
6 PROGRAM CONTROL UNIT

This section describes the program control unit (PCU) hardware and its programming model. The instruction pipeline description is also included since understanding the pipeline is particularly important in understanding the DSP56300 Core. Note that the pipelined operation remains essentially hidden from the user, thus easing programmability.

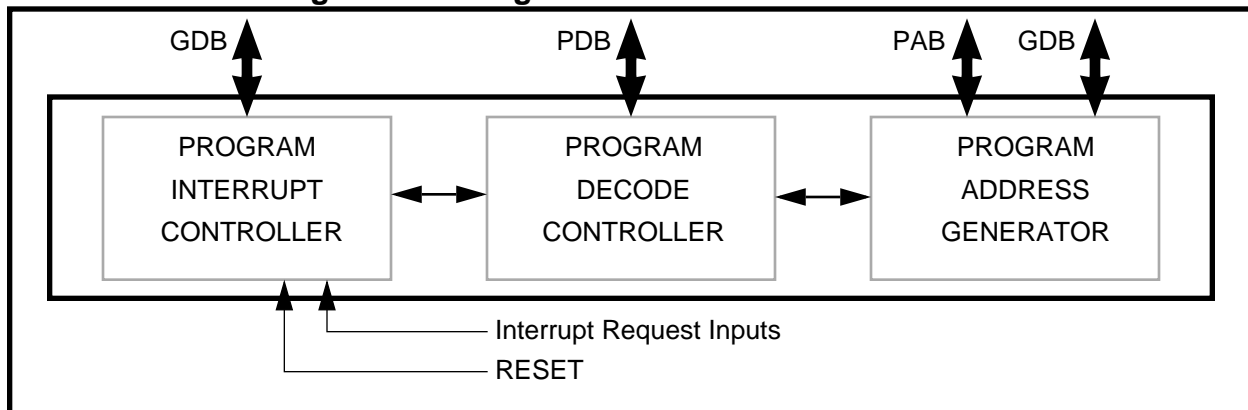
6.1 OVERVIEW

The program control unit performs instruction prefetch, instruction decoding, hardware DO loop control and exception processing. Its programmers model consists of eight read/write 24-bits registers, one read/write 5-bits register and a hardware system stack (SS). In addition to the standard program flow-control resources (e.g. interrupts, jumps), the program control unit support hardware DO loop and REPEAT mechanism.

The SS is a 16-level by 48-bit separate internal memory used to automatically store the PC and SR during subroutine calls and long interrupts. The SS automatically stores the LC and LA registers in addition to the PC and SR registers for hardware loops. All other data and control registers can be stored in the SS via software control. Each location in the SS is addressable as two 24-bit registers, system stack high (SSH) and system stack low (SSL), which are pointed to by the four LSBs of a 24-bit stack pointer (SP). The SS is extended in the data memory, in a space specified by the stack control registers that monitor the accesses to the SS. This hardware will copy the Least-Recently-Used location of the SS to data memory in case the on-chip hardware stack is full, and will bring data from data memory in case the on-chip hardware stack is empty.

The program control unit implements a seven-stage (prefetch-I, prefetch-II, decode, address gen-I, address gen-II, execute-I, execute-II) pipeline and controls the five processing states of the DSP56300 Core: normal, exception, reset, wait, and stop.

Figure 6-1. Program Control Unit Architecture



6.2 PROGRAM CONTROL UNIT ARCHITECTURE

The program control unit (Figure 6-1) consists of three hardware blocks:

Program Decode Controller (PDC), that decodes the 24-bit instruction loaded into the instruction latch and generates all signals necessary for pipeline control.

Program Address Generator (PAG), which contains all the hardware needed for program address generation, system stack and loop control.

Program Interrupt Controller (PIC) which arbitrates among all interrupt requests (internal and the five external: IRQA, IRQB, IRQC, IRQD and NMI) and generates the appropriate interrupt vector address.

6.2.1 Instruction Pipeline

The program control unit implements a seven-stage pipelined architecture in which concurrent stages of this pipeline occur. These seven stages consist of two prefetch stages, one decode stage, two address generation stages and two execute stages, as illustrated in Figure 6-2 and described in Table 6-1.

Although composed of many stages, the pipelined operation remains essentially hidden from the user, thus easing programmability. This is achieved by means of interlock hardware that is present in every execution unit of the processor. Due to this feature, programs written for the DSP56000/1/2 will execute correctly on the DSP56300 Core without any need for modification. Modification of the program may reduce the occurrence

of interlocks and improve execution speed.

Figure 6-2. Seven Stage Pipeline

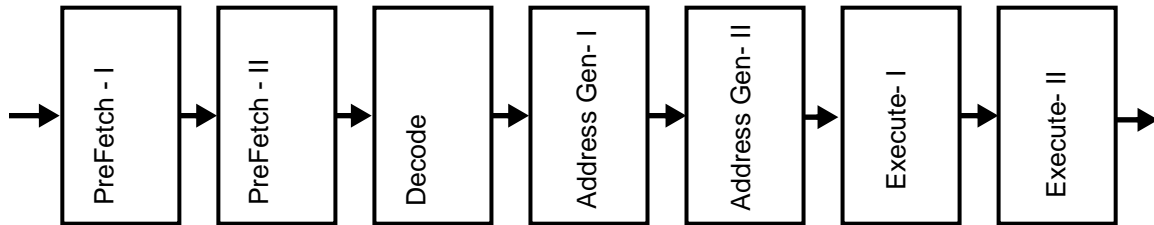


Table 6-1. Seven Stage Pipeline

Pipeline Stage	Description of Pipeline Stage
PreFetch-I	<ul style="list-style-type: none"> • Address generation for Program Fetch • Increment PC
PreFetch-II	<ul style="list-style-type: none"> • Instruction word read from memory
Decode	<ul style="list-style-type: none"> • Instruction Decode
Address Gen-I	<ul style="list-style-type: none"> • Address generation for Data Load/Store operations
Address Gen-II	<ul style="list-style-type: none"> • Address pointer update
Execute-I	<ul style="list-style-type: none"> • Read source operands to Multiplier and Adder • Read source register for memory store operations • Multiply • Write destination register for memory load operations
Execute-II	<ul style="list-style-type: none"> • Read source operands for Adder if written by previous ALU operation • Add • Write Adder results to the Adder destination operand • Write Multiplier results to the Multiplier destination operands

