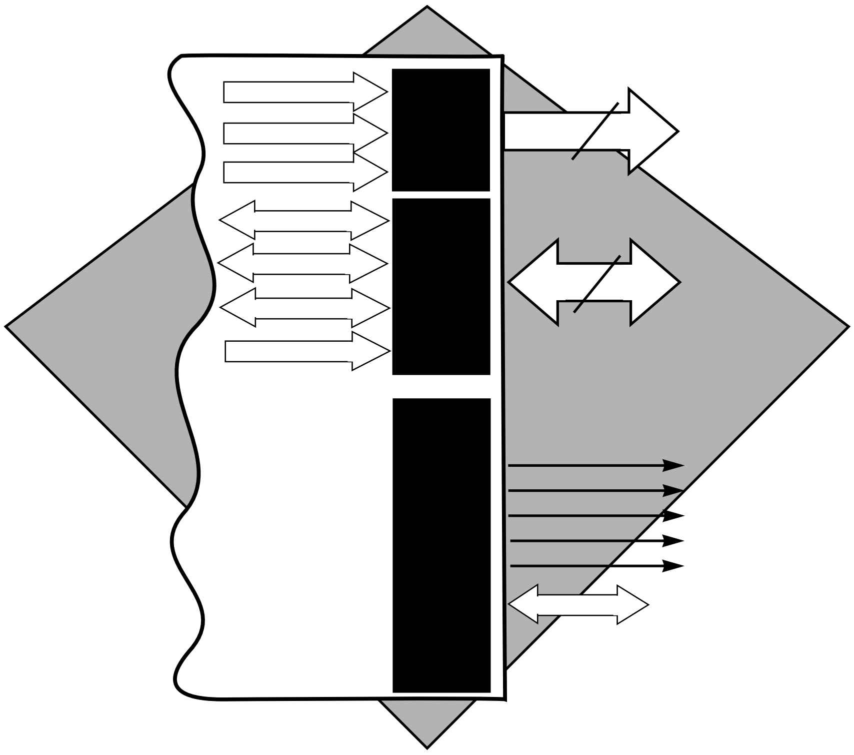
SECTION 4

CORE CONFIGURATION



4.1 INTRODUCTION

This chapter presents DSP56300 core configuration details specific to the DSP56307. These configuration details include the following:

- Operating modes
- Bootstrap program
- Interrupt sources and priorities
- DMA request sources
- OMR
- PLL control register
- AA control registers
- JTAG boundary scan register

For information on specific registers or modules in the DSP56300 core, refer to the *DSP56300 Family Manual*.

4.2 OPERATING MODES

The DSP56307 begins operation by leaving the Reset state and going into one of eight operating modes. As the DSP56307 exits the Reset state, it loads the values of MODA, MODB, MODC, and MODD into bits MA, MB, MC, and MD of the OMR. These bit settings determine the chip's operating mode, which determines the bootstrap program option the chip uses to start up.

The MA–MD bits of the OMR can also be set directly by software. A jump directly to the bootstrap program entry point ($FF0000) after the OMR bits are set causes the DSP56307 to execute the specified bootstrap program option (except modes 0 and 8).

Table 4-1 shows the DSP56307 bootstrap operation modes, the corresponding settings of the external operational mode signal lines (the mode bits MA–MD in the OMR), and the reset vector address to which the DSP56307 jumps once it leaves the Reset state.

Table 4-1 DSP56307 Operating Modes

Mode	MODD	MODC	MODB	MODA	Reset Vector	Description
0	0	0	0	0	$C00000	Expanded mode [1]
1	0	0	0	1	$FF0000	Reserved
2	0	0	1	0	$FF0000	Reserved
3	0	0	1	1	$FF0000	Reserved
4	0	1	0	0	$FF0000	Reserved
5	0	1	0	1	$FF0000	Reserved
6	0	1	1	0	$FF0000	Reserved
7	0	1	1	1	$FF0000	Reserved
8	1	0	0	0	$008000	Expanded mode
9	1	0	0	1	$FF0000	Bootstrap from byte-wide memory
A	1	0	1	0	$FF0000	Bootstrap through SCI
B	1	0	1	1	$FF0000	Reserved
C	1	1	0	0	$FF0000	HI08 bootstrap in ISA/DSP5630X mode
D	1	1	0	1	$FF0000	HI08 bootstrap in HC11 nonmultiplexed mode
E	1	1	1	0	$FF0000	HI08 bootstrap in 8051 multiplexed bus mode
F	1	1	1	1	$FF0000	HI08 bootstrap in MC68302 bus mode
Note: 1. Address $C00000 is reflected as address $00000 on Port A signals A0–A17.						

