IBM

Series/1

GA34-0851-0

IBM Series/1
4956 Processor Model K00
Description

IBM

Series/1

GA34-0851-0

IBM Series/1
4956 Processor Model K00
Description

First Edition (February 1987)

Use this publication for the purpose stated in the Preface.

Changes are periodically made to the information herein; any such changes will be reported in subsequent revisions or Technical Newsletters.

It is possible that this material may contain reference to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such IBM products, programming, or service in your country.

Publications are not stocked at the address given below. Request for copies of IBM publications should be made to your IBM represenative or the IBM branch office serving your locality.

This publication could contain technical inaccuracies or typographical errors. A form for readers' comments is provided at the back of this publication. If the form has been removed, address your comments to IBM Corporation, Information Development, Department 27U, Internal Zip 5408, P. O. Box 1328, Boca Raton, Florida 33429-1328. IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Preface

This publication describes the unique functional characteristics and the optional features of the IBM Series/1 4956 Processor Model K00. This publication also provides reference information about the possible configurations and feature operations of the processor. Refer to the *IBM Series/1 Principles of Operation*, GA34-0152, for the common Series/1 processor functional characteristics and instructions.

This publication is intended primarily as a reference manual for experienced programmers who require machine code information to plan, correct, and modify programs written in the assembler language. The reader should understand data processing terminology and be familiar with binary and hexadecimal numbering systems.

Chapter 1, "Introduction," contains a general description of the processor, processor storage, and processor features. This chapter also contains information about changes to the programming instruction set.

Chapter 2, "Processor Storage Addressing Using the Relocation Translator," describes the relocation translator, including:

- Relocation addressing
- Storage-protection mechanism
- Error-recovery considerations.

Chapter 3, "Console," describes the keys, switches, and indicators for the basic console and the optional programmer console. Typical manual operations, such as storing into and displaying processor storage, are presented.

Chapter 4, "Diagnose (DIAG) Instruction," describes the Diagnose instruction.

Appendix A, "Instruction Execution Times," contains information for determining instruction execution times and instruction throughput.

Appendix B, "Software Notes," lists some software notes for the processor.

Prerequisite Publication

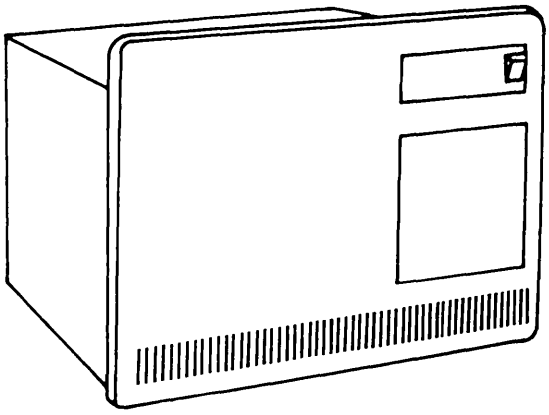
For a description of the processor architecture and a detailed description of the instruction set for the IBM Series/1 processors, refer to the *IBM Series/1 Principles of Operation*, GA34-0152.

Contents

Chapter 1. Introduction

The IBM Series/1 4956 Processor Model K00 is a compact general-purpose computer. The model K00 has 2 megabytes of basic storage.

The processor is microcode-controlled for both automatic functions and program instruction functions. It occupies the full width of a standard 483-millimeter (19-inch) rack. It contains 14 card sockets for data channel features and channel repower cards.



The processor has the following characteristics:

- Four priority interrupt levels, with independent registers and status indicators for each level.
- Automatic and program-controlled level switching.
- An instruction set that includes stacking and linking facilities, multiply and divide, variable-field-length byte operations, and a variety of arithmetic and branching instructions.
- Supervisor and problem states.
- A basic console that is a standard feature; a programmer console that is an optional feature.
- Basic main storage of 2 megabytes.
- A storage address relocation translator that allows addressing of main storage larger than 64 kilobytes.
- Program-controlled mode switching that allows the user to specify the maximum direct mappable storage range:
  - 512K bytes in 3-bit mode
  - 1024K bytes (1MB) in 4-bit mode
- Prefetching instruction stream.
- An error correction code (ECC) that is implemented in storage to provide the capability for single-bit error correction and double-bit error detection.
- Sixteen 64-bit hardware floating-point registers. For a detailed description of this feature, refer to the *IBM Series/1 Principles of Operation*, GA34-0152.
- A clock/comparator with four instructions provided to set or copy the clock and comparator.

- Channel capability as follows:
  - Asynchronous, multidropped channel
  - 256 input/output (I/O) devices can be addressed
  - Direct program control and cycle-steal operations
  - Maximum burst output data rate of 2.08 million bytes per second
  - Maximum burst input data rate of 3.03 million bytes per second
  - Maximum burst aggregate data rate (assuming 30% Read and 70% Write) of 2.74 million bytes per second.

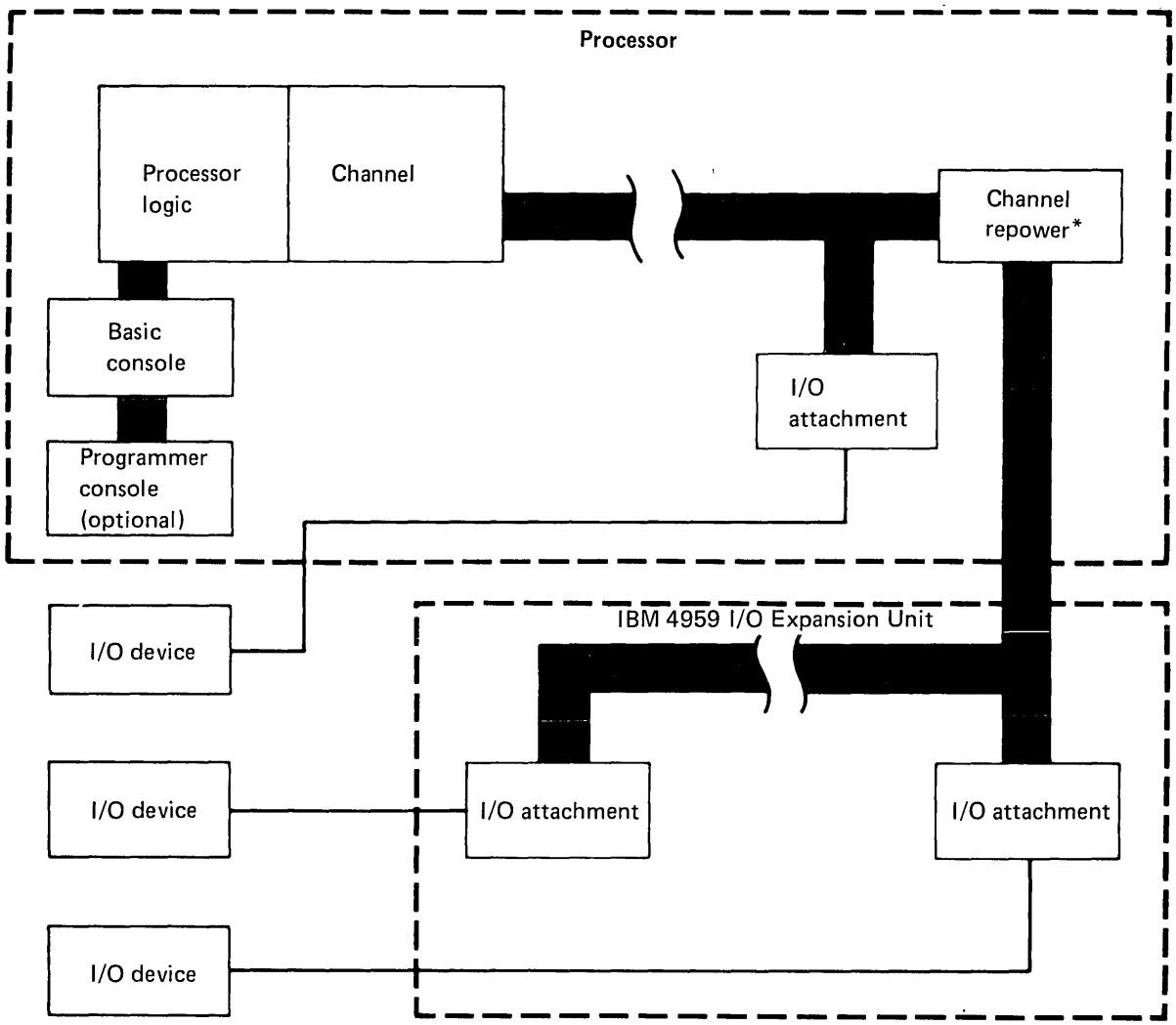
Note: The burst output and burst input (16-bit) data rates are reduced from the values shown by data channel attachment characteristics, channel loading during instruction processing, channel repowering, and processor storage refresh requirements.

