-12 = 0x0
- Address attribute area three wait states, Bits 13-15 = 0x0
- Default address area wait states, Bits 16-20 = 0x0
- 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 0x00018C.

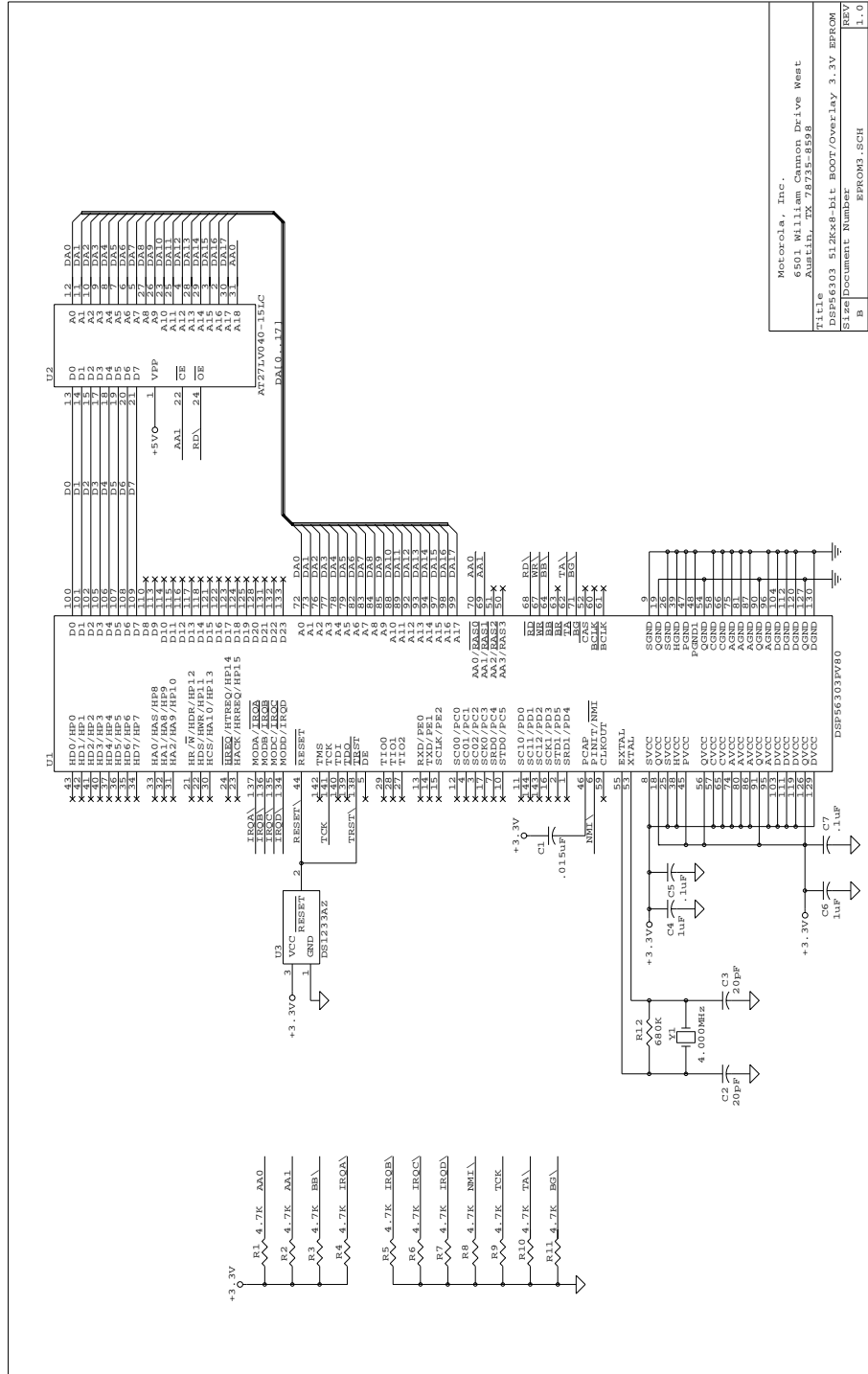


Figure 4-14. 512Kx8 BOOT/Overlay EPROM Schematic

Motorola, Inc.  
6501 William Cannon Drive West  
Austin, TX 78732-9398

Title: DSP56303 512Kx8-bit BOOT/Overlay 3.3V EPROM  
Size: Document Number  
B EPR003.SCH  
Date: February 26, 1997 Sheet 1 of 1