4.2.1 Mode 0—Expanded Mode

Mode	MODD	MODC	MODB	MODA	Reset Vector	Description
0	0	0	0	0	$C00000	Expanded mode

The bootstrap ROM is bypassed and the DSP56307 starts fetching instructions beginning at address $C00000. Memory accesses are performed using SRAM memory access type with 31 wait states and no address attributes selected (default).

4.2.2 Modes 1 to 7: Reserved

These modes are reserved for future use.

4.2.3 Mode 8—Expanded Mode

Mode	MODD	MODC	MODB	MODA	Reset Vector	Description
8	1	0	0	0	$008000	Expanded mode

The bootstrap ROM is bypassed and the DSP56307 starts fetching instructions beginning at address $008000. Memory accesses are performed using SRAM memory access type with 31 wait states and no address attributes selected.

4.2.4 Mode 9—Boot from Byte-Wide External Memory

Mode	MODD	MODC	MODB	MODA	Reset Vector	Description
9	1	0	0	1	$FF0000	Bootstrap from byte-wide memory (at $D00000)

The bootstrap program loads instructions through Port A from external byte-wide memory, starting at P:$D00000. The SRAM memory access type is selected by the values in address attribute register 1 (AAR1). Thirty-one wait states are inserted between each memory access. Address $D00000 is reflected as address $00000 on Port A signals A0–A17. The boot program concatenates every 3 bytes read from the external memory into a 24-bit wide DSP56307 word.

4.2.5 Mode A—Boot from SCI

Mode	MODD	MODC	MODB	MODA	Reset Vector	Description
A	1	0	1	0	$FF0000	Bootstrap through SCI

Instructions are loaded through the SCI. The bootstrap program sets the SCI to operate in 10-bit asynchronous mode, with 1 start bit, 8 data bits, 1 stop bit, and no parity. Data is received in this order: start bit, 8 data bits (LSB first), and one stop bit. Data is aligned in the SCI receive data register with the LSB of the least significant byte of the received data appearing at Bit 0.The user must provide an external clock source with a frequency at least 16 times the transmission data rate. Each byte received by the SCI is echoed back through the SCI transmitter to the external transmitter. The boot program concatenates every 3 bytes read from the SCI into a 24-bit wide DSP56307 word.

4.2.6 Mode B—Reserved

Mode	MODD	MODC	MODB	MODA	Reset Vector	Description
B	1	0	1	1	$FF0000	Reserved

This mode is reserved for future use.

4.2.7 Mode C—Boot from HI08 in ISA Mode (8-Bit Bus)

Mode	MODD	MODC	MODB	MODA	Reset Vector	Description
C	1	1	0	0	$FF0000	HI08 bootstrap in ISA/DSP5630X

In this mode, the HI08 is configured to interface with an ISA bus or with the memory expansion port of a master DSP5630*n* processor through the HI08. The HI08 pin configuration is optimized for connection to the ISA bus or memory expansion port of a master DSP based on DSP56300 core.

4.2.8 Mode D—Boot from HI08 in HC11 Nonmultiplexed Mode

Mode	MODD	MODC	MODB	MODA	Reset Vector	Description
D	1	1	0	1	$FF0000	HI08 Bootstrap in HC11 nonmultiplexed

The bootstrap program sets the host interface to interface with the Motorola HC11 microcontroller through the HI08. The HI08 pin configuration is optimized for connection to Motorola HC11 nonmultiplexed bus.

.

4.2.9 Mode E—Boot from HI08 in 8051 Multiplexed Bus Mode

Mode	MODD	MODC	MODB	MODA	Reset Vector	Description
E	1	1	1	0	$FF0000	HI08 bootstrap in 8051 multiplexed bus

The bootstrap program sets the host interface to interface with the Intel 8051 bus through the HI08. The program stored in this location, after testing MODA, MODB, MODC, and MODD, bootstraps through HI08. The HI08 pin configuration is optimized for connection to the Intel 8051 multiplexed bus.

4.2.10 Mode F—Boot from HI08: MC68302/68360 Bus Mode

Mode	MODD	MODC	MODB	MODA	Reset Vector	Description
7	F	1	1	1	$FF0000	HI08 bootstrap in MC68302 bus

The bootstrap program sets the host interface to interface with the Motorola MC68302 or MC68360 bus through the HI08. The HI08 pin configuration is optimized for connection to a Motorola MC68302 or MC68360 bus.

