IACs

11

CHAPTER 11
IACs

This chapter describes the intra-agent communication (IAC) mechanism of the 80960SA/SB processor. Included is a description of the IAC-message structure, the IAC-message sending and receiving mechanism, and reference information on the available IAC messages.

NOTE

The 80960SA/SB processor's intra-agent communication mechanism is an extension to the i960 architecture and may not be supported in other processors based on this architecture.

INTRODUCTION TO IAC MESSAGES

Programs running on the 80960SA/SB processor can use this message-passing mechanism to send messages internally to the processor.

The primary function of these facilities is to provide an alternative to the external interrupt mechanism to communicate with the processor. Also, certain processor functions like re-initialization, purging the instruction cache, and setting breakpoint registers can only be carried out with this mechanism.

IAC messages (referred to here as IACs) are four words in length and are exchanged by means of message buffers that are mapped to memory. All the usable IACs are predefined. The processor handles an IAC in much the same way as it handles an instruction.

IAC MESSAGE FORMAT

Figure 11-1 shows the format for an IAC message. Each message is four-words in length and consists of a message-type field and up to five parameter fields.

The message type is an 8-bit binary code. Each IAC has a unique message type.

The parameters can be 8, 16, or 32-bits in length, depending on the specified field. Many of the IACs do not require parameters. When a message type does require one or more parameters, the processor only looks at the required parameter fields. Those fields not used are ignored.

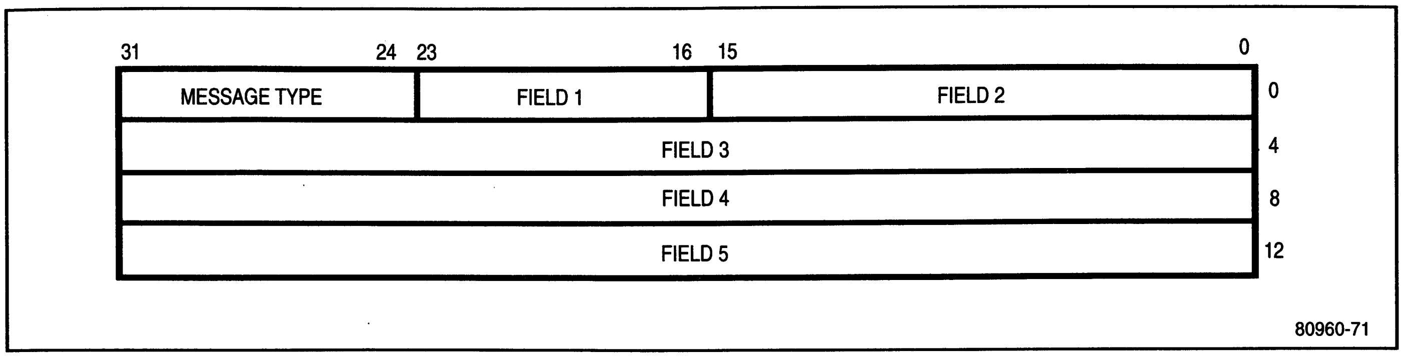

Figure 11-1: IAC Message Format

SOFTWARE REQUIREMENTS FOR HANDLING IACs

No special software, such as dedicated data structures or stacks, are required to handle IACs. An IAC is sent with a quad synchronous move instruction (**synmovq**). When the processor receives an IAC, it handles it independently from the program execution environment. It does not use the instruction execution unit, the registers (global or local), the stack, or memory. Thus, the state of the processor when the IAC is received does not need to be saved.

Some IACs, such as the purge instruction cache IAC, do not affect the processor's state. The processor treats these IACs as if they were an instruction inserted in the control flow of the process. When the IAC action is complete, the processor resumes work on the program it is currently running.

Other IACs, such as the re-initialize processor IAC, cause the state of the processor or the control of the currently running program to be permanently changed. In these instances, the processor resumes activity in its new processor state, following the execution of the IAC.

All IACs are assumed to have a priority of 31, so the processor executes the action requested in the IAC message immediately, even if the processor's current priority is 31. While the processor is handling an IAC, it will not respond to interrupts signaled on the interrupt pins.