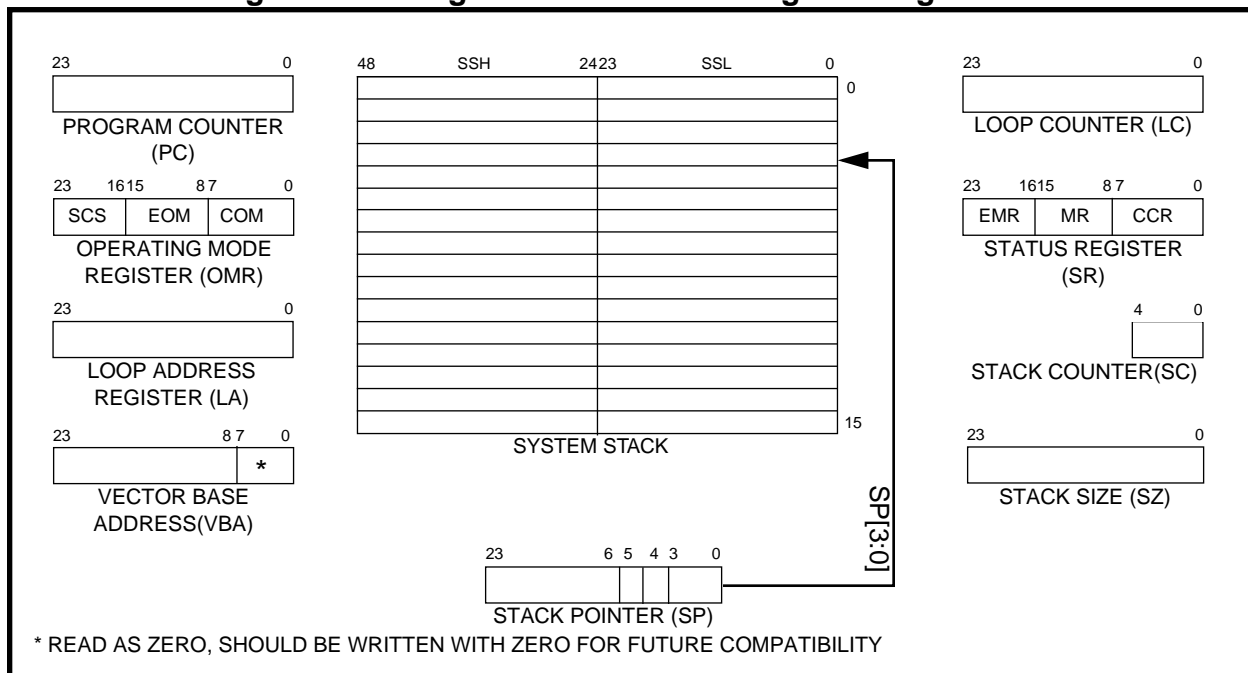
6.2.2 Clock Oscillator

The DSP56300 Core uses a two-phase clock for instruction execution; therefore, the clock runs at the same rate as the instruction execution. The clock can be provided by connecting an external crystal between XTAL and EXTAL, or by an external oscillator connected to EXTAL. The PLL can be used in order to determine the internal frequency related to the external.

6.3 PROGRAMMING MODEL

The program control unit features LA and LC registers dedicated to supporting the hardware DO loop instruction in addition to the standard program flow-control resources, such as a PC, SR, and SS. All registers are read/write to facilitate system debugging. Figure 6-3 shows the program control unit programming model with the registers and the SS. The following paragraphs give a detailed description of each register.

Figure 6-3. Program Control Unit Programming Model.



6.3.1 Program Counter (PC)

This is a special-purpose 24-bit address register that contains the address of instruction words in the program memory space. The PC can point to instructions, data operands, or addresses of operands. References to this register are always inherent and are implied by most instructions. The PC is stacked when hardware loops are initialized, when a JSR is performed, or when a long interrupt occurs. The PC is the source for the calculation of the real address in all position-independent instructions (like BRA).

6.3.2 Vector Base Address Register (VBA)

The VBA is a 24-bit register, 8 of them (bits 7-0) are read-only and are always cleared. The VBA is used as a base address of the interrupt vector and interrupt vector+1. When executing a fast or long interrupt, the vector address bits 7-0 are driven from the program interrupt control unit while bits 23-8 are driven from the VBA. The VBA register is a read/

write register that is referenced implicitly by interrupt processing or directly by the MOVEC instruction. The VBA is cleared during hardware reset.

6.3.3 Loop Counter Register (LC)

The LC register is a special read/write 24-bit counter used to specify the number of times a hardware program loop is to be repeated, in the range of 0 to $(2^{24}-1)$. This register is stacked into the SSL by a DO instruction and unstacked by end-of-loop processing or by execution of an ENDDO and BRKcc instructions. The LC is used also in the REP instruction, to specify the number of times the repeated instruction is to be repeated.