4.3 BOOTSTRAP PROGRAM

The bootstrap program is factory-programmed in an internal 192-word by 24-bit bootstrap ROM located in program memory space at locations $FF0000–$FF00BF. The bootstrap program can load any program RAM segment from an external byte-wide EPROM, the SCI, or the host port. The bootstrap program code is listed in **Appendix A Bootstrap Programs**.

Upon exiting the Reset state, the DSP56307 performs these steps:

1. Samples the MODA, MODB, MODC, and MODD signal lines.
2. Loads their values into bits MA, MB, MC, and MD in the OMR.

As explained in Section 4.2, the mode input signals (MODA–MODD) and the resulting MA, MB, MC, and MD bits determine which bootstrap mode the DSP56307 enters:

- If MA, MB, MC, and MD are all cleared (bootstrap mode 0), the program bypasses the bootstrap ROM, and the DSP56307 starts loading instructions from external program memory location $C00000.
- If MA, MB, and MC are cleared and MD is set (bootstrap mode 8), the program bypasses the bootstrap ROM and the DSP56307 starts loading in instruction values from external program memory location $008000.
- Otherwise, the DSP56307 jumps to the bootstrap program entry point at $FF0000.

Note: The bootstrap in any HI08 bootstrap mode may be stopped by setting the Host Flag 0 (HF0). This starts execution of the loaded user program from the specified starting address.

You can invoke the bootstrap program options (except modes 0 and 8) at any time by setting the MA, MB, MC, and MD bits in the OMR and jumping to the bootstrap program entry point, $FF0000. Software can directly set the mode selection bits in the OMR. Bootstrap modes 0 and 8 are the normal DSP56307 functioning modes. Bootstrap modes 9, A, and C–F select different specific bootstrap loading source devices. In these modes, the bootstrap program expects the following data sequence when downloading the user program through an external port:

1. Three bytes defining the number of (24-bit) program words to be loaded
2. Three bytes defining the (24-bit) start address to which the user program loads in the DSP56307 program memory
3. The user program (three bytes for each 24-bit program word)

Note: The three bytes for each data sequence must be loaded least significant byte first.

Once the bootstrap program completes loading the specified number of words, it jumps to the specified starting address and executes the loaded program.

4.4 INTERRUPT SOURCES AND PRIORITIES

DSP56307 interrupt handling, like that of all DSP56300 family members, is optimized for DSP applications. Refer to Section 7 of the *DSP56300 Family Manual.* The interrupt table is located in the 256 locations of program memory to which by the vector base address (VBA) register in the PCU points.

4.4.1 Interrupt Sources

Because each interrupt is allocated two instructions in the table, there are 128 table entries for interrupt handling. **Table 4-2** shows the table entry address for each interrupt source. The DSP56307 initialization program loads the table entry for each interrupt serviced with two interrupt servicing instructions. In the DSP56307, only some of the 128 vector addresses are used for specific interrupt sources. The remaining interrupt vectors are reserved and can be used for host NMI (IPL = 3) or for host command interrupt (IPL = 2). If certain interrupts are not to be used, those interrupt vector locations can be used for program or data storage.

Table 4-2 Interrupt Sources

Interrupt Starting Address	Interrupt Priority Level Range	Interrupt Source
VBA:$00	3	Hardware $\overline{\text{RESET}}$
VBA:$02	3	Stack error
VBA:$04	3	Illegal instruction
VBA:$06	3	Debug request interrupt
VBA:$08	3	Trap
VBA:$0A	3	Nonmaskable interrupt ($\overline{\text{NMI}}$)
VBA:$0C	3	Reserved
VBA:$0E	3	Reserved
VBA:$10	0–2	$\overline{\text{IRQA}}$
VBA:$12	0–2	$\overline{\text{IRQB}}$
VBA:$14	0–2	$\overline{\text{IRQC}}$
VBA:$16	0–2	$\overline{\text{IRQD}}$
VBA:$18	0–2	DMA channel 0
VBA:$1A	0–2	DMA channel 1
VBA:$1C	0–2	DMA channel 2
VBA:$1E	0–2	DMA channel 3
VBA:$20	0–2	DMA channel 4
VBA:$22	0–2	DMA channel 5
VBA:$24	0–2	TIMER 0 compare
VBA:$26	0–2	TIMER 0 overflow
VBA:$28	0–2	TIMER 1 compare
VBA:$2A	0–2	TIMER 1 overflow
VBA:$2C	0–2	TIMER 2 compare
VBA:$2E	0–2	TIMER 2 overflow
VBA:$30	0–2	ESSI0 receive data
VBA:$32	0–2	ESSI0 receive data with exception status
VBA:$34	0–2	ESSI0 receive last slot
VBA:$36	0–2	ESSI0 transmit data
VBA:$38	0–2	ESSI0 transmit data with exception status
VBA:$3A	0–2	ESSI0 transmit last slot