- Faster throughput.
- A redefined 16-bit address key register (AKR).
- A redefined 16-bit processor status word (PSW) register.
- Several instructions modified for extended registers (ER) architecture:
  - Set Address Key Register (SEAKR)
  - Copy Address Key Register (CPAKR)
  - Set Segmentation Registers (SESR)
  - Copy Segmentation Registers (CPSR)
  - Enable (EN)
  - Disable (DIS)
  - Copy Processor Status and Reset (CPPSR).

For an explanation of the standard Series/1 instruction set, refer to the *IBM Series/1 Principles of Operation*, GA34-0152.

Processor Description

The basic processor includes the processor card with 2 megabytes of storage and a basic console. Figure 1-1 shows a block diagram of the processor and an IBM Series/1 4959 Input/Output Expansion Unit.



*Required with an expansion unit.

Figure 1-1. IBM 4956 Processor Model K00 and an IBM 4959 Expansion Unit

Four priority interrupt levels (0–3) are implemented in the processor. Each level has an independent set of machine registers. Level switching can occur in two ways: (1) by program control, or (2) automatically upon acceptance of an I/O interrupt request. The interrupt mechanism provides 256 unique entry points for I/O devices.

Note: A Prepare command to levels 4–15 is executed so that condition code reporting occurs; however, the Prepare command is not executed at the addressed device and effectively results in a no-operation.

The processor instruction set contains a variety of instruction types. These include:

- Bit manipulation
- Shift
- Branch
- Register immediate
- Storage immediate
- Register to register
- Register to storage
- Storage to register
- Storage to storage
- System register to register
- System register to storage
- Multiple register to storage
- Variable byte field
- Input/output.

Supervisor and problem states are implemented, with appropriate privileged instructions for the supervisor.

The basic console is intended for dedicated systems that are used in a primarily unattended environment. Only minimal controls are provided. A programmer console, which can be added as a feature, provides a variety of indicators and controls for operator-oriented systems.

An error correction code (ECC) is implemented in storage. ECC consists of single-bit error correction and double-bit error detection. ECC provides the user with a higher system availability.

Note: When a double-bit error in storage is detected during a processor read, a machine check interrupt occurs with PSW bit 8 set to 1 (storage parity error).

There is no storage-protect feature in the processor. However, there is a read-only protect capability provided by the address translator when it is enabled.

Note: The Set Storage Key (SESK) and Copy Storage Key (CPSK) instructions are not supported by this processor. Execution of SESK or CPSK results in a program check: Invalid Function, Invalid Storage Address, or Specification Check.

I/O devices are attached to the processor through the processor data channel. The data channel directs the flow of information between the I/O devices, the processor, and main storage. The data channel supports a maximum of 256 addressable devices.

The data channel supports:

- **Direct program control operations.** Each Operate I/O instruction transfers a byte or word of data between main storage and the device. The operation may or may not terminate in an interrupt.

- **Cycle-steal operations.** Each Operate I/O instruction initiates multiple data transfers between main storage and the device. The maximum aggregate cycle-steal rate is 1.37 million transfers per second. Cycle-steal operations are overlapped with processor operations and always terminate in an interrupt.

- **Interrupt servicing.** Interrupt requests from the devices, along with cycle-steal requests, are presented and polled concurrently with data transfers.

Card Plugging Assignments

The processor unit contains power and space for additional features. The IBM Series/1 4959 Input/Output Expansion Unit and the IBM Series/1 4965 Storage and I/O Expansion Unit are available for adding additional features, if desired. Figure 1-2 shows the card plugging assignments for the processor.

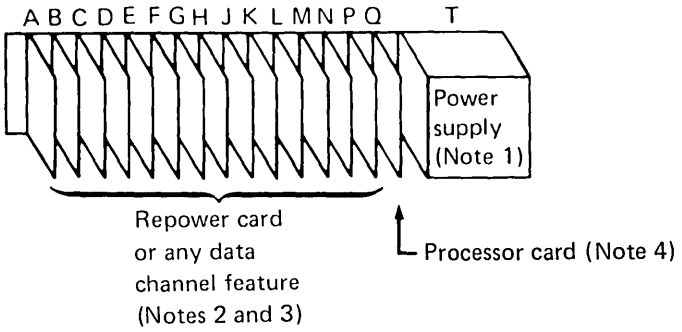


Figure 1-2. IBM Series/1 4956 Model K00 Plugging Assignments

Notes:

1. *The pluggable high-frequency power supply plugs into card socket T.*
2. *If a channel repower card is used, it must be plugged to the left of and adjacent to the leftmost I/O card installed.*
3. *A maximum of five serially-connected channel repower features can be driven by each processor. Any processor system that includes an IBM I/O expansion unit with the two-channel switch feature is limited to three channel repower features.*
4. *The processor card in socket Q contains 2 megabytes of basic storage.*

I/O Units and I/O Features

A variety of I/O units and features are available for use with the processor. For a list and description of system units and features, refer to the *IBM Series/1 System Selection Guide*, GA34-0143, and the *IBM Series/1 Digest*, G360-0061. Detailed information about I/O units and features can be found in separate publications.

Extended Registers Architecture

The extended registers (ER) architecture extends the definition of various system registers. These include AKR K-field definitions, the address keys, and the number of segmentation registers provided in the relocation and I/O translators. These extensions allow for the expansion of physical storage in excess of 512KB by providing more keys for the active address space key. These extensions provide for another set of segmentation registers called the I/O translator to be dedicated to I/O.

The 3-bit active address space key has been extended to 4-bit keys without extending the physical length of any of the existing 16-bit registers. The 4-bit keys allows up to 16 address spaces and 512 segmentation registers to map up to 1MB of physical storage.

Two sets of segmentation registers (translators) are provided: one set is for the processor; one set is dedicated to I/O. The processor has 512 segmentation registers with a 16-bit AKR to map the allowable physical storage. The I/O translator has 256 segmentation registers. When the I/O translator has been enabled, the processor uses 3-bit keys from I/O and the five high-order bits of the logical address to select one of the I/O translator registers. If the I/O translator is not enabled, then the 3-bit key with the five high-order bits of logical address is used to select a register from the processor translator.

Address Key Register (AKR)

The extension of the AKR bits allows the low-order bits to remain physically the same, but changes its bit-position name. For extended registers, the high-order bit is connected to the existing 3-bit definition; however, the high-order bit will always be known as bit 0. Thus, the lower-order bits in the key registers are renumbered.

The address key register (AKR) contains 16 bits. It is formatted as follows:

Bit	*3-bit mode*	*4-bit mode*
0	Eq Op Spaces (EOS)	EOS
1	Reserved	Reserved
2	Reserved	Reserved
3	Reserved	Reserved
4	Reserved (ER 4-bit keys)	OP1K bit 0
5	OP1K bit 0	OP1K bit 1
6	OP1K bit 1	OP1K bit 2
7	OP1K bit 2	OP1K bit 3
8	Reserved (ER 4-bit keys)	OP2K bit 0
9	OP2K bit 0	OP2K bit 1
10	OP2K bit 1	OP2K bit 2
11	OP2K bit 2	OP2K bit 3
12	Reserved (ER 4-bit keys)	ISK bit 0
13	ISK bit 0	ISK bit 1
14	ISK bit 1	ISK bit 2
15	ISK bit 2	ISK bit 3

Note: Reserved bits must be set to 0's.

Processor Status Word (PSW) Register

The processor status word (PSW) register contains 16 bits, and is formatted as follows:

Bit	Data
0	Specification check
1	Invalid storage address
2	Privilege violate
3	Protect check
4	Invalid function
5	Floating-point exception
6	Stack exception
7	ER architecture
8	Storage parity check
9	Reserved
10	Processor control check
11	I/O check
12	Sequence indicator
13	Auto IPL
14	Translator enabled
15	Power thermal warning

To indicate that the extended registers are active, bit 7=1 in the PSW. The I/O translator is enabled and 4-bit keys are supported, with the existing 16-bit address key register bits defined as follows:

Bit 7	Bit 14	
1	1	Processor and I/O translators and extended registers enabled.
0	1	Processor translator enabled; 3-bit keys active.
0	0	Translators disabled and only 64K of physical storage is accessible.
1	0	ER enabled; both translators disabled.

Both PSW bits 7 and 14 are reset to 0's with a power-on reset, system reset, IPL, or check restart.

Modified Instructions

Modifications have been incorporated into several Series/1 instructions pertaining to their use with the processor. These modifications apply only for 4-bit mode operation; they do not apply for 3-bit mode operation. The instructions that have been modified are:

- Set Address Key Register (SEAKR)
- Copy Address Key Register (CPAKR)
- Set Segmentation Registers (SESR)
- Copy Segmentation Registers (CPSR)
- Enable (EN)
- Disable (DIS)
- Copy Processor Status and Reset (CPPSR).