6.3.4 Loop Address Register (LA)

The contents of the 24-bit LA register indicate the location of the last instruction word in a hardware loop. This register is stacked into the SSH by a DO instruction and is unstacked by end-of-loop processing or by execution of an ENDDO and BRKcc instructions. The LA register, a read/write register, is written by a DO instruction and read by the SS when stacking the register.

6.3.5 System Stack (SS)

The System Stack (SS) is a separate 16x48-bit internal memory divided into two banks: System Stack High (SSH) and System Stack Low (SSL), 24 bits wide each. The SS is used for the following main tasks:

- Storing return address and status for subroutine calls.
- Storing LA, LC, PC and SR for the hardware DO loops.
- Storing calling routine variables for subroutine calls.

When a subroutine is called e.g. using the JSR instruction, the return address (PC) is automatically stored in the SSH and the chip status (SR) is automatically stored in the SSL.

When a return from subroutine is initiated by using the RTS instruction, the contents of the top location in the SSH is pulled and loaded into the PC and the SR is not affected. When a return is initiated using the RTI instruction, the contents of the top locations in the SS are pulled and loaded into the PC and SR (from SSH and SSL respectively).

The SS is also used to implement no-overhead nested hardware DO loops. When a hardware do-loop is initiated e.g. by using the DO instruction, the previous contents of the Loop Counter (LC) register is automatically stored in the SSL, the previous contents of the Loop Address (LA) register is automatically stored in the SSH and the Stack Pointer (SP) is incremented. The address of the loop's first instruction (PC) is also stored in the SSH and the chip status register (SR) is stored in the SSL.

The SS may be extended in the data memory by means of control hardware that monitors the accesses to the SS. This extension is enabled by Stack Extension Enable (SEN) bit in the chip Operating Mode Register (OMR). If this bit is cleared, the extension of the system stack is disabled and the amount of nesting is determined by the limited level of the hardware stack (15, one location is unusable when the stack extension is disabled). Up to 15 long interrupts, 7 DO loops, 15 JSRs, or combinations of these can be accommodated by the SS when its extension in data memory is disabled. When the SS limit is exceeded (either in the extended or in the non-extended mode), a nonmaskable stack error interrupt occurs.

By enabling the Stack extension, the limits on the level of nesting of subroutines or DO loops can be set to any desired value.

A stack extension algorithm is applied to all accesses to the stack:

- If an explicit (e.g. move to ssh) or implicit (e.g. jsr) push operation is performed, then the stack is examined by the stack extension control logic after that push has finished. If the on-chip hardware stack is full, then the least recently used word is moved into data memory to the location specified by the stack extension pointer (EP).
- If an explicit (e.g. move from ssh) or implicit (e.g. rts) pop operation is performed, then the stack is examined by the stack extension control logic after that pop has finished. If the on-chip hardware stack is empty, then the stack is loaded from the location (in data memory) specified by the stack extension pointer (EP).

6.3.6 Stack Extension Pointer (EP)

The contents of the 24-bit EP register is used to point to the stack extension in data memory whenever the stack extension is enabled and move operations to/from the on-chip hardware stack are needed. The EP register is located in the Address Generation Unit (AGU). For more details, please refer to Section 4.3.1.

6.3.7 Stack Size Register (SZ)

The 24-bit SZ register determines the number of data words allocated in memory for the stack in the extended mode. The extended stack overflow flag is generated when the value in SP equals the value in SZ. The stack size register is not initialized during hardware reset, and must be set (using a MOVEC instruction) prior to enabling the stack extension.

6.3.8 Stack Counter Register (SC)

The 5-bit SC register is used to monitor how many entries of the hardware stack are in use. The SC register is a read/write register and is referenced implicitly by some

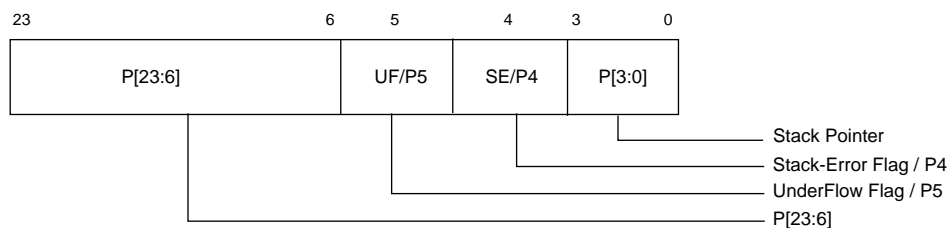
instructions (DO, JSR, RTI etc.) or directly by the MOVEC instruction. The stack counter register is cleared during hardware reset.

Note: During normal operation, the Stack Counter register should not be written. If a task switch is needed, writing a value greater than 14 or smaller than 2 will automatically activate the stack extension control hardware.
For proper operation, the SC should not be written with values greater than 16.

6.3.9 Stack Pointer Register (SP)

The 24-bit SP register indicates the location of the top of the SS. The status of the SS is also indicated in SP, when the extended mode is disabled (underflow, empty, full, and overflow). The SP register is referenced implicitly by some instructions (DO, JSR, RTI, etc.) or directly by the MOVEC instruction. The SP register format, shown in Figure 6-4, is described in the following paragraphs. The SP register is implemented as a 24-bit counter that addresses (selects) a 16-locations stack with its four LSBs. The possible SP values in the non-extended mode are shown in Table 6-2.

Figure 6-4. SP Register Format



6.3.9.1 Stack Pointer (Bits 0–3)

The SP points to the last used location on the SS. Immediately after hardware reset, these bits are cleared (SP=0), indicating that the SS is empty.