Table 4-2 Interrupt Sources (Continued)

Interrupt Starting Address	Interrupt Priority Level Range	Interrupt Source
VBA:$3C	0–2	Reserved
VBA:$3E	0–2	Reserved
VBA:$40	0–2	ESSI1 receive data
VBA:$42	0–2	ESSI1 receive data with exception status
VBA:$44	0–2	ESSI1 receive last slot
VBA:$46	0–2	ESSI1 transmit data
VBA:$48	0–2	ESSI1 transmit data with exception status
VBA:$4A	0–2	ESSI1 transmit last slot
VBA:$4C	0–2	Reserved
VBA:$4E	0–2	Reserved
VBA:$50	0–2	SCI receive data
VBA:$52	0–2	SCI receive data with exception status
VBA:$54	0–2	SCI transmit data
VBA:$56	0–2	SCI idle line
VBA:$58	0–2	SCI timer
VBA:$5A	0–2	Reserved
VBA:$5C	0–2	Reserved
VBA:$5E	0–2	Reserved
VBA:$60	0–2	Host receive data full
VBA:$62	0–2	Host transmit data empty
VBA:$64	0–2	Host command (default)
VBA:$66	0–2	Reserved
VBA:$68	0 - 2	EFCOP data input buffer empty
VBA:$6A	0 - 2	EFCOP data output buffer full
VBA:$6C	0 - 2	Reserved
VBA:$6E	0 - 2	Reserved
:	:	:
VBA:$FE	0–2	Reserved

4.4.2 Interrupt Priority Levels

There are two interrupt priority registers in the DSP56307. The IPR–C is dedicated to DSP56300 core interrupt sources, and IPR–P is dedicated to DSP56307 peripheral interrupt sources. IPR–C is shown on **Figure 4-1** and IPR–P is shown in **Figure 4-2**.

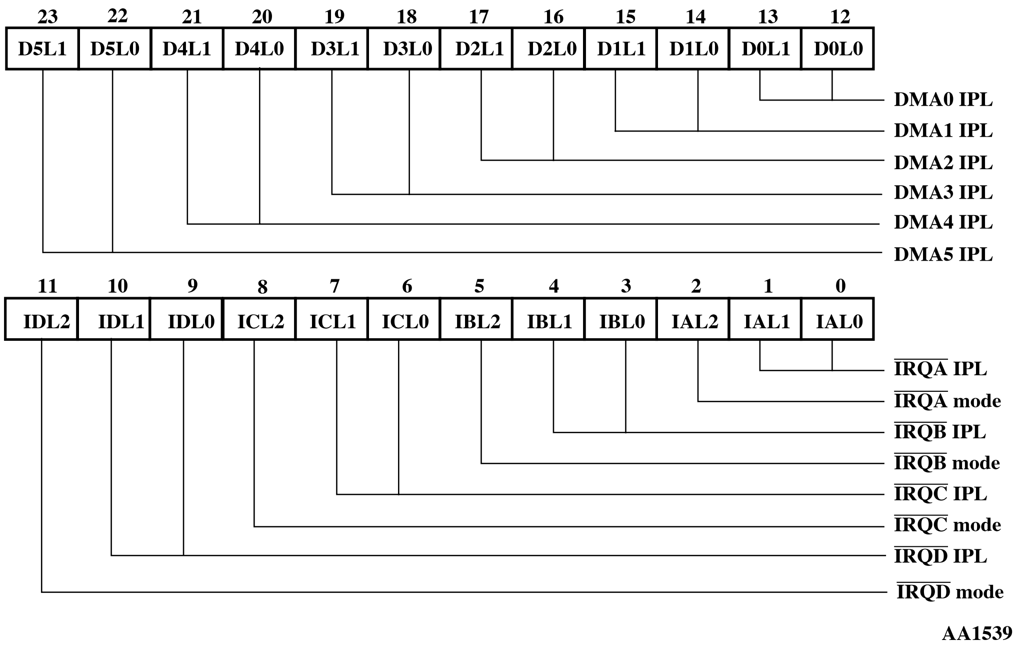

23 22 21 20 19 18 17 16 15 14 13 12
D5L1 D5L0 D4L1 D4L0 D3L1 D3L0 D2L1 D2L0 D1L1 D1L0 D0L1 D0L0
DMA0 IPL
DMA1 IPL
DMA2 IPL
DMA3 IPL
DMA4 IPL
DMA5 IPL
11 10 9 8 7 6 5 4 3 2 1 0
IDL2 IDL1 IDL0 ICL2 ICL1 ICL0 IBL2 IBL1 IBL0 IAL2 IAL1 IAL0
$\overline{\text{IRQA}}$ IPL
$\overline{\text{IRQA}}$ mode
$\overline{\text{IRQB}}$ IPL
$\overline{\text{IRQB}}$ mode
$\overline{\text{IRQC}}$ IPL
$\overline{\text{IRQC}}$ mode
$\overline{\text{IRQD}}$ IPL
$\overline{\text{IRQD}}$ mode
AA1539