**Example 4-3. 512K x 8-bit BOOT/Overlay EPROM Checksum Program**

Motorola DSP56300 Assembler Version 6.0.1.6 97-02-27 21:07:11 eprom3.asm

```

1
2
3
4
5
6
7
8
9
10
11
12      100000      MemStart      equ $100000
13      180000      MemEnd        equ $180000
14      080000      MemSize       equ MemEnd-MemStart      ; Last Word is stored Checksum value
15
16      ;--- Program Specific Storage Locations (X DATA SPACE)
17      NEW_CHECKSUM
18      000000      equ $000000      ; Computed Checksum Value
19      OLD_CHECKSUM
20      000001      equ $000001      ; Old Checksum from EPROM
21
22      ;--- DSP56303 Control Registers (X I/O SPACE)
23      FFFFFB      BCR            equ $FFFFFFB      ; Bus Control Register
24      FFFFFD      PCTL          equ $FFFFFFD      ; PLL Control Register
25      FFFFF9      AAR0          equ $FFFFFF9      ; Address Attribute Register 0
26      FFFFF8      AAR1          equ $FFFFFF8      ; Address Attribute Register 1
27
28      ;--- PCTL value = 0x0E0013
29      000000      prediv         equ 0              ; Pre-Divider = 1
30      000000      lowdiv         equ 0              ; Low Power Divider = 1
31      000013      pllmul         equ 19             ; VCO Mult = 20; (19+1)*4.00 MHz=80.00 MHz
32      000000      crystal        equ 0              ; No, Crystal not less than 200 kHz
33      000000      disXTAL        equ 0              ; No, do not disable crystal use
34      020000      pllstop        equ $020000       ; Yes, PLL runs during STOP
35      040000      enpll          equ $040000       ; Yes, enable PLL operation
36      080000      disclk         equ $080000       ; Yes, disable CORE clock output
37      0E0013      PCTL_value     equ prediv+lowdiv+pllmul+crystal+disXTAL+pllstop+enpll+disclk
38
39      ;--- AAR0 value = 0x140695
40      000001      acctype0        equ 1              ; External Memory access type = 0x1
41      000004      aahigh0         equ $4              ; Enable AA0 pin to be high when selected
42      000000      aap0            equ 0              ; No, Enable AA0 pin on ext 'P' accesses
43      000010      aax0            equ $10             ; Yes, Enable AA0 pin on ext 'X' accesses
44      000000      aay0            equ 0              ; No, Enable AA0 pin on ext 'Y' accesses
45      000000      aswap0          equ 0              ; No, Enable address bus swap
46      000080      enpack0         equ $80             ; Yes, Enable packing/unpacking logic
47      000600      nadd0           equ $000600       ; Compare 6 address bits
48      140000      msadd0          equ $140000       ; Most significant portion of address,
49
50
51
52
53
54      140695      AAR0_value     equ acctype0+aahigh0+aap0+aax0+aay0+aswap0+enpack0+nadd0+msadd0
55
56      ;--- AAR1 value = 0x100591
57      000001      acctype1        equ 1              ; External Memory access type = 0x1
58      000000      aahigh1         equ 0              ; Enable AA1 pin to be low when selected
59      000000      aap1            equ 0              ; No, Enable AA1 pin on ext 'P' accesses
60      000010      aax1            equ $10             ; Yes, Enable AA1 pin on ext 'X' accesses
61      000000      aay1            equ 0              ; No, Enable AA1 pin on ext 'Y' accesses
62      000000      aswap1          equ 0              ; No, Enable address bus swap
63      000080      enpack1         equ $80             ; Yes, Enable packing/unpacking logic
64      000500      nadd1           equ $000500       ; Compare 5 address bits
65      100000      msadd1          equ $100000       ; Most significant portion of address,
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999