Note: For an explanation of the standard Series/1 instruction set, refer to *IBM Series/1 Principles of Operation*, GA34-0152.

Set Address Key Register (SEAKR)

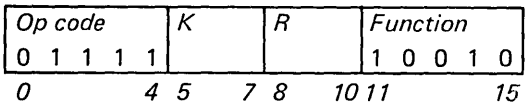
The Set Address Key Register (SEAKR) instruction has two formats:

- System register/register format for operations that load data from a specified register into the AKR.
- System register/storage format for operations that load data from main storage into the AKR.

System Register/Register Format

The system register/register format is:

Mnemonic	Syntax	Instruction name	K-field
SEAKR	reg	Set Address Key Register	011
SEISK	reg	Set Instruction Space Key	000
SEOOK	reg	Set Operand 1 Key	010
SEOTK	reg	Set Operand 2 Key	001



The address key register (AKR) field specified by the K-field is loaded from the register specified by the R-field. The contents of the register are not changed.

Note: The K-field can specifiy either a field within the AKR or an entire AKR.

K-field	AKR field name	3-bit mode bit number	4-bit mode bit number
000	ISK	13–15	12–15
001	OP2K	9–11	8–11
010	OP1K	5–7	4–7
011	AKR	0–15	0–15
100	Reserved		
101	Reserved		
110	Reserved		
111	Reserved		

Reserved K-fields must not be used.

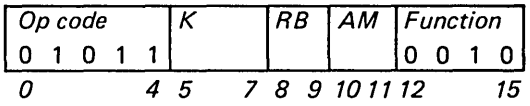
If the K-field specifies a specific field within the AKR, bits 13–15 (for 3-bit mode) or bits 12–15 (for 4-bit mode) from the word location in main storage are loaded into the AKR field. If the K-field specifies the entire AKR, bits 0–15 from the word location in main storage are loaded into the AKR.

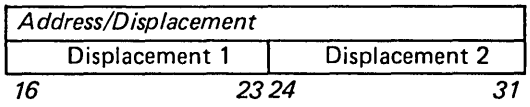
Indicators: The indicators are not changed.

Program Checks: This instruction format has the following program check:

- **Privilege violate.** The instruction is encountered while in problem state. The instruction is suppressed and a program-check interrupt occurs with privilege violate set in the PSW.

Mnemonic	Syntax	Instruction name	K-field
SEAKR	addr4	Set Address Key Register	011
SEISK	addr4	Set Instruction Space Key	000
SEOOK	addr4	Set Operand 1 Key	010
SEOTK	addr4	Set Operand 2 Key	001





The address key register (AKR) field specified by the K-field is loaded from the word location in main storage that is specified by the effective address. The contents of the word in main storage are not changed.

Note: The K-field can specify either a field within the AKR or an entire AKR.

K-field	AKR field name	3-bit mode bit number	4-bit mode bit number
000	ISK	13–15	12–15
001	OP2K	9–11	8–11
010	OP1K	5–7	4–7
011	AKR	0–15	0–15
100	Reserved		
101	Reserved		
110	Reserved		
111	Reserved		

If the K-field specifies a specific field within the AKR, bits 13–15 (for 3-bit mode) or bits 12–15 (for 4-bit mode) from the word location in main storage are loaded into the AKR field. If the K-field specifies the entire AKR, bits 0–15 from the word location in main storage are loaded into the AKR.

Indicators: The indicators are not changed.

Program Checks: This instruction format has the following program checks:

- **Invalid storage address.** One or more words of the instruction or the effective address are outside the installed storage size of the system. The instruction ends and a program-check interrupt occurs with invalid storage address set in the PSW.

- **Privilege violate.** The instruction is encountered while in problem state. The instruction is suppressed and a program-check interrupt occurs with privilege violate set in the PSW.

- **Specification check.** The effective address or indirect address results in an even-byte boundary violation. The instruction ends and a program-check interrupt occurs with specification check set in the PSW.

Copy Address Key Register (CPAKR)

The Copy Address Key Register (CPAKR) instruction has two formats:

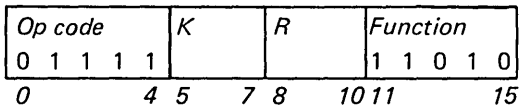
- System register/register format for operations that load data from the AKR into a specified register.

- System register/storage format for operations that load data from the AKR into main storage.

System Register/Register Format

The system register/register format is:

Mnemonic	Syntax	Instruction name	K-field
CPAKR	reg	Copy Address Key Register	011
CPISK	reg	Copy Instruction Space Key	000
CPOOK	reg	Copy Operand 1 Key	010
CPOTK	reg	Copy Operand 2 Key	001



The contents of the address key register (AKR) field specified by the K-field are loaded into the register specified by the R-field. The contents of the AKR are not changed.

Note: The K-field can specify a field within the AKR or the entire AKR.

K-field	AKR field name	3-bit mode bit number	4-bit mode bit number
000	ISK	13–15	12–15
001	OP2K	9–11	8–11
010	OP1K	5–7	4–7
011	AKR	0–15	0–15
100	Reserved		
101	Reserved		
110	Reserved		
111	Reserved		

If the K-field specifies a specific field within the AKR, the specified field is loaded into bits 13–15 (for 3-bit mode) or bits 12–15 (for 4-bit mode) of the register specified in the R-field. Bits 0–12 (for 3-bit mode) or bits 0–11 (for 4-bit mode) are set to 0's. If the K-field specifies the entire AKR, the AKR is loaded into the register.

Indicators: The indicators are not changed.

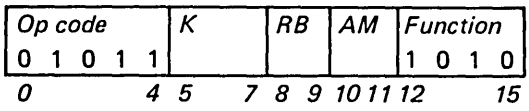
Program Checks: This instruction format has the following program check:

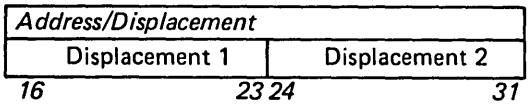
- **Privilege violate.** The instruction is encountered while in problem state. The instruction is suppressed and a program-check interrupt occurs with privilege violate set in the PSW.

System Register/Storage Format

The system register/storage format is:

Mnemonic	Syntax	Instruction name	K-field
CPAKR	addr4	Copy Address Key Register	011
CPISK	addr4	Copy Instruction Space Key	000
CPOOK	addr4	Copy Operand 1 Key	010
CPOTK	addr4	Copy Operand 2 Key	001





The contents of the address key register (AKR) field specified by the K-field are stored in the word location specified by the effective address. The contents of the AKR are not changed.

Note: The K-field can specify a field within the AKR or the entire AKR.

K-field	AKR field name	3-bit mode bit number	4-bit mode bit number
000	ISK	13–15	12–15
001	OP2K	9–11	8–11
010	OP1K	5–7	4–7
011	AKR	0–15	0–15
100	Reserved		
101	Reserved		
110	Reserved		
111	Reserved		

If the K-field specifies a specific field within the AKR, the specified field is stored in bits 13–15 (for 3-bit mode) or bits 12–15 (for 4-bit mode) of the word location in main storage. Bits 0–12 (for 3-bit mode) or bits 0–11 (for 4-bit mode) of the word in main storage are set to 0's. If the K-field specifies the entire AKR, the AKR is stored in the word location in main storage.

Indicators: The indicators are not changed.

Program Checks: This instruction format has the following program checks:

- **Invalid storage address.** One or more words of the instruction or the effective address are outside the installed storage size of the system. The instruction ends and a program-check interrupt occurs with invalid storage address set in the PSW.

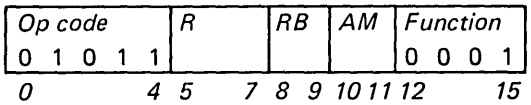
- **Privilege violate.** The instruction is encountered while in problem state. The instruction is suppressed and a program-check interrupt occurs with privilege violate set in the PSW.

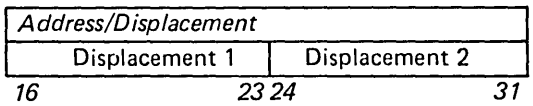
- **Specification check.** The effective address or indirect address results in an even-byte boundary violation. The instruction ends and a program-check interrupt occurs with specification check set in the PSW.

Set Segmentation Register (SESR)

The syntax for this instruction is:

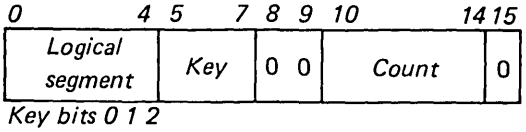
SESR reg,addr4





This instruction loads the contents of one or more doubleword storage locations, the first of which is specified by the effective address, into the segmentation registers specified by the contents of the register specified by the R-field.