Figure 4-1 Interrupt Priority Register C (IPR-C) (X:$FFFFFF)

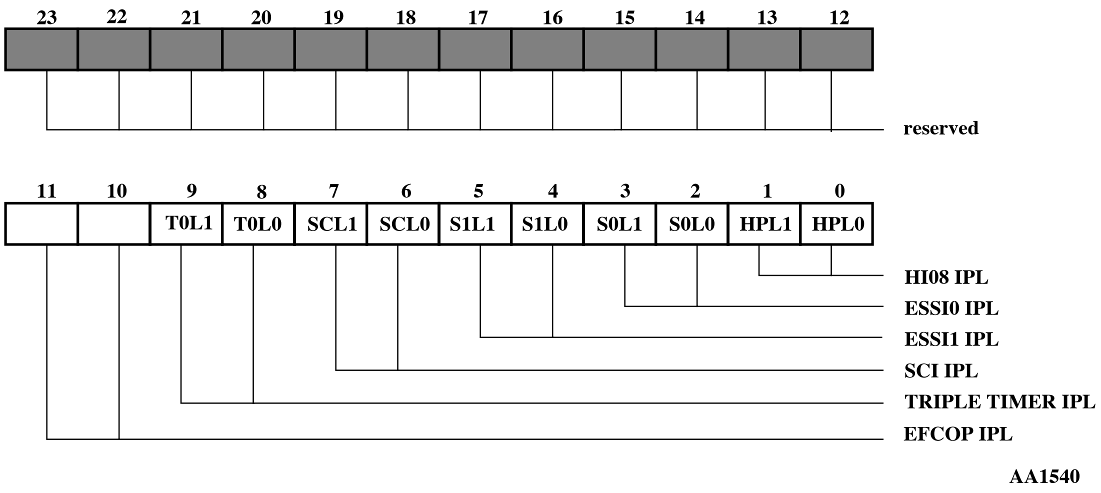


Figure 4-2 Interrupt Priority Register P (IPR-P) (X:$FFFFFE)

Table 4-3 Interrupt Priority Level Bits

IPL bits		Interrupts Enabled	Interrupts Masked	Interrupt Priority Level
xxL1	xxL0			
0	0	No	—	0
0	1	Yes	0	1
1	0	Yes	0, 1	2
1	1	Yes	0, 1, 2	3

4.4.3 Interrupt Source Priorities within an IPL

If more than one interrupt request is pending when an instruction executes, the interrupt source with the highest IPL is serviced first. When several interrupt requests with the same IPL are pending, another fixed-priority structure within that IPL determines which interrupt source is serviced first. This fixed-priority list of interrupt sources within an IPL is shown in **Table 4-4**.

Table 4-4 Interrupt Source Priorities within an IPL

Priority	Interrupt Source
Level 3 (nonmaskable)	
Highest	Hardware $\overline{\text{RESET}}$
	Stack error
	Illegal instruction
	Debug request interrupt
	Trap
Lowest	Nonmaskable interrupt
Levels 0, 1, 2 (maskable)	
Highest	$\overline{\text{IRQA}}$ (external interrupt)
	$\overline{\text{IRQB}}$ (external interrupt)
	$\overline{\text{IRQC}}$ (external interrupt)
	$\overline{\text{IRQD}}$ (external interrupt)
	DMA channel 0 interrupt

Table 4-4 Interrupt Source Priorities within an IPL (Continued)

Priority	Interrupt Source
	DMA channel 1 interrupt
	DMA channel 2 interrupt
	DMA channel 3 interrupt
	DMA channel 4 interrupt
	DMA channel 5 interrupt
	Host command interrupt
	Host transmit data empty
	Host receive data full
	ESSI0 RX data with exception interrupt
	ESSI0 RX data interrupt
	ESSI0 receive last slot interrupt
	ESSI0 TX data with exception interrupt
	ESSI0 transmit last slot interrupt
	ESSI0 TX data interrupt
	ESSI1 RX data with exception interrupt
	ESSI1 RX data interrupt
	ESSI1 receive last slot interrupt
	ESSI1 TX data with exception interrupt
	ESSI1 transmit last slot interrupt
	ESSI1 TX data interrupt
	SCI receive data with exception interrupt
	SCI receive data
	SCI transmit data
	SCI idle line
	SCI timer
	TIMER0 overflow interrupt
	TIMER0 compare interrupt
	TIMER1 overflow interrupt
	TIMER1 compare interrupt