IACs are used for functions such as setting breakpoint registers, purging the instruction cache, or sending software initiated interrupts.

To send an IAC, software must perform the following steps:

1. Load the message into four consecutive words in memory, with the first word aligned on a quad-word boundary.
2. Execute a **synmovq** instruction to move the message from its source address to destination address $FF000010_{16}$.

When the destination operand of a **synmovq** instruction is $FF000010_{16}$, the processor interprets the instruction as a send internal-IAC instruction. The processor then receives the IAC by moving the message from memory into an internal message buffer.

The action of the **synmovq** move instruction insures that the loading of the message into the processor is completed before the processor is allowed to perform any other chores.

NOTE

The addresses in the range $FF000000_{16}$ to $FFFFFFFF_{16}$ are reserved for implementation-defined functions.

SUMMARY OF IAC MESSAGES

Table 11-1 gives a list of the IAC messages that the processor can send. The following section provides detailed reference information on these messages.

Table 11-1: IAC Messages

Interrupt Handling	Processor Management
Interrupt	Purge Instruction Cache
Test Pending Interrupt	Set Breakpoint Register Store System Base Reinitialize Processor

IAC MESSAGE REFERENCE

The following section provides detailed descriptions of the operations carried out for each of the IACs. This section is organized alphabetically by IAC title for easy reference.

Interrupt

Message Type:	40_{16}	
Parameters:	Field 1	Interrupt vector
	Fields 2 - 5	Not Used

Function:

Generates an interrupt request. The interrupt vector is given in field 1 of the IAC message. The processor handles the interrupt request just as it does interrupts received from other sources. If the interrupt priority is higher than the processor's current priority, the processor services the interrupt request immediately. Otherwise, it posts the interrupt in the pending interrupts section of the interrupt table.

Refer to Chapter 5 for further information on the servicing of interrupt IACs.

Purge Instruction Cache

Message Type: 89_{16}

Function: Invalidates all entries in the processor's internal instruction cache.

Reinitialize Processor

Message Type:	93_{16}	
Parameters:	Fields 1 - 2	Not Used
	Field-3	Address of System Address Table
	Field-4	Address of Processor Control Block
	Field 5	Start Instruction IP

Function:

Reestablishes the processor state. In re-initializing itself, the processor first locates the system address table and the processor control block in the IMI from the addresses given in fields 3 and 4.

The processor then begins executing the instruction list beginning with the IP given in field 5.

Set Breakpoint Register

Message Type:	$8F_{16}$	
Parameters:	Fields 1 - 2	Not Used
	Field 3	Breakpoint IP
	Field 4	Breakpoint IP
	Field 5	Not Used

Function:

Enables or disables two breakpoints. When the processor receives this IAC, it conditionally loads the parameters from fields 3 and 4 into breakpoint registers 0 and 1, respectively. Field 3 provides a breakpoint IP for breakpoint register 0, and field 4 provides a breakpoint IP for breakpoint register 1. Bit 1 in each of these fields is a breakpoint disable flag.

If the disable flag in one of these fields is set, the breakpoint for the corresponding breakpoint register is disabled. Otherwise, the IP value in the field is loaded into the corresponding breakpoint register and the breakpoint is enabled.

Breakpoints are described in the section in Chapter 7 titled "Breakpoint-Trace."

Store System Base

Message Type:	80_{16}	
Parameters:	Fields 1 - 2	Not Used
	Field 3	Destination Address
	Fields 4 - 5	Not Used

Function: Stores the current locations of the system address table and the PRCB in a specified location in memory. The address of the system address table is stored in the word starting at the byte specified in field 3, and the address of the PRCB is stored in the next word in memory (field 3 address plus 4).

Test Pending Interrupts

Message Type:	41_{16}
Function:	Tests for pending interrupts. The processor checks the pending interrupt section of the interrupt table for a pending interrupt with a priority higher than the processor's current priority. If a higher priority interrupt is found, it is serviced immediately. Otherwise, no action is taken.