```

## Erasable Programmable Read Only Memory

```

75      000000      enblh      equ 0                ; Enable Bus Lock Hold = 0
76      000000      enbrh      equ 0                ; Enable Bus Request Hold = 0
77      00018C      BCR_value  equ aaa0ws+aaalws+aaa2ws+aaa3ws+defws+busss+enblh+enbrh
78
79      ;-----
80      ;           Header for DSP56303 Boot Code
81      ;-----
82      P:0000FE      org p:$100-2
83
84      P:0000FE      dc pgm_end-eprom3      ; Number of words in program
85      P:0000FF      dc eprom3          ; Starting address for program
86
87
88      ;-----
89      P:000100      org p:$100                ; Keep the program in internal RAM
90
91      eprom3
92      P:000100      0D1080          bsr          init          ; Initialize DSP
93                      000005
94      P:000102      0D1080          bsr          calc_checksum      ; Calculate Checksum of EPROM
95                      000010
96
97      P:000104      050C00          bra          *                ; DONE, Do a dynamic HALT
98
99
100     ;-----
101     ;           Initialization Section
102     ;-----
103     init
104     P:000105      08F4BD          movep      #PCTL_value,x:PCTL      ; Set PLL Control Register
105                      0E0013
106     P:000107      05F43A          movec      #$004000,OMR          ; Disable Address Attribute Priorities
107                      004000
108     P:000109      05F439          movec      #$080000,SR          ; Enable 1K Cache
109                      080000
110     P:00010B      08F4BB          movep      #BCR_value,x:BCR      ; Set external wait states
111                      00018C
112     P:00010D      08F4B9          movep      #AAR0_value,x:AAR0     ; Set Address Attribute Reg0
113                      140695
114     P:00010F      08F4B8          movep      #AAR1_value,x:AAR1     ; Set Address Attribute Reg1
115                      100591
116
117     P:000111      00000C          rts
118
119     ;-----
120     ;           Routine to Calculate 8-bit Checksum
121     ;-----
122     calc_checksum
123     P:000112      05F420          move       #-1,m0                ; Set LINEAR addressing mode
124                      FFFFFFFF
125     P:000114      60F400          move       #MemStart,r0          ; Set Starting Address of EPROM
126                      100000
127     P:000116      70F400          move       #MemSize-1,n0         ; Set to Size of Flash - Checksum
128                      07FFFF
129
130     P:000118      200013          clr        a
131     P:000119      20001B          clr        b
132     P:00011A      540000          move       a1,x:NEW_CHECKSUM     ; Initialize computed checksum -> $000000
133     P:00011B      540100          move       a1,x:OLD_CHECKSUM     ; Initialize read checksum -> $000000
134
135     ; Compute the 8-bit Checksum
136     P:00011C      44F400          move       #>$FF,x0              ; Set lower Byte Mask
137                      0000FF
138
139     P:00011E      06D810          dor        n0,_ploop
140                      000004
141     P:000120      54D800          move       x:(r0)+,a1            ; Get the EPROM location Value
142     P:000121      200046          and        x0,a                  ; Mask for lower byte
143     P:000122      200018          add        a,b                    ; Compute checksum
144     _ploop
145
146     P:000123      54E000          move       x:(r0),a1            ; Get EPROM's Old Checksum value
147     P:000124      20004E          and        x0,b                  ; Limit calculated Checksum to lower byte
148     P:000125      200046          and        x0,a                  ; Limit Old Checksum to lower byte
149     P:000126      550000          move       b1,x:NEW_CHECKSUM     ; Save the Computed Checksum value
150     P:000127      540100          move       a1,x:OLD_CHECKSUM     ; Save the Old Checksum value
151
152