Table 4-4 Interrupt Source Priorities within an IPL (Continued)

Priority	Interrupt Source
	TIMER2 overflow interrupt
	TIMER2 compare interrupt
	EFCOP data input buffer empty
Lowest	EFCOP data output buffer full

4.5 DMA REQUEST SOURCES

The DMA request source bits (DRS[4:0]) in the DMA control/status registers) encode the source of DMA requests used to trigger DMA transfers. The DMA request sources may be internal peripherals or external devices requesting service through the $\overline{\text{IRQA}}$, $\overline{\text{IRQB}}$, $\overline{\text{IRQC}}$, or $\overline{\text{IRQD}}$ signals. **Table 4-5** shows the values of the DRS bits.

Table 4-5 DMA Request Sources

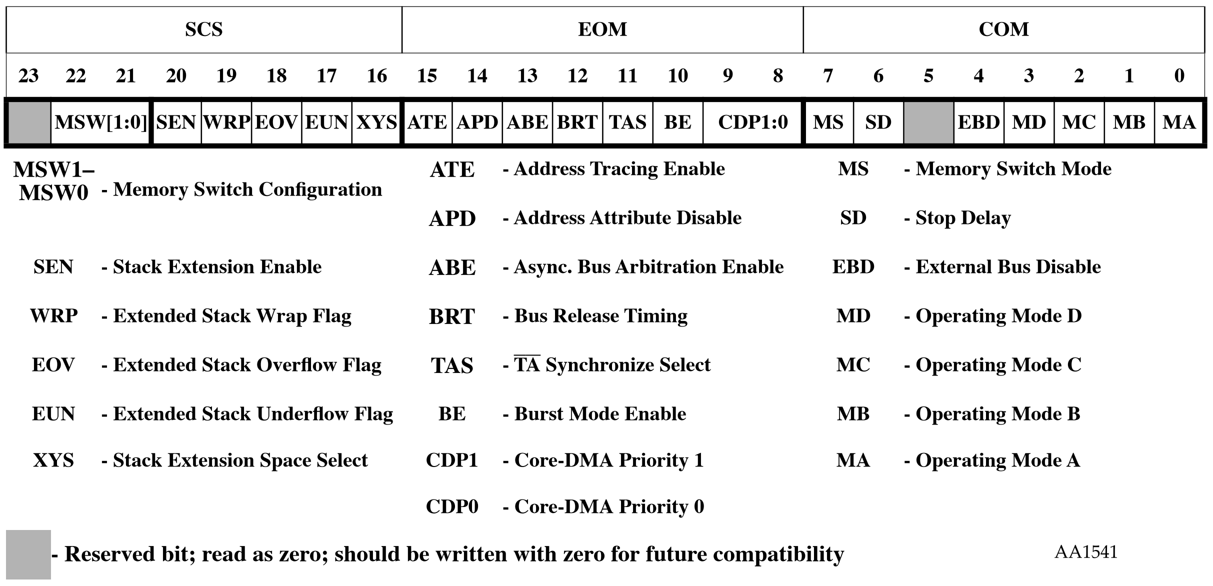
DMA Request Source Bits DRS4 . . . DRS0	Requesting Device
00000	External ($\overline{\text{IRQA}}$ signal)
00001	External ($\overline{\text{IRQB}}$ signal)
00010	External ($\overline{\text{IRQC}}$ signal)
00011	External ($\overline{\text{IRQD}}$ signal)
00100	Transfer done from DMA channel 0
00101	Transfer done from DMA channel 1
00110	Transfer done from DMA channel 2
00111	Transfer done from DMA channel 3
01000	Transfer done from DMA channel 4
01001	Transfer done from DMA channel 5
01010	ESSI0 receive data (RDF0 = 1)
01011	ESSI0 transmit data (TDE0 = 1)
01100	ESSI1 receive data (RDF1 = 1)
01101	ESSI1 transmit data (TDE1 = 1)
01110	SCI receive data (RDRF = 1)
01111	SCI transmit data (TDRE = 1)

Table 4-5 DMA Request Sources (Continued)