For processors with 3-bit keys enabled, the format of the register specified by the R-field is:

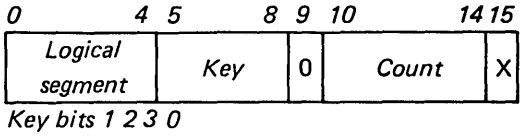


Bits 0–7 form the number of the segmentation register to be loaded (0–255). This number is comprised of 3-bits from the address key (values 0–7) and the five high-order bits of the logical storage address, which is the logical segment (values 0–31). Bits 8–15 of the register are reserved and must be set to 0's.

Notes:

1. *The count is equal to count plus 1. For example, a count of 0 loads one register.*
2. *If AM=01, the register selected by the RB field is incremented by 4 for each segmentation register set.*

When PSW bit 7 is on, the address key register extension bits are active. In 4-bit mode, the format of the register specified by the R-field is:



Note: Bit 8 is the most-significant bit in the key field and bit 7 is the least-significant bit. For example, 0011 in the key field would indicate stack 9.

Bits 0–8 form the number of the segmentation registers to be referenced (0–511) where the value of key bit 0 provides a select control between the first and second 256 segment register groups. This number is comprised of four bits from the address key (values 0–15) and the five high-order bits of the logical storage address, which is the logical segment (values 0–31). Bit 9 of the register is reserved and must be set to 0.

Bits 10–14 specify the number of contiguous segmentation registers to be set from contiguous doubleword locations (count plus 1). The instruction ends when the count is exhausted or when register 31 is loaded.

Bit 15, when a 1, indicates that the segmentation register stack referenced is the one reserved for I/O. Bit 8 (in the key field) must be set to 0.

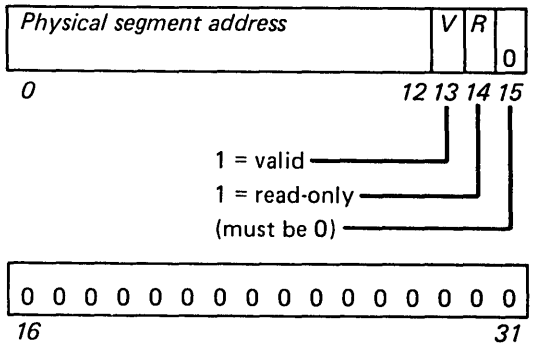
Notes:

1. *If AM=01, the register selected by the RB field is incremented by 4 for each segmentation register set.*

2. *All indicators are unchanged.*

Format of Doublewords

The first word (bits 0–15) of the specified doubleword that is loaded into the selected segmentation register has the following format:



The segment address (bits 0–12) contains the high-order bits of the physical address.

Bit 13, if a 1, signifies that the contents of the segmentation register are valid, and translation can be performed. If an attempt is made to use a segmentation register with bit 13 set to 0, a program-check interrupt occurs with invalid storage address set in the PSW.

Bit 14, if a 1, signifies that the block is read-only. If an attempt is made to write into the block when bit 14 of the associated segmentation register is a 1 and while in problem state, a program-check interrupt occurs, with protect check set in the PSW. The contents of main storage are not changed. When in supervisor state or on a cycle-steal access, bit 14 is ignored.

Bits 15–31 are reserved and must be set to 0's.

Indicators: The indicators are not changed.

Program Checks: This instruction has the following program checks:

- **Invalid storage address.** One or more words of the instruction or the effective address are outside the installed storage size of the system. The instruction ends and a program-check interrupt occurs with invalid storage address set in the PSW.

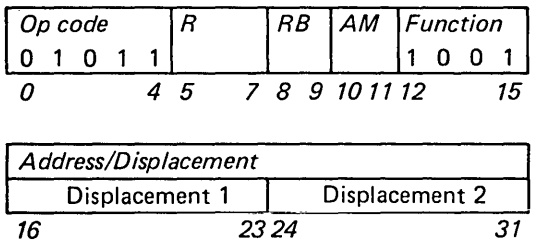
- **Privilege violate.** The instruction is encountered while in problem state. The instruction is suppressed and a program-check interrupt occurs with privilege violate set in the PSW.

- **Specification check.** The effective address or indirect address results in an even-byte boundary violation. The instruction ends and a program-check interrupt occurs with specification check set in the PSW.

Copy Segmentation Register (CPSR)

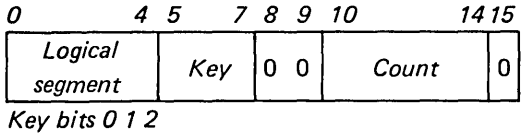
The syntax for the register/storage format is:

CPSR reg,addr4



This instruction stores the contents of one or more segmentation registers, specified by the contents of the register specified by the R-field, into contiguous doubleword storage locations, the first of which is specified by the effective address.

In the 3-bit mode, the format of the general register specified by the R-field is:

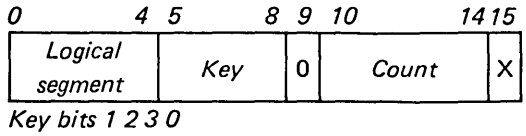


Bits 0–7 form the number of the segmentation register to be copied (0–255). This number is comprised of 3-bits from the address key (values 0–7) and the five high-order bits of the logical storage address which is the logical segment (values 0–31). Bits 8, 9, and 15 of the register are reserved and must be set to 0's.

Bits 10–14 specify the number of contiguous segmentation registers to be copied to contiguous doubleword storage locations and is equal to the count plus 1 (for example a count of 0 loads one doubleword storage location). The instruction ends when the count is exhausted or when register 31 of the key space has been copied.

Note: If AM=01, the register selected by the RB field is incremented by 4 for each segmentation register copied.

For processors with 4-bit mode enabled, the format of the register specified by the R-field is:



Note: Bit 8 is the most-significant bit in the key field, and bit 7 is the least-significant bit. For example, 0011 in the key field would indicate stack 9.

Bits 0–8 form the number of the segmentation register to be referenced (0–511) where the value of key bit 0 provides a select control between the first and second 256 segment register groups. This number is comprised of 4-bits from the address key (values 0–15) and the five high-order bits of the logical storage address, which is the logical segment (values 0–31).

Bit 9 of the register is reserved and must be set to 0.

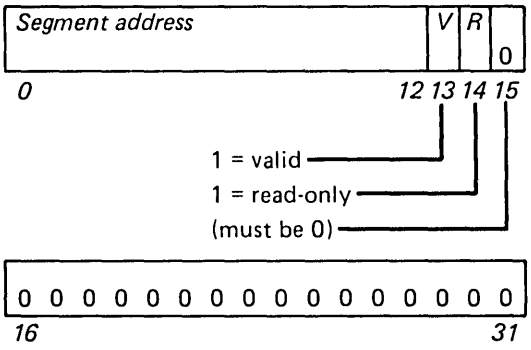
Bits 10–14 specify the number of contiguous segmentation registers to be copied to contiguous doubleword storage locations, with the count equal to the count plus 1 (for example, if the count is 0, one doubleword storage location is loaded).

Bit 15, when set to 1, indicates that the segmentation register stack referenced is the one reserved for I/O. Bit 8 (in the key field) must be set to 0.

Note: If AM=01, the register selected by the RB field is incremented by 4 for each segmentation register copied.

Format of Doublewords

The first word of the specified doubleword that is copied from the selected segmentation register has the following format:



The segment address (bits 0–12) contains the high-order bits of the physical address, which is used by the translator to select a 2K-byte block of main storage.

Bit 13, if a 1, signifies that the contents of the segmentation register is valid, and the translation can be performed. If an attempt is made to use a segmentation register in which bit 13 is a 0, a program-check interrupt occurs,with invalid storage address set in the PSW.

Bit 14, if a 1, signifies that the block is read-only. If an attempt is made to write into the block when bit 14 of the associated segmentation register is a 1 and while in problem state, a program-check interrupt occurs, with protect check set in the PSW. When in supervisor state or on a cycle-steal access, bit 14 is ignored; the contents of main storage are not changed.

The bits 15–31 are reserved and must be set to 0's.

Indicators: The indicators are not changed.

Program Checks: This instruction has the following program checks:

- **Invalid storage address.** One or more words of the instruction or the effective address are outside the installed storage size of the system. The instruction ends and a program-check interrupt occurs with invalid storage address set in the PSW.