```

```
141 P:000128 00000C rts
142
143 ;-----
144 pgm_end
145
146 end eprom3

0 Errors
0 Warnings
```



## 5 Advantages

EPROM, EEPROM, PROM, and ROM memory provide the designer with a non-volatile memory storage at a reasonable speed. EPROM, EEPROM, PROM and ROM memory allows the DSP to load programs and run immediately after RESET, as in an embedded application. Programs stored in the EPROM's non-volatile memory storage can be read and run directly as 24-bit data or indirectly as 8-bit data, using the DMA to access, pack and store the data into DSP Program memory. Additionally, programs and data can be stored in the EEPROM's non-volatile memory with the device still in the circuit. However, for new programs or data to be stored in the EPROM's non-volatile memory, the EPROM must be removed from the circuit, erased by exposure to Ultraviolet Light (UV), and then reprogrammed by an EPROM programmer. For development, this is very inconvenient. However, for production this is possibly very cost-effective. ROM parts are purchased with program data already stored on the part. This allows easier assembly and lower cost compared to EPROM, EEPROM, and PROM.

### 5.1 Flexible Memory Configuration Capabilities

EPROM, EEPROM, PROM, and ROM memory provide the designer with flexible memory configurations for the DSP56300 family. Using a common hardware memory design with the DSP56300 family 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 address attribute control registers, allows the memory bank to be configured for the following:

- Seven different memory space arrangements if one address attribute control line is used:
  1. Memory Bank 'P' Space
  2. Memory Bank 'X' Space
  3. Memory Bank 'Y' Space
  4. Memory Bank 'P'/'X' Space
  5. Memory Bank 'P'/'Y' Space
  6. Memory Bank 'X'/'Y' Space
  7. Memory Bank 'P'/'X'/'Y' Space
- Thirteen different memory space arrangements if two address attribute control lines are used:
  1. Memory Bank 'P' Space
  2. Memory Bank 'X' Space
  3. Memory Bank 'Y' Space
  4. Memory Bank 'P'/'X' Space
  5. Memory Bank 'P'/'Y' Space
  6. Memory Bank 'X'/'Y' Space
  7. Memory Bank 'P'/'X'/'Y' Space

8. Memory Bank/2 'P'/'X' and Memory Bank/2 'Y' Space
9. Memory Bank/2 'P'/'Y' and Memory Bank/2 'X' Space
10. Memory Bank/2 'P' and Memory Bank/2 'X'/'Y' Space
11. Memory Bank/2 'P' and Memory Bank/2 'X' Space
12. Memory Bank/2 'P' and Memory Bank/2 'Y' Space
13. Memory Bank/2 'X' and Memory Bank/2 'Y' Space
- Fourteen different memory space arrangements if three address attribute control lines are used:
  1. Memory Bank/2 'P' Space
  2. Memory Bank/2 'X' Space
  3. Memory Bank/2 'Y' Space
  4. Memory Bank/2 'P'/'X' Space
  5. Memory Bank/2 'P'/'Y' Space
  6. Memory Bank/2 'X'/'Y' Space
  7. Memory Bank/2 'P'/'X'/'Y' Space
  8. Memory Bank/2 'P'/'X' and Memory Bank/2 'Y' Space
  9. Memory Bank/2 'P'/'Y' and Memory Bank/2 'X' Space
  10. Memory Bank/2 'P' and Memory Bank/2 'X'/'Y' Space
  11. Memory Bank/2 'P' and Memory Bank/2 'X' Space
  12. Memory Bank/2 'P' and Memory Bank/2 'Y' Space
  13. Memory Bank/2 'X' and Memory Bank/2 'Y' Space
  14. Memory bank/4 BOOT, memory bank/4 'P', memory bank/4 'X' and memory bank/4 'Y' Space

### 5.1.1 Immediate Load and Run Capability

EPROM, EEPROM, PROM, and ROM memory allows the DSP to load programs and run them immediately after RESET, as in an embedded application. Programs stored in the EPROM's non-volatile memory storage can be read and run directly as 24-bit data. Arranging the EPROM, EEPROM, PROM, or ROM memory in a 24-bit bank of memory allows the DSP to run directly from the memory after a RESET. This was shown in the 128Kx24-bit BOOT, 128Kx24-bit Program, 128Kx24-bit 'X', and 128Kx24-bit 'Y' EPROM example.