DMA Request Source Bits DRS4 . . . DRS0	Requesting Device
10000	Timer0 (TCF0 = 1)
10001	Timer1 (TCF1 = 1)
10010	Timer2 (TCF2 = 1)
10011	Host receive data full (HRDF = 1)
10100	Host transmit data empty (HTDE = 1)
10101	EFCOP input buffer empty (FDIBE=1)
10110	EFCOP output buffer full (FDOBF=1)
10111–11111	Reserved

4.6 OMR

The OMR is a 24-bit read/write register divided into three byte-sized units. The lowest two bytes (EOM and COM) are used to control the chip's operating mode. The high byte (SCS) is used to control and monitor the stack extension. The OMR control bits are shown in **Figure 4-3**. See the *DSP56300 Family Manual* for a complete description of the OMR.



SCS								EOM								COM							
23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
▨	MSW[1:0]		SEN	WRP	EOV	EUN	XYS	ATE	APD	ABE	BRT	TAS	BE	CDP1:0		MS	SD	▨	EBD	MD	MC	MB	MA

MSW1–MSW0	- Memory Switch Configuration	ATE	- Address Tracing Enable	MS	- Memory Switch Mode
		APD	- Address Attribute Disable	SD	- Stop Delay
SEN	- Stack Extension Enable	ABE	- Async. Bus Arbitration Enable	EBD	- External Bus Disable
WRP	- Extended Stack Wrap Flag	BRT	- Bus Release Timing	MD	- Operating Mode D
EOV	- Extended Stack Overflow Flag	TAS	- $\overline{\text{TA}}$ Synchronize Select	MC	- Operating Mode C
EUN	- Extended Stack Underflow Flag	BE	- Burst Mode Enable	MB	- Operating Mode B
XYS	- Stack Extension Space Select	CDP1	- Core-DMA Priority 1	MA	- Operating Mode A
		CDP0	- Core-DMA Priority 0		

▨ - Reserved bit; read as zero; should be written with zero for future compatibility

AA1541

Figure 4-3 DSP56307 Operating Mode Register (OMR) Format

4.7 PLL CONTROL REGISTER

The PLL control register (PCTL) is an X-I/O mapped, 24-bit read/write register used to direct the operation of the on-chip PLL. The PCTL control bits are shown in **Figure 4-4**. See the *DSP56300 Family Manual* for a full description of the PCTL.

23	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**
PD3	**PD2**	**PD1**	**PD0**	**COD**	**PEN**	**PSTP**	**XTLD**	**XTLR**	**DF2**	**DF1**	**DF0**
11	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MF11	**MF10**	**MF9**	**MF8**	**MF7**	**MF6**	**MF5**	**MF4**	**MF3**	**MF2**	**MF1**	**MF0**

AA0852

Figure 4-4 PLL Control Register (PCTL)

4.7.1 Predivider Factor Bits (PD[3:0])—PCTL Bits 23–20

The predivider factor bits (PD[3:0]) define the predivision factor (PDF) to be applied to the PLL input frequency. The PD[3:0] bits are cleared during DSP56307 hardware reset, which corresponds to a PDF of one.

4.7.2 Clock Output Disable (COD) Bit—PCTL Bit 19

The COD bit controls the output buffer of the clock at the CLKOUT pin. When COD is set, the CLKOUT output is pulled high. When COD is cleared, the CLKOUT pin provides a 50% duty cycle clock synchronized to the internal core clock.

4.7.3 PLL Enable (PEN) Bit—PCTL Bit 18

The PEN bit enables PLL operation.

4.7.4 PLL Stop State (PSTP) Bit—Bit 17

The PSTP bit controls PLL and on-chip crystal oscillator behavior during the stop processing state.

4.7.5 XTAL Disable (XTLD) Bit—PCTL Bit 16

The XTLD bit controls the on-chip crystal oscillator XTAL output. The XTLD bit is cleared during DSP56307 hardware reset; consequently, the XTAL output signal is active, permitting normal operation of the crystal oscillator.

4.7.6 Crystal Range (XTLR) Bit—PCTL Bit 15

The XTLR bit controls the on-chip crystal oscillator transconductance. The XTLR bit is set to a predetermined value during hardware reset.

4.7.7 PCTL Bits 14–12

The division factor bits DF[2:0] define the DF of the low-power divider. These bits specify the DF as a power of two in the range from 2^0 to 2^7.

4.7.8 PLL Multiplication Factor—PCTL Bits 11–0

The multiplication factor bits (MF[11:0]) define the multiplication factor (MF) that is applied to the PLL input frequency. The MF bits are cleared during DSP56307 hardware reset, and thus it corresponds to an MF of one.