Data is pushed onto the SS by incrementing the SP, then writing data to the location pointed to by the SP. An item is pulled off the stack by copying it from the location pointed to by the SP and then decrementing SP.

6.3.9.2 Stack Error Flag / P4 bit (Bit 4)

This is a dual function bit. In the extended mode it acts as bit 4 of the Stack Pointer, as part of a 24-bit up/down counter. In the non-extended mode, it serves as the stack error (SE) flag that indicates that a stack error has occurred. The transition of the stack error flag from zero to one in the non-extended mode causes a priority level-3 stack error exception.

When the non-extended stack is completely full, the SP reads 001111, and any operation that pushes data onto the stack will cause a stack error exception to occur. The SP will read 010000 (or 010001 if an implied double push occurs).

Any implied pull operation with SP equal to zero will cause a stack error exception, and the SP will read \$0000FF (or \$0000FE if an implied double pull occurs). During such case, the stack error bit is set as shown in Table 6-2.

The stack error flag is a “sticky bit” which, once set, remains set until cleared by the user. The overflow/underflow bit remains latched until the first move to SP is executed.

Table 6-2. SP Register Values in the non-extended mode

UF	SE	P3	P2	P1	P0	Description
1	1	1	1	1	0	Stack Underflow condition after double pull
1	1	1	1	1	1	Stack Underflow condition
0	0	0	0	0	0	Stack Empty (RESET); Pull causes underflow
0	0	0	0	0	1	Stack Location 1
.	Stack Locations 2 - 13
0	0	1	1	1	0	Stack Location 14
0	0	1	1	1	1	Stack Location 15; Push causes overflow
0	1	0	0	0	0	Stack Overflow condition
0	1	0	0	0	1	Stack Overflow condition after double push

6.3.9.3 Underflow Flag / P5 bit (Bit 5)

This is a dual function bit. In the extended mode it acts as bit 5 of the Stack Pointer, as part of a 24-bit up/down counter. In the non-extended mode, the underflow flag is set when a stack underflow occurs. The stack underflow flag is a “sticky bit”, i.e. once the stack error flag is set, the underflow flag will not change state until explicitly written by a move instruction. The combination of “underflow=1” and “stack error=0” is an illegal combination and will not occur unless it is forced by the user. Also see the description for the stack error flag (Section 6.3.9.2) for additional information.

6.3.10 Status Register (SR)

This 24-bit register consists of an 8-bit condition code register (CCR), 8-bit mode register (MR) and an 8-bit extended mode register (EMR). The SR is stacked when program

looping is initialized, when a JSR is performed, or when interrupts occur (except for no-overhead fast interrupts). The SR format is shown in Figure 6-5.

Each of the SR bits are masked programmable. They can be programmed to one of the following configurations:

- read/write bit with the functionality as described in the following paragraphs
- read as zero bit

The two reserved bits are also outputs of the DSP56300 core, with derivative-dependent functionality. These outputs are also mask programmed to one of the following states:


- connected to the SR bit, reflecting its state
- connected to GND (forced to '0')

The CCR is a special-purpose control register that defines the results of previous arithmetic computations. The CCR bits are affected by data arithmetic logic unit (ALU) operations, parallel move operations, and by instructions that directly reference the CCR (ORI and ANDI) or instructions that specify SR as its destination (e.g. MOVEC). Parallel move operations only affect the S and L bits of the CCR. During processor reset all CCR bits are cleared.

The MR is a special-purpose control register defining the current system state of the processor. The MR bits are affected by processor reset, exception processing, DO, ENDDO (end current DO loop), RTI (return from interrupt) and TRAP instructions, and by instructions that directly reference the MR register - ANDI and ORI or instructions that specify SR as its destination (e.g. MOVEC). During processor reset the interrupt mask bits of the MR will be set while all the other bits will be cleared.

Figure 6-5. Status Register Format

EMR								MR								CCR								
23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
CP1:0				RM	SM	CE		SA	FV	LF	DM	SC		S1:0		I1:0	S	L	E	U	N	Z	V	C
CP1		- Core Priority Bit 1						LF		- DO-Loop Flag						S		- Scaling Bit						
CP0		- Core Priority Bit 0						DM		- Double Precision Multiply						L		- Limit						
RM		- Rounding Mode						SC		- Sixteen-Bit Compatibility						E		- Extension						
SM		- Arithmetic Saturation						S1		- Scaling Mode Bit 1						U		- Unnormalized						
CE		- Instruction Cache Enable						S0		- Scaling Mode Bit 0						N		- Negative						
SA		- Sixteenth-Bit Arithmetic						I1		- Interrupt Mask Bit 1						Z		- Zero						
FV		- DO-Forever Flag						I0		- Interrupt Mask Bit 0						V		- Overflow						
																C		- Carry						

 - Reserved bit. Read as zero, should be written with zero for future compatibility

The EMR is a special-purpose control register defining the current system state of the processor. The EMR bits are affected by processor reset, exception processing, DO FOREVER, ENDDO (end current DO loop), BRKcc, RTI (return from interrupt) and TRAP instructions, and by instructions that specify SR as its destination (e.g. MOVEC). During processor reset, all EMR bits will be cleared.

6.3.10.1 Carry (Bit 0)

The carry (C) bit is set if a carry is generated out of the MSB of the result in an addition operation. This bit is also set if a borrow is generated in a subtraction operation; Otherwise, this bit is cleared. The carry or borrow is generated from bit 55 of the result. The carry bit is also affected by bit manipulation, rotate, and shift instructions.