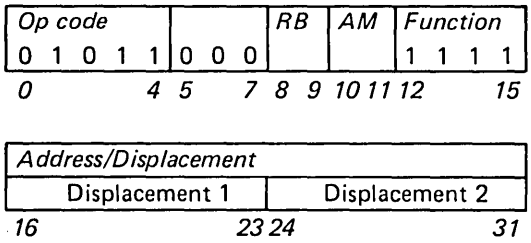
- **Privilege violate.** The instruction is encountered while in problem state. The instruction is suppressed and a program-check interrupt occurs with privilege violate set in the PSW.

- **Specification check.** The effective address or indirect address results in an even-byte boundary violation. The instruction ends and a program-check interrupt occurs with specification check set in the PSW.

Copy Processor Status and Reset (CPPSR)

The syntax for this instruction is:

CPPSR addr4



The contents of the processor status word (PSW) are stored at the word location in main storage specified by the effective address.

This instruction resets PSW bits 0–6 and 8–12; bits 7 and 13–15 are not changed. Bits 5–7 of the instruction are reserved and should be set to 0's.

Indicators: The indicators are not changed.

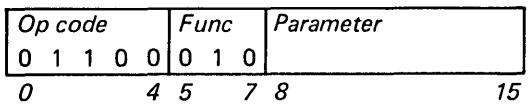
Program Checks: This instruction has the following program checks:

- **Invalid storage address.** One or more words of the instruction or the effective address are outside the installed storage size of the system. The instruction ends and a program-check interrupt occurs with invalid storage address set in the PSW.
- **Privilege violate.** The instruction is encountered while in problem state. The instruction is suppressed and a program-check interrupt occurs with privilege violate set in the PSW.
- **Specification check.** The effective address or indirect address results in an even-byte boundary violation. The instruction ends and a program-check interrupt occurs with specification check set in the PSW.

Enable (EN)

The syntax for this instruction is:

EN ubyte



The parameter field bits have the following significance:

Bit	Significance
8	Reserved
9	Reserved
10	Enable extended registers (PSW bit 7=1)
11	Reserved
12	Enable storage protect
13	Enable equate operand spaces (AKR bit 0 set to 1)
14	Enable translator (PSW bit 14 set to 1)
15	Enable summary mask (LSR bit 11 set to 1)

Note: Reserved bits must be set to 0's.

If bit 12 is set to 1, the relocation translator (if enabled) is disabled and bit 14 is not checked.

If bit 14 is set to 1, and bit 12 is set to 0, the relocation translator is enabled.

If parameter bit 14 and parameter bit 10 are both 1's (extended registers is enabled) or if parameter bit 14 and PSW bit 7 are both 1's, the I/O translator is enabled.

If parameter bit 14 is a 1, parameter bit 10 is a 0, and PSW bit 7 is a 0, the I/O translator is not enabled.

Indicators: The indicators are not changed.

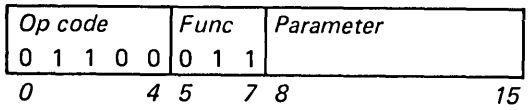
Program Checks: This instruction has the following program check:

- **Privilege violate.** The instruction is encountered while in problem state. The instruction is suppressed and a program-check interrupt occurs with privilege violate set in the PSW.

Disable (DIS)

The syntax for this instruction is:

DIS ubyte



The bits in the parameter field have the following significance:

Bit	Significance
8	Reserved
9	Reserved
10	Disable ER (PSW bit 7=0)
11	Reserved
12	Disable storage protect
13	Disable equate operand spaces (AKR bit 0 set to 0)
14	Disable translator (PSW bit 14 set to 0)
15	Disable summary mask (LSR bit 11 set to 0)

Note: Reserved bits must be set to 0's.

If a Disable instruction immediately follows an Enable Summary Mask instruction, the interrupt disable function may occur before an interrupt can be accepted. Thus, at least one other instruction (for example, no-op) must be inserted between the Enable Summary Mask and Disable instructions to ensure the occurrence of the interrupt.

If parameter bit 14 is set to 0 and parameter bit 10 = 1, the I/O and extended registers are disabled and bit 7 of the PSW is turned off.

If PSW bit 14 is a 1, all translators are disabled and PSW bits 7 and 14 are turned off.

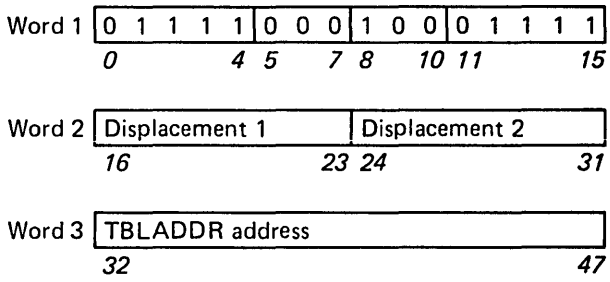
Indicators: The indicators are not changed.

Program Checks: This instruction has the following program check:

- **Privilege violate.** The instruction is encountered while in problem state. The instruction is suppressed and a program-check interrupt occurs with privilege violate set in the PSW.

Address Resolution and Indirect Branch (ARIB)

The ARIB instruction is a three-word instruction that performs the mapping actions. This instruction is intended to be used exclusively by the operating system. Mapping actions are the operation of fetching a command with an op code, operand 1, and operand 2, and then resolving the addressing modes used by those operands. The format of the instruction follows:



The instruction uses the values of displacement 1, displacement 2, and the address of the first word of the branch table TBLADDR for loading the ARIB instruction work registers, where:

- Displacement 1 is the index parameter displacement byte used with the control block designated by R2.
- Displacement 2 is the displacement byte parameter used with the control block designated by R2 to store R1.
- TBLADDR is the address of the first word of the branch-to table.

Register and Storage Conventions

The register conventions used by the mapping routine of the ARIB are as follows:

- Entry conditions:
  - R1 is the address of first word of the current command for ARIB.
  - R2 is the address of a control block.
- Exit conditions:
  - R1 and R2 are unchanged.
  - R3 is the resolved address of operand 1.
  - R4 is the resolved address of operand 2.
  - R5 is the contents of storage word pointed to by R1.
  - R6 is the TBLADDR index value that is defined as the command op code, multiplied by 2.
  - IAR is the contents of the storage location at the effective address TBLADDR plus the contents of R6.
  - Storage location is the effective address of the contents of R2 plus displacement 2 of ARIB instruction word 2, which contains the contents of R1.

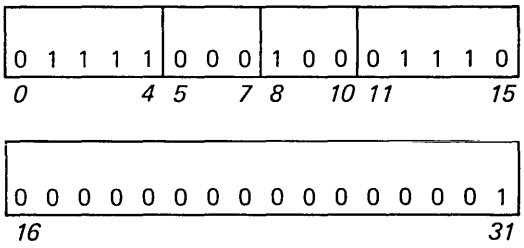
Indicators: The indicators are not changed.

Program Checks: This instruction has the following program checks:

- **Invalid function.** The instruction encountered the ARIB mode condition off. The instruction is suppressed and a program-check interrupt occurs with invalid function set in the PSW.
- **Invalid storage address.** One or more words of the instruction or any of the effective addresses is outside the installed storage size of the system. The indirect branch does not occur and the instruction is terminated. Some of the level registers may have changed.
- **Specification check.** The effective address or indirect address results in an even-byte boundary violation. Branching does not occur and the instruction is terminated. Some of the level registers may have changed.

Address Resolution Indirect Branch On (ARIBON)

The ARIBON instruction is a two-word instruction that activates the address resolution and indirect branch feature by turning on an internal flag that allows execution of the ARIB instruction. This mode is only reset by an IPL, a system reset, a power-on reset, or the ARIBOFF instruction. The ARIBON instruction is issued by the operating system in either the problem state or privilege state.

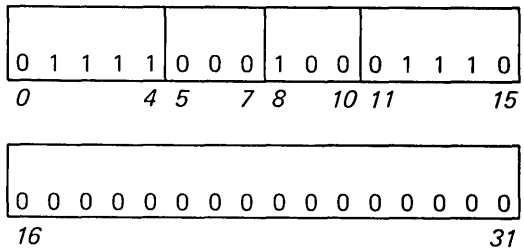


Word 2 bits 0–14 are reserved and must be set to 0's. Bit 15 must be set to 1.

Indicators: The indicators are not changed.

Address Resolution and Indirect Branch Off (ARIBOFF)

The ARIBOFF instruction is a two-word instruction that deactivates the address resolution and indirect branch feature by turning off the ARIBON flag. This instruction is issued by the operating system in the privilege state.



Word 2 is set to all 0's.

Indicators: The indicators are not changed.

ARIB Execution

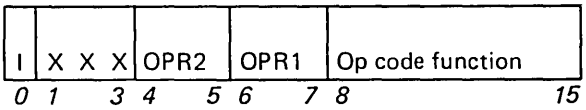
The required operations for the ARIB instruction are as follows:

Initialization: Prior to the invocation of the ARIB instruction, the operating system must issue the ARIBON instruction. For ordinary usage, this should be part of the IPL procedure.

Setup: Displacement 2 of the ARIB instruction word 2 is added to R2 to obtain a logical storage address, which used with OP2K to generate a physical storage address. The content of R1 is then placed in this register.

Note: There is no use of this storage address during the ARIB instruction, and both registers are maintained intact.

Command: A high-level command is fetched from the main storage location specified by the contents of R1 and placed into R5. The contents of R5 are defined as a command with the following definition:



Bit	Meaning
0	Operand 2 type (1=constant, 0=address)
1–3	Not defined
4–5	Register flag for operand 2
6–7	Register flag for operand 1
8–15	Operation code function

The I-field of the op code indicates to the mapping routine that the second operand is an immediate value. The OPR2 and OPR1 fields are used to define the addressing mode used for each of the operands.

The op code function field is used by the mapping routine to generate the index value for displacement in the table "TBLADDR."

The 2-bit register flag for each operand is defined as follows:

Field value	Meaning
0	Register not specified
1	Operand 1 used in the form (d, index register 1)
2	Operand 2 used in the form (d, index register 2)
3	Operand 1 or operand 2 used explicitly

Operand Address Resolution: Operand 1 is fetched from the main storage location at the effective address specified by the contents of R1 plus 2, and placed into R3.

Operand 2 is fetched from the main storage location specified by the contents of R1 plus 4, and placed into R4. If operand 2 is a constant, R4 contains the value R1 plus 4, which is a pointer to that constant.

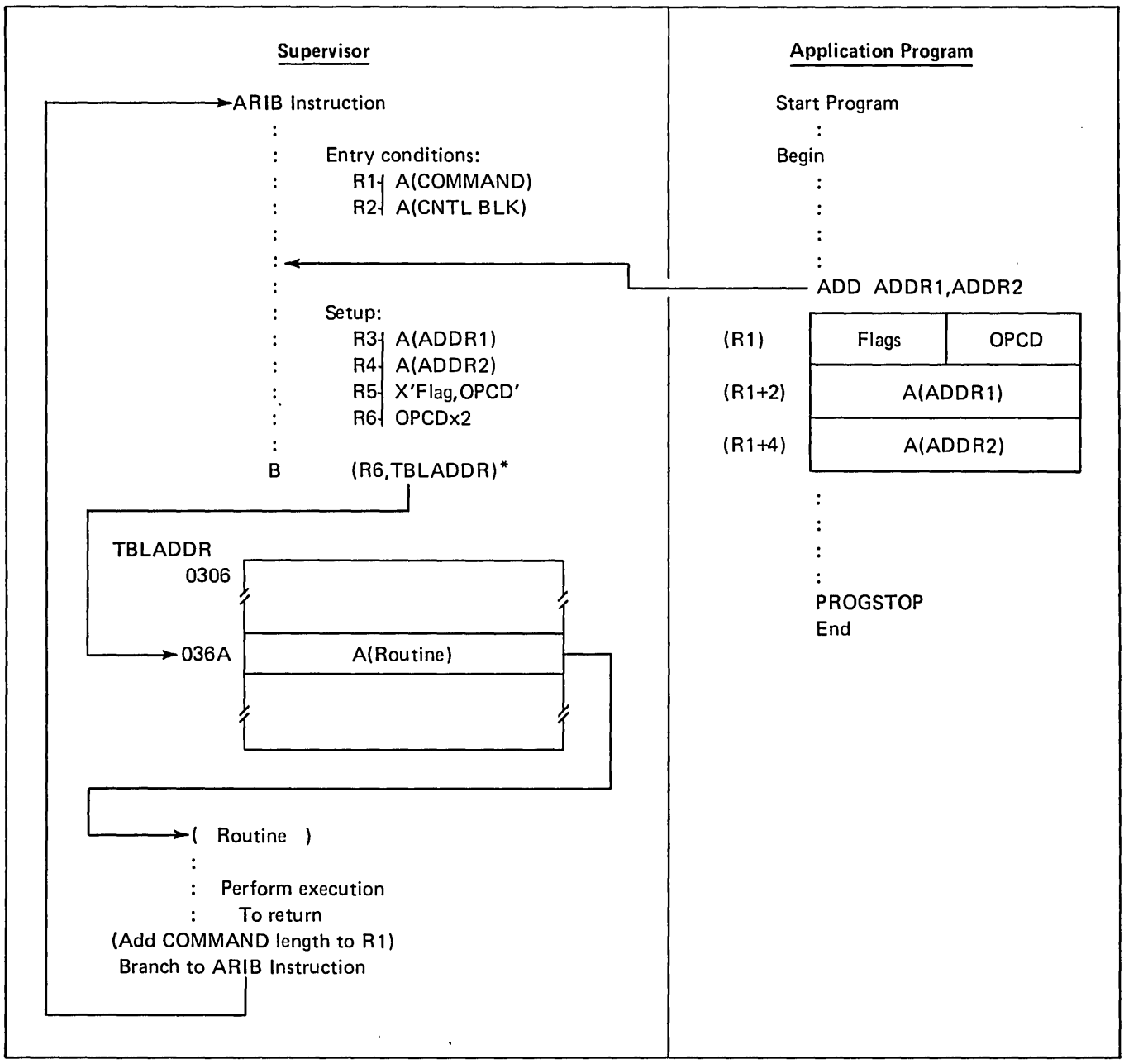
If operand 1 is an indexed value (parameter or index register), the setup routine adds the contents of the selected register (operand 1 or operand 2) to R3. If operand 2 is an indexed value (parameter, index register), the setup routine adds the contents of the selected register (operand 1 or operand 2) to R4.

These are accomplished as follows: an effective address is generated by adding displacement 1 with the contents of level register R2. The effective address is used with OP2K to generate the physical address. The contents of main storage at the generated address contains an address which points to the first word of a pair of index values. Reference to this storage location is required whenever command word bits 4 and 5, or 6 and 7, are respectively non-zero. Another address is then formed by adding to these contents the bits 4–5, for operand 1, or bits 6–7, for operand 2, to the low-order address word bit positions, thus adding bit positions 13, 14, and 15, where bit 15 is 0. This address is then used to fetch the storage location that is added to the appropriate operand register, either R3 or R4. This represents the operand indexed addressing.

After setting up the registers with the proper addresses, the branch-to table address is resolved as follows:

Address Resolution: The address of the routine that processes the next instruction is located in the branch-to table at an offset that is equal to the command op code multipled by 2. Thus, add the TBLADDR base address to twice the product of the op code in hex.

The next instruction routine for the ARIB processing is calculated as shown in the following figure. Using the contents of R6, plus the table address (TBLADDR), as an effective address used with the ISK to generate the physical address, the content of that storage location is loaded into the level instruction address register (IAR). Thus, performing an indirect branch through the branch-to table to the proper next instruction routine completes the ARIB instruction execution.



Op code	Hex 32
Multiplied by 2	Hex 64
Plus TBLADDR base offset of	Hex 0306
Equals the offset into TBLADDR of the routine that processes the next instruction	Hex 036A

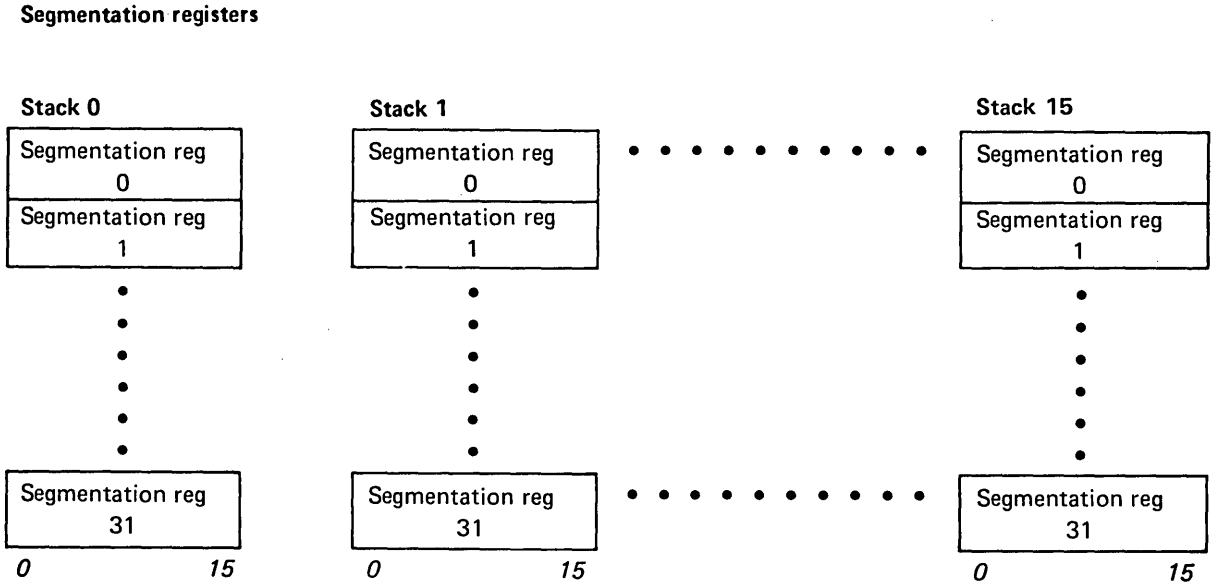
Chapter 2. Main Storage Addressing Using the Relocation Translator

The relocation translator and segmentation registers permit addressing of main storage locations beyond 64K bytes and provide a read-only type of storage protection. The first 64K bytes can be addressed directly when the translator is disabled; therefore, the translator must be enabled when main storage above 64K bytes is accessed.

Translator Description

The translator provides 16 stacks of 16-bit segmentation registers. The stacks are numbered 0–15 to correspond to the 16 possible values of the address keys. Each stack consists of 32 registers (0–31). In 3-bit mode, only stacks 0–7 (eight stacks) are used for translation, thereby allowing only 512K bytes of direct mappable storage.

When 4-bit keys are used, the translator uses 16 stacks of segmentation registers with 32 registers in each stack, allowing 1024K bytes (1 megabyte) of direct mappable storage. A particular stack is specified by the 4-bit active address key, whose value is from 0–15. Together the active address key and logical segment determine a unique segmentation register.



The stacks of segmentation registers are under supervisory program control. Four privileged instructions are used with the relocation translator and segmentation registers.

- **Set Segmentation Register (SESR).** This instruction loads segmentation registers.

- **Copy Segmentation Register (CPSR).** This instruction allows the supervisor to inspect the contents of segmentation registers.

- **Enable (EN).** This instruction enables the relocation translator. Until the translator is enabled, 16-bit addressing is in effect for the low-order 64K bytes of storage. Any storage above 64K bytes is not accessible to the program until the translator is enabled.

- **Disable (DIS).** This instruction disables the relocation translator.

For further information about the preceding instructions, refer to their descriptions in Chapter 1.

Storage Mapping

Mapping of main storage is achieved through the segmentation registers. Each segmentation register controls a 2K-byte segment of storage. The SESR instruction is used to load each segmentation register with the unique physical address of a 2K-byte segment of storage.

Note: More than one segmentation register can be loaded with the same segment address. For example, stack 0, register 15 (associated with the supervisor address key of 0), can be loaded with the same number as stack 1, register 6. This arrangement allows the supervisor to address control blocks within a problem program even though the address key for the supervisor is different than the key for the problem program. Once loaded, each stack of segmentation registers contains a complete map of 64K bytes divided into 2K-byte physical segments.

Relocation Addressing

The relocation translator generates a physical address that allows any byte in storage to be addressed. Figure 2-1 shows an example of address translation. The letters in the following description correspond to the letters in Figure 2-1:

Ⓐ The active address key from the address key register selects a segmentation register stack. The address key pertains to the instruction being executed on the current priority level. Bit 12 is ignored in 3-bit mode.

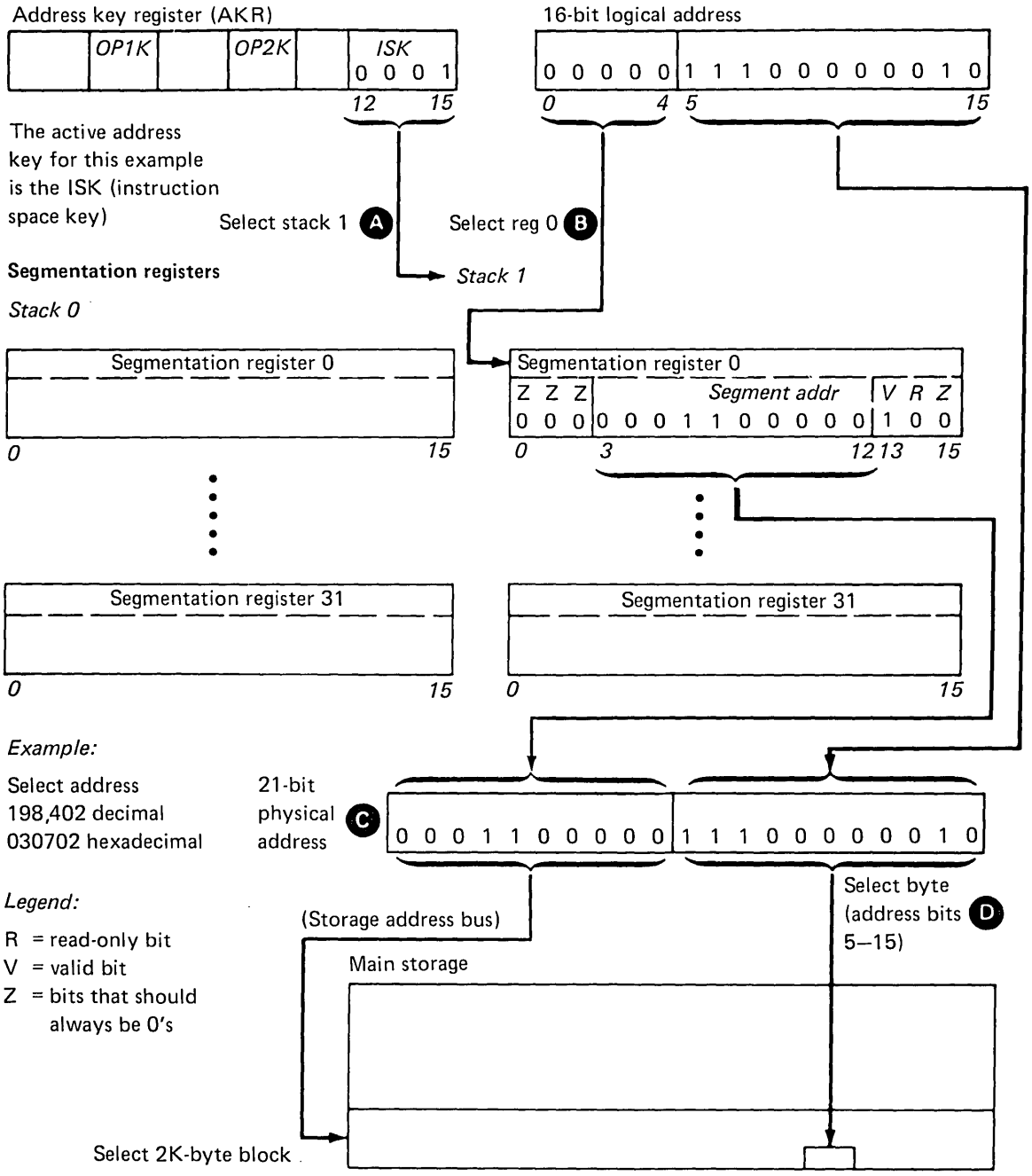
Ⓑ The five high-order bits (0–4) of the 16-bit address (generated for the instruction being executed) select a segmentation register within the stack selected in description Ⓐ. These bits define the logical segment.

Ⓒ The physical address is generated. The high-order bits are from the segmentation register; these bits specify the physical address of a 2K-byte segment of storage.

Segmentation Register Bit 13—Valid Bit: When set to 1, this bit specifies that the contents of the segmentation register are valid; the segmentation register can be used to perform the translation. When bit 13 is a 0, the segmentation register cannot be used for translation (no access). If translation is attempted, a program-check interrupt occurs with invalid storage address set in the processor status word (PSW). (All valid bits are set to 0's after power is switched on.)

Segmentation Register Bit 14—Read-Only Bit: When set to 1, this bit specifies that the block is read-only. If an attempt is made to write into storage using a segmentation register with the read-only bit set to 1, a program-check interrupt occurs with protect check set in the PSW. Storage is not changed. Bit 14 is ignored by a cycle-steal access or when the processor is in supervisor state.

Ⓓ The 11 low-order bits (13–15) of the physical address are the 11 low-order bits (5–15) of the 16-bit logical address (generated for the instruction being executed); these bits specify the byte address within the 2K-byte segment.



Note: When the translator is disabled, address bits 0–15 only are used for main storage address selection.