4.8 DEVICE IDENTIFICATION REGISTER (IDR)

The IDR is a 24-bit, read-only factory-programmed register that identifies DSP56300 family members. It specifies the derivative number and revision number of the device. This information is used in testing or by software. **Figure 4-5** shows the contents of the IDR.

Revision numbers are assigned as follows: $0 is revision 0, $1 is revision A, and so on.

23 16	15 12	11 0
Reserved	**Revision Number**	**Derivative Number**
$00	**$0**	**$307**

AA1503

Figure 4-5 Identification Register Configuration (Revision 0)

4.9 ADDRESS ATTRIBUTE REGISTERS (AAR1–AAR4)

An AAR is shown in Figure 4-6. There are four of these registers in the DSP56307 (AAR0–AAR3), one for each address attribute signal. For a full description of the address attribute registers see the *DSP56300 Family Manual*. Address multiplexing is not supported by the DSP56307. Bit 6 (BAM) of the AAR is reserved and should be written with 0 only.

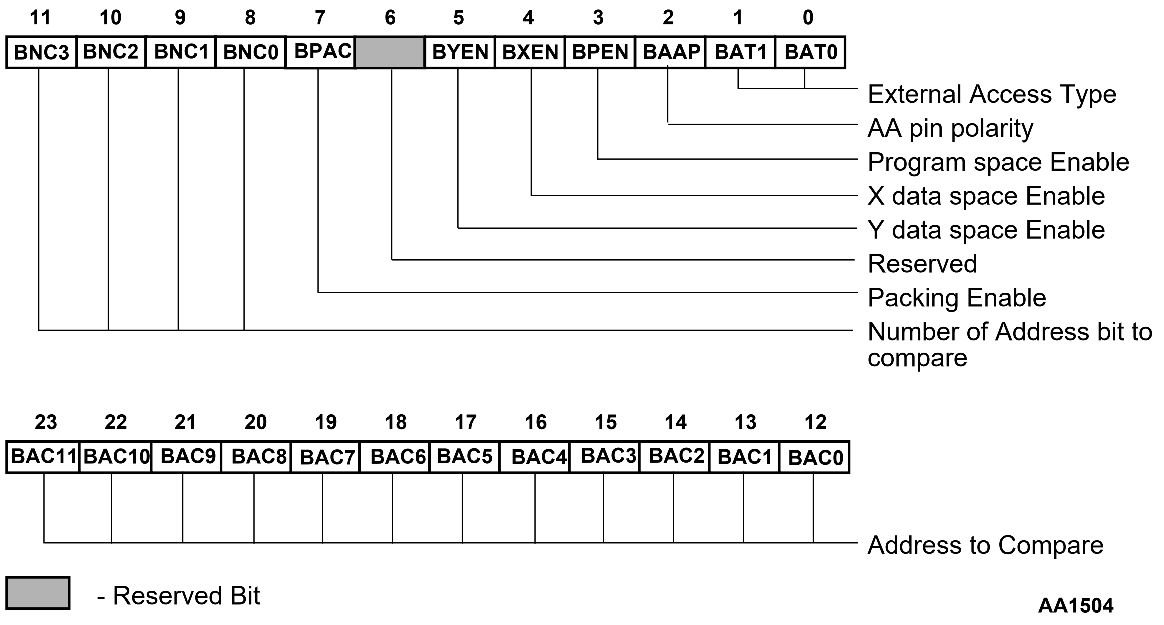


Figure 4-6 Address Attribute Registers (AAR0–AAR3) (X:$FFFFF9–$FFFFF6)

4.10 JTAG IDENTIFICATION (ID) REGISTER

The JTAG ID register is a 32-bit read-only factory-programmed register that distinguishes the component on a board according to the IEEE 1149.1 standard. **Figure 4-7** shows the JTAG ID register configuration. Version information corresponds to the revision number ($0 for revision 0, $1 for revision A, etc.).

31 28	27 22	21 12	11 1	0
Version Information	Customer Part Number	Sequence Number	Manufacturer Identity	1
0000	000110	0000000111	00000001110	1 AA1505

Figure 4-7 JTAG Identification Register Configuration (Revision 0)

4.11 JTAG BOUNDARY SCAN REGISTER (BSR)

The BSR in the DSP56307 JTAG implementation contains bits for all device signals, clock pins, and their associated control signals. All DSP56307 bidirectional pins have a corresponding register bit in the BSR for pin data and are controlled by an associated control bit in the BSR. The BSR is documented in **Section 12 Joint Test Action Group Port** of this manual, and the JTAG code listing is in **Appendix C DSP56307 BSDL Listing**.