6.3.10.2 Overflow (Bit 1)

The overflow (V) bit is set if an arithmetic overflow occurs in the 56-bit result; Otherwise, this bit is cleared. This bit indicates that the result cannot be represented in the accumulator register; thus, the register has overflowed. In Arithmetic Saturation Mode, an arithmetic overflow occurs if the Data ALU result is not representable in the accumulator without the extension part, i.e. 48-bit accumulator (32-bit in Sixteen Bit Mode).

6.3.10.3 Zero (Bit 2)

The zero (Z) bit is set if the result equals zero. Otherwise, this bit is cleared.

6.3.10.4 Negative (Bit 3)

The negative (N) bit is set if the MSB of the result is set. Otherwise, this bit is cleared.

6.3.10.5 Unnormalized (Bit 4)

The unnormalized (U) bit is set if the two MSBs of the most significant (MSP) portion of the result are identical. Otherwise, this bit is cleared. The MSP portion of the A or B accumulators is defined by the scaling mode. The U bit is computed as described in Table 6-3.

6.3.10.6 Extension (Bit 5)

The extension (E) bit is cleared if all the bits of the integer portion of the 56-bit result are all ones or all zeros. Otherwise, this bit is set. The integer portion is defined by the scaling mode as described in Table 6-4. If the E bit is cleared, then the low-order fraction portion contains all the significant bits; the high-order integer portion is just sign extension. In this case, the accumulator extension register can be ignored. If the E bit is set, it indicates that the accumulator extension register is in use.

Table 6-3. Unnormalized Bit definition

S1	S0	Scaling Mode	U Bit Computation
0	0	No Scaling	$U = \overline{(\text{Bit } 47 \text{ xor Bit } 46)}$
0	1	Scale Down	$U = \overline{(\text{Bit } 48 \text{ xor Bit } 47)}$
1	0	Scale Up	$U = \overline{(\text{Bit } 46 \text{ xor Bit } 45)}$

Table 6-4. Extension Bit definition

S1	S0	Scaling Mode	Integer Portion
0	0	No Scaling	Bits 55,54.....48,47
0	1	Scale Down	Bits 55,54.....49,48
1	0	Scale Up	Bits 55,54.....47,46

6.3.10.7 Limit (Bit 6)

The limit (L) bit is set if the overflow bit is set or if the data shifter/limiter circuits perform a

limiting operation. In arithmetic Saturation Mode, the limit bit is also set when an arithmetic saturation occurs in the Data Alu result; otherwise, it is not affected. The L bit is cleared only by a processor reset or by an instruction that specifically clears it, which allows the L bit to be used as a latching overflow bit (i.e., a “sticky” bit). L is affected by data movement operations that read the A or B accumulator registers.

6.3.10.8 Scaling (Bit 7)

The Scaling bit (S) is set upon moving a result from accumulator A or B to the XDB or YDB buses (during an accumulator to memory or accumulator to register move) and will remain set until explicitly cleared, that is, the “S” bit is a “sticky” bit. The logical equations of this bit are dependent on the scaling mode. The scaling bit will be set if the absolute value in the accumulator, before scaling, was greater or equal to 0.25 or smaller than 0.75. This bit is cleared during hardware reset.

S0	S1	Scaling Mode	S equation
0	0	No scaling	$S = (A_{46} \text{ XOR } A_{45}) \text{ OR } (B_{46} \text{ XOR } B_{45}) \text{ OR } S \text{ (previous)}$
0	1	Scale down	$S = (A_{47} \text{ XOR } A_{46}) \text{ OR } (B_{47} \text{ XOR } B_{46}) \text{ OR } S \text{ (previous)}$
1	0	Scale up	$S = (A_{45} \text{ XOR } A_{44}) \text{ OR } (B_{45} \text{ XOR } B_{44}) \text{ OR } S \text{ (previous)}$
1	1	Reserved	S undefined

6.3.10.9 Interrupt Masks (Bits 8 and 9)

The interrupt mask bits, I1 and I0, reflect the current IPL of the processor and indicate the IPL needed for an interrupt source to interrupt the processor. The current IPL of the processor can be changed under software control. The interrupt mask bits are set during hardware reset but not during software reset.

I1	I0	Exceptions Permitted	EXceptions Masked
0	0	IPL 0, 1, 2, 3	None
0	1	IPL 1, 2, 3	IPL 0
1	0	IPL 2, 3	IPL 0, 1
1	1	IPL 3	IPL 0, 1, 2

6.3.10.10 Scaling Mode (Bits 10 and 11)

The scaling mode bits, S1 and S0, specify the scaling to be performed in the data ALU

shifter/limiter and the rounding position in the data ALU MAC unit. The shifter/limiter scaling mode affects data read from the A or B accumulator registers out to the XDB and YDB. Different scaling modes can be used with the same program code to allow dynamic scaling. One application of dynamic scaling is to facilitate block floating-point arithmetic. The scaling mode also affects the MAC rounding position to maintain proper rounding when different portions of the accumulator registers are read out to the XDB and YDB. The scaling mode bits, which are cleared at the start of a long interrupt service routine, are also cleared during a processor reset.

S1	S0	Rounding Bit	Scaling Mode
0	0	23	No Scaling
0	1	24	Scale down (1-Bit Arithmetic Right Shift)
1	0	22	Scale Up (1-Bit Arithmetic Left Shift)
1	1	-	Reserved for Future Expansion

6.3.10.11 Reserved SR Bit (Bit 12)

This bit is reserved for future expansion; It will read as zero during DSP56300 Core read operations and should be written with zero for future compatibility.