Figure 2-1. Address Translation

I/O Storage Access Using the Relocation Translator

All storage access requests from I/O devices are translated by the same hardware that handles storage requests from the processor. The device control blocks (DCBs) must reside in the supervisor's address space; therefore, all I/O devices must use address key 0 to gain access to the DCBs and to store the individual residual status blocks. The address key of the process requiring a cycle-steal operation resides in a DCB. An I/O device presents this address key, along with a 16-bit logical address, to the relocation translator. This allows an I/O device to directly address the storage space for a particular process. The address key allows I/O storage protection to be established between address spaces, assuming that the supervisor ensures the integrity of the DCBs. In 4-bit translated mode, cycle-steal operations gain access to main storage through the I/O segmentation registers.

Status of Translator After Power Transitions and Resets

The translator is enabled by the Enable (EN) instruction, or by the PSW key of the programmer console, if installed. The translator is disabled by any of the following:

- Disable (DIS) instruction
- Power-on reset
- Check Restart key on programmer console
- Initial program load (IPL)
- System Reset key on programmer console.

All translator controls are reset when the translator is disabled.

Notes:

1. *A machine-check interrupt does not disable the translator.*
2. *The segmentation registers are not reset when the translator is disabled.*
3. *The valid bits are all set to 0's when power is switched on.*

Error-Recovery Considerations

Invalid Storage Address (ISA)

The invalid storage address bit (bit 1 of the PSW) is set to 1 by any one of the following:

- Storage access was attempted using a physical address greater than the physical storage size installed.
- Storage access was attempted with bit 13 (valid bit) of the segmentation register set to 0. This signifies that the contents of the segmentation register are invalid.

The specific nature of the invalid storage address can be resolved as follows:

- Store the segmentation register following the program-check interrupt.
- Test the value of bit 13 in the selected segmentation register. When set to 1, this bit specifies that the contents of the segmentation register are valid; the segmentation register can be used to perform the translation. When bit 13 is a 0, the segmentation register cannot be used for translation (no access). If translation is attempted, a program-check interrupt occurs with invalid storage address set in the processor status word (PSW).
- Ensure that the segment address does not exceed the limits of the physical processor storage installed.

Protect Check

When the translator is enabled, a program-check interrupt with protect check set in the PSW is caused by an attempt to write into storage, while in the problem state, using a segmentation register with bit 14 (read-only) set to 1.

Storage is not changed. Bit 14 is ignored by a cycle-steal access or when in supervisor state.

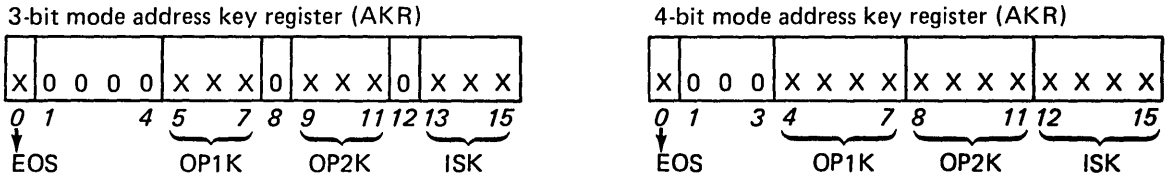
Address Space Management

Active Address Key

Cycle-steal devices have a cycle-steal address key specified in their device control block.

Any one of the four address keys (ISK, OP1K, OP2K, or cycle-steal address key) may be used during a storage access as the active address key. The address key in use (active) depends on the type of operation being performed at a specific instant in time. The active address key defines storage access through a particular block of segmentation registers.

Each priority level in the processor has an associated address key register (AKR) that contains an equate-operand-spaces (EOS) bit and three address keys (OP1K, OP2K, and ISK). In 3-bit mode, three bits are used in each address key; in 4-bit mode, four bits are used in each address key.



EOS *Equate operand spaces.* This bit, when set to 1, causes all data operands to use the OP2K address key. See "Equate Operand Spaces (EOS)" in this chapter.

OP1K *Operand 1 key.* These bits contain the binary-coded operand 1 address key, with bit 7 as the low-order bit.

OP2K *Operand 2 key.* These bits contain the binary-coded operand 2 address key, with bit 11 as the low-order bit.

ISK *Instruction space key.* These bits contain the binary-coded instruction-space address key, with bit 15 as the low-order bit.

Equate Operand Spaces (EOS)

The equate operand spaces bit (bit 0) in the address key register controls the use of the OP1K address key.

When the EOS bit is set to 1 (enabled), all processor data fetches use a single address space defined by the OP2K address key. The OP1K is ignored, but not changed, and all normal OP1K operations use OP2K as an active key. When the EOS bit is set to 0 (disabled), the OP1K address key functions in a normal manner.

Equate operand spaces (EOS) may be enabled by an Enable (EN) instruction, a Set Level Block (SELB) instruction, or a Set Address Key Register (SEAKR) instruction. EOS may be disabled by a Disable (DIS) instruction, a Set Level Block (SELB) instruction, or a Set Address Key Register (SEAKR) instruction. The EOS is also disabled by a priority interrupt or a class interrupt. These instructions are described in Chapter 1 and in *IBM Series/1 Principles of Operation*, GA34-0152.

Address Space

When the relocation translator is enabled, an address key defines a specific address space where:

- The address space is a range of logically contiguous storage.
- The address space is accessible by the effective address without operating system intervention (the address space is not greater than 64K bytes).

All instruction fetches use the address space defined by the instruction space key (ISK). For storage-to-storage instructions, all reads and writes for data operand 1 use the address space defined by the OP1K, assuming that the EOS bit is a 0. All other storage data accesses, reads, and writes use the address space defined by the OP2K, excluding branch and jump instructions.

Examples:

ISK=OP1K=OP2K. For instruction processing, all storage accesses occur within the same address space.

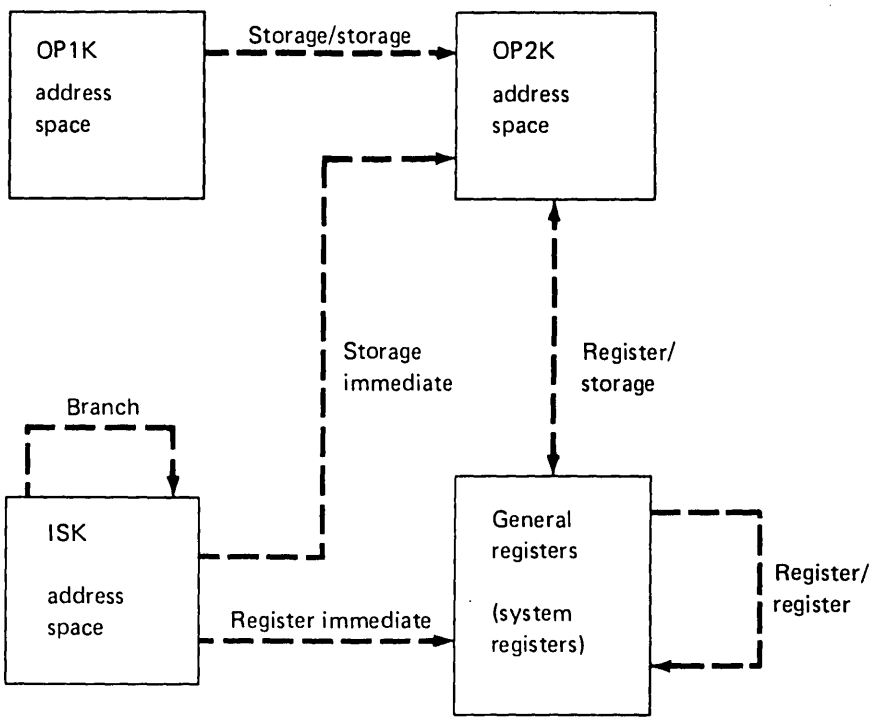
ISK≠OP1K, OP1K=OP2K. Instruction fetches occur in the ISK address space. Data access occurs in the OP2K address space.

ISK≠OP1K, OP1K≠OP2K. OP1K is only used for the source operand in storage-to-storage operations. OP2K is used for storage data access in all other operations (excluding branch/jump). ISK is used for instruction fetch and branch/jump operations. Refer to Figure 2-2 for this example.

I/O operations that access main storage also use an address key. Cycle-steal operations (read or write) use the cycle-steal address key specified within the device control block. An address key of 0 is used when the device fetches the device control block. Direct program control (DPC) operations that write data to storage use the OP2K address key.

Other defined uses of the address key register are as follows:

- All indirect access for branching uses the ISK.
- Effective-address generation occurs in the address space of the particular data operand. The appended words in the instruction are accessed by the ISK.
- Storage access from the console is defined by the SAR address key. Stop-on-address is based on the Stop On Address key when the translator is enabled.
- System reset and IPL set all address keys and the EOS bit to 0's.



Assembler syntax for address spaces (see Appendix A)

ISK	OP1K	OP2K	Example instructions	
	addr5	addr4	AW	addr5,addr4
	(reg)	(reg)	MVFD	(