### 5.1.2 Program and Data Overlay Capability

Programs stored in the EPROM's non-volatile memory storage can be read and run indirectly as 8-bit data, using a program or DMA to access, pack and store the data into DSP program memory. Arranging the EPROM, EEPROM, PROM, and ROM memory in an 8-bit bank allows the DSP at

Boot time to load and pack the 8-bit memory values into 24-bit words for storage into Program RAM. After the data loading is complete, the DSP turns over program control to this newly loaded program. This program, overlay loader, can then programmatically or via DMA control, load additional data from the 8-bit memory bank into Program, X data or Y data spaces for later execution or use. This was shown in the 2K x 8-bit Boot EEPROM, 32K x 8-bit EEPROM, 16K x 8-bit EPROM and 512K x 8-bit EPROM examples.

### 5.1.3 EEPROM In-circuit Reprogrammability

Programs and data can be stored in the EEPROM's non-volatile memory with the device still in the circuit. This allows for easy in-circuit program and data modification or upgrading. For development, this is very convenient, allowing for fast test-program-retest development cycles.

## 5.2 Disadvantages

For new programs or data to be stored in an EPROM, the EPROM must be removed from the circuit, erased by exposure to Ultraviolet Light (UV), and then reprogrammed by an EPROM programmer. For development, this is very inconvenient.


New programs or data storage in a PROM requires a new unprogrammed PROM to be programmed by a PROM programmer for the part. Then the old parts have to be removed from existing boards. This is very inconvenient for a product and impractical during development.

For new programs or data to be stored in a ROM part, a new ROM mask and new parts must be manufactured. Then the old parts have to be removed from existing boards. This is very inconvenient for a product and impractical during development.

## 5.3 Speed Selection

For EPROM, EEPROM, PROM, and ROM memory speed selection, the critical timing specification used is typically based on the memory's data access time,  $t_{AA}$ . However, all timing specifications have to be met and should always be reviewed for compliance.


OnCE and Mfax are registered trademarks of Motorola, Inc.

Motorola and  are registered trademarks of Motorola, Inc.

QuickSwitch is a registered trademark of Quality Semiconductor, Inc.

All other trademarks are those of their respective owners.

®Reg. U.S. Pat. & Tm. Off.

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola 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 Motorola 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. Motorola does not convey any license under its patent rights nor the rights of others. Motorola 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 life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola 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 Motorola was negligent regarding the design or manufacture of the part. Motorola and  are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

How to reach us:

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

Motorola Literature Distribution  
P.O. Box 5405  
Denver, Colorado 80217  
1 (800) 441-2447  
1 (303) 675-2140

**Motorola Fax Back System (Mfax™):**

TOUCHTONE (602) 244-6609  
USA and Canada ONLY:  
1 (800) 774-1848  
RMFAX0@email.sps.mot.com

**Asia/Pacific:**

Motorola Semiconductors H.K. Ltd.  
8B Tai Ping Industrial Park  
51 Ting Kok Road  
Tai Po, N.T., Hong Kong  
852-26629298

**Technical Resource Center:**

1 (800) 521-6274

**DSP Helpline**

dsphelp@dsp.sps.mot.com

**Japan:**

Nippon Motorola Ltd  
SPD, Strategic Planning Office 141  
4-32-1, Nishi-Gotanda  
Shinagawa-ku, Japan  
81-3-5487-8488

**Internet:**

<http://www.motorola-dsp.com/>