6.3.10.12 Sixteen-Bit Compatibility Mode (Bit 13)

The Sixteen-Bit Compatibility Mode (SC) enables full compatibility to object code written for the DSP56000 Family of Digital Signal Processors. When the SC bit is set, move operations to/from any of the AGU registers and to/from any of the PCU registers clear the 8 MSBits of the destination. The SC is cleared during processor reset.

6.3.10.13 Double Precision Multiply Mode (Bit 14)

The double precision multiply (DM) bit enables the operation of four multiply/multiply-accumulate operations, for the implementation of a double precision algorithm. This algorithm involves the multiplication of two 48-bit operands with a 96-bit result. The mode is disabled by clearing the DM bit.

Note: The Double Precision Multiply Mode is supported in order to maintain object code compatibility with the 56k Family of Digital Signal Processors. For a more efficient way of executing double precision multiply, please refer to Section 3.

While in Double Precision Multiply mode, the behavior of the four specific operations listed in the double precision algorithm is modified. Therefore these operations (with those

specific register combinations) should not be used, while in Double Precision Multiply mode, for any other purpose but for the double precision multiply algorithm. All other Data ALU operations (or the four listed operations but with other register combination) may be used.

The double precision multiply algorithm uses the Y0 register at all stages. Therefore Y0 should not be changed when running the double precision multiply algorithm. If the use of the Data ALU is required in an interrupt service routine, Y0 should be saved together with other Data ALU registers to be used, and should be restored before leaving the interrupt routine. The DM is cleared during a processor reset.

6.3.10.14 DO-Loop Flag (Bit 15)

The loop flag (LF) bit, set when a program loop is in progress, enables the detection of the end of a program loop. The LF is restored from stack when terminating a program loop. Stacking and restoring the LF when initiating and exiting a program loop, respectively, allow the nesting of program loops. At the start of a long interrupt service routine, the SR (including the LF) is pushed on the SS and the LF is cleared. When returning from the long interrupt with an RTI instruction, the SS is pulled and the LF is restored. The LF is cleared during a processor reset.

6.3.10.15 DO-Forever flag (Bit 16)

The DO-Forever flag (FV) bit is set when a DO FOREVER. The FV flag, like LF flag, is restored from stack when terminating a DO FOREVER program loop. Stacking and restoring the FV flag when initiating and exiting a DO FOREVER program loop, respectively, allow the nesting of program loops. At the start of a long interrupt service routine, the SR (including the FV) is pushed on the SS and the FV is cleared. When returning from the long interrupt with an RTI instruction, the SS is pulled and the FV is restored. The FV is cleared during a processor reset.

6.3.10.16 Sixteen-Bit Arithmetic Mode (Bit 17)

The Sixteen-Bit Arithmetic Mode (SA) when set, enables the sixteen bit arithmetic mode of operation. In this mode the rounding of the arithmetic operation will be performed on bit 15 of the accumulator A1/B1 instead of the usual bit 23 of A0/B0. The scaling, as well as the shifting/limiting operation of the Data-ALU will be affected accordingly. The SA bit is cleared during processor reset. At the start of a long interrupt service routine, the SR (including the SA) is pushed on the SS and the SA is cleared. The SA is cleared during a processor reset.

6.3.10.17 Reserved SR Bit (Bit 18)

This bit is reserved for future expansion; It will read as zero during DSP56300 Core read operations and should be written with zero for future compatibility.

6.3.10.18 Cache Enable (Bit 19)

The Cache Enable (CE) bit is used to enable or to disable the operation of the instruction cache controller. If the bit is set, the cache is enabled, instructions are cached into the internal PRAM and fetch from there. If the bit is cleared, the cache is disabled and the DSP56300 Core will fetch instructions from external or internal program memory, according to the memory space table of the specific DSP56300 Core-based chip. This bit is cleared during a processor reset.

NOTE To guarantee proper operation cache enable mode (CE bit in SR) should not be cleared while burst mode is enabled (BE bit in OMR is set).

6.3.10.19 Arithmetic Saturation Mode (Bit 20)

The Arithmetic Saturation Mode (SM) bit, when set, selects automatic saturation on 48 bits for the results going to the accumulator. This saturation is done by a special circuit inside the MAC unit. The purpose of this bit is to provide an arithmetic saturation mode for algorithms which do not recognize or cannot take advantage of the extension accumulator. This bit is cleared during processor reset.

6.3.10.20 Rounding Mode (Bit 21)

The Rounding Mode (RM) bit selects the type of rounding performed by the DATA ALU during arithmetic operations. If the bit is cleared, convergent rounding is selected. If the bit is set, two's complement rounding is selected. At the start of a long interrupt service routine, the SR (including the RM) is pushed on the SS and the RM is cleared. The RM bit is cleared during processor reset.

6.3.10.21 Core Priority (Bits 22 and 23)

Under the control of CDP1:0 bits in the Operating Mode Register (OMR), see Section 6.3.11.6, the CORE priority bits, CP1 and CP0, specify the priority of CORE accesses to external memory. These bits are compared against the priority bits of the active DMA

OMR - CDP1:0	CP1:0	Core Priority
00	00	0 (lowest)
00	01	1
00	10	2
00	11	3 (highest)
01	xx	see Section 6.3.11.6
10	xx	see Section 6.3.11.6
11	xx	see Section 6.3.11.6

channel. If CORE priority is greater than DMA priority, the DMA will wait for a free access slot in the external bus. If CORE priority is less than DMA priority, the CORE will wait for a free access slot in the external bus. If CORE priority equals to the DMA priority, CORE and DMA will access the external bus in a linear fix pattern. The Core priority bits are set during processor reset.

6.3.11 Operating Mode Register

The OMR is a 24-bit register, partitioned into three bytes. The least significant byte of OMR (bits 7-0) is the Chip Operating Mode byte (COM) which is used to determine the operating mode of the chip. This byte is only affected by processor reset and by instructions directly referencing the OMR: ANDI, ORI or other instructions that specify OMR as a destination (e.g. MOVEC). During processor reset, the chip operating mode bits (MD, MC, MB and MA) will be loaded from the external mode select pins MODD, MODC, MODB and MODA respectively.

Each of the OMR bits are masked programmable. They can be programmed to one of the following configurations:

- read/write bit with the functionality as described in the following paragraphs
- read as zero bit

Some of the reserved bits, as described later, are also outputs of the DSP56300 core, with derivative-dependent functionality. These outputs are also mask programmed to one of the following states:


- connected to the SR bit, reflecting its state
- connected to GND (forced to '0')

The middle part of OMR (bits 15-8) is the Extended Chip Operating Mode byte (EOM) which is used to determine the operating mode of the chip. This byte is only affected by processor reset and by instructions directly referencing the OMR: ANDI, ORI or other instructions that specify OMR as a destination (e.g. MOVEC).

The most significant byte of OMR (bits 23-16) is the System Stack Control Status byte (SCS) which is used to control and monitor the Stack extension in the data memory. The SCS byte is referenced implicitly by some instructions (DO, JSR, RTI, etc.) or directly by the MOVEC instruction.

Figure 6-6. Operating Mode Register (OMR) Format

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
			SEN	WRP	EOV	EUN	XYS				BRT	TAS	BE	CDP1:0		MS	SD		EBD	MD	MC	MB	MA
SEN - Stack Extension Enable								BRT - Bus Release Timing								MS - Memory Switch Mode							
WRP - Extended Stack Wrap Flag								TAS - TA Synchronize Select								SD - Stop Delay							
EOV - Extended Stack Overflow Flag								BE - Burst Mode Enable								EBD - External Bus Disable							
EUN - Extended Stack Underflow Flag								CDP1 - Core-Dma Priority 1								MD - Operating Mode D							
XYS - Stack Extension Space Select								CDP0 - Core-Dma Priority 0								MC - Operating Mode C							
																MB - Operating Mode B							
																MA - Operating Mode A							

 - Reserved bit. Read as zero, should be written with zero for future compatibility

6.3.11.1 Chip Operating Mode (Bits 0, 1,2 and 3)

The chip operating mode bits MD,MC, MB and MA, indicate the operating mode of the DSP56300 Core. On processor reset, these bits are loaded from the external mode select pins, MODD, MODC, MODB and MODA, respectively. After the DSP56300 Core leaves the reset state, MD, MC, MB and MA can be changed under program control.

6.3.11.2 External Bus Disable (Bit 4)

The External Bus Disable (EBD) bit is use to disable the external bus controller, in order to reduce the power consumption when external memories are not used. When EBD bit is set the external bus controller is disabled and external memory should not be accessed. When the EBD bit is cleared the external bus controller is enabled and external access may be performed. The EBD bit is cleared on hardware reset. For a detailed description of the EBD and it's applications please refer to Chapter 2.

6.3.11.3 Reserved COM Bit (Bit 5)

This bit is reserved for future expansion; it will read as zero during DSP56300 Core read operations and should be written with zero for future compatibility. The bit is also output of the DSP56300 core, with derivative-dependent functionality. The output is mask programmed to one of the states mentioned in previous paragraph.

6.3.11.4 Stop Delay (Bit 6)

The STOP instruction causes the DSP56300 Core to indefinitely suspend processing in the middle of the STOP instruction. When exiting the stop state, if the stop delay bit is cleared, a 128K clock cycle delay is selected before continuing the stop instruction cycle. However, if the stop delay bit is set, the delay before continuing the instruction cycle is 16 clock cycles. The long delay allows a clock stabilization period for the internal clock to begin oscillating and to stabilize. When a stable external clock is used, the shorter delay allows faster start-up of the DSP56300 Core. The Stop Delay bit is cleared during processor reset.

6.3.11.5 Memory Switch (Bit 7)

The Memory Switch (MS) Mode bit is used to turn on the memory space switch mode in which some addresses of the internal data memory (X, Y or both) become part of the chip internal program RAM. This bit is cleared during processor reset.

NOTE 1 Program data placed in PRAM/I-Cache area changes its placement after the setting of MS bit, such as it always occupies the top most internal PRAM addresses.

NOTE 2 To assure proper operation, six NOP instructions should be placed after instruction that changes MS bit.

NOTE 3 To assure proper operation, MS bit should not be set while I-Cache is enabled (CE bit is set in SR).

6.3.11.6 Core-Dma Priority Bits (Bits 9 and 8)

The Core-Dma Priority (CDP1, CDP0) bits specify the priority between the Core accesses and DMA accesses to external bus.

CDP1:0	Core-Dma Priority
00	determined by comparing CP1:0 with the active DMA channel priority
01	DMA accesses have higher priority than CORE accesses
10	DMA accesses have the same priority as the CORE accesses
11	DMA accesses have lower priority than the CORE accesses

These bits are set during processor reset.

6.3.11.7 Burst Mode Enable (Bit 10)

The burst mode enable (BE) bit is used to enable or to disable the burst mode in the memory expansion port during instruction cache miss. If the bit is cleared, the burst mode is disabled and only one program word will be fetched from the external memory when an instruction cache miss condition is detected. If the bit is set, the burst mode is enabled, and up to four program words will be fetched from the external memory when an instruction cache miss is detected. For detailed description of the Burst Mode, refer to Chapter 5. This bit is cleared by hardware reset.

6.3.11.8 TA Synchronize Select (Bit 11)

The TA synchronize select (TAS) bit is used to select the synchronization method for the input Port A pin - \overline{TA} (Transfer Acknowledge). If TAS is cleared, the user is responsible to assert the TA pin synchronously to the chip clock, as described in the detailed data sheet. If TAS is set, the \overline{TA} input pin is synchronized inside the chip, thus eliminating the need for an off-chip synchronizer. The user must negate TA pin synchronously to the chip clock, independently of the TAS bit value. See Section 2.2.4 for more details. The TAS bit is cleared on hardware reset.

6.3.11.9 Bus Release Timing (Bit 12)

The Bus Release Timing (BRT) bit is used to select between fast or slow bus release. If BRT is cleared, a fast bus release mode is selected (i.e. no additional cycles are added to the access and BB# is not guaranteed to be the last port A pin that is tri-stated at the end of the access). If BRT is set, a slow bus release mode is selected (i.e. additional one cycle is added to the access, and BB# is the last port A pin that is tri-stated at the end of the access). The BRT bit is cleared on hardware reset. For a detailed description of the Bus Release Modes and their applications please refer to Chapter 2.

6.3.11.10 Reserved EOM Bits (Bits 15, 14 and 13)

These bits are reserved for future expansion; they will read as zero during DSP56300 Core read operations and should be written with zero for future compatibility. These bits are also outputs of the DSP56300 core, with derivative-dependent functionality. The outputs are mask programmed to one of the states mentioned in previous paragraph.

6.3.11.11 XY Select for Stack extension (Bit 16)

The XY Select bit for the Stack extension determines if the extension will be mapped onto the X memory space or onto the Y memory space. If the bit is clear, then the stack extension is mapped onto the X memory space. If it is set, the stack extension is mapped to the Y memory space. This bit is cleared by hardware reset.

6.3.11.12 Extended Stack Underflow Flag (Bit 17)

The extended stack underflow (EUN) flag is set when a stack underflow occurs in the stack extended mode. Extended stack underflow is recognized when a pull operation is requested when SP equals to 0, and the extended mode is enabled by the EN bit. The extended stack underflow flag is a “sticky bit” i.e. the only way to clear this bit is by hardware reset or by an explicit move operation to the OMR. The transition of the extended stack underflow flag from zero to one causes a priority level-3 stack error exception. The extended stack underflow flag is cleared by hardware reset.

6.3.11.13 Extended Stack Overflow Flag (Bit 18)

The extended stack overflow (EOV) flag is set when a stack overflow occurs in the stack extended mode. Extended stack overflow is recognized when a push operation is requested while SP equals to SZ (Stack Size register), and the extended mode is enabled by the EN bit. The extended stack overflow flag is a “sticky bit” i.e., the only way to clear this bit is by hardware reset or by an explicit move operation to the OMR. The transition of the extended stack overflow flag from zero to one causes a priority level-3 stack error exception. The extended stack overflow flag is cleared by hardware reset.

6.3.11.14 Extended Stack Wrap Flag (Bit 19)

The extended stack wrap (WR) flag is set when it is first recognized that a copy from the on-chip hardware stack to the stack extension memory is needed. This flag may be used during the debugging phase of the software as means of evaluating and increasing the speed of the software implemented algorithms. The extended stack wrap flag is a “sticky bit” i.e., the only way to clear this bit is by hardware reset or by an explicit move operation to the OMR. The extended stack wrap flag is cleared by hardware reset.

6.3.11.15 Extended Stack Enable (Bit 20)

The extended stack enable (EN) bit is used to enable or to disable the stack extension in the data memory. If the EN bit is set, the extension is enabled. This bit is cleared by hardware reset, thus disabling the stack extension as default.

6.3.11.16 Reserved SCS Bits (Bits 21-23)

These OMR bits, reserved for future expansion, will read as zero during DSP56300 Core read operations, and should be written with zero for future compatibility. These bits are not outputs of the DSP56300 core.

6.4 SIXTEEN-BIT COMPATIBILITY MODE

When the SIXTEEN-BIT COMPATIBILITY mode bit (SC, see Figure 6-5 on page 6-10) is set, move operations to/from any of the following PCU registers clear the 8 MSBits of the destination: LA, LC, SP, SSL, SSH, EP, SZ, VBA and SC. This guarantees compatibility for object code written for the DSP56000 Family of Digital Signal Processors.

If the source is either SR or OMR, then the 8 MSBits of the destination will also be cleared. If the destination is either SR or OMR, then the 8 MSBits of the destination will be left unchanged.

In order to change the value of one of the 8 MSBits of SR or OMR, the SIXTEEN-BIT COMPATIBILITY mode bit (SC) should be cleared.

The LOOP count is also affected by the SIXTEEN-BIT COMPATIBILITY mode bit. If it is cleared (normal operation), then loop count value of 0 will cause the loop body to be skipped, and loop count value of 0xFFFFF will cause execution of the loop the maximum number of $2^{24}-1$ times. If the bit is set, the loop count value of 0 will cause the loop to be executed 2^{16} times, and loop count value of 0x00FFFF will cause execution of the loop the maximum number of $2^{16}-1$ times.

Due to pipelining, a change in the SC bit takes affect only after three instruction cycles. Inserting three NOP instructions after the instruction that changes the value of this bit will ensure proper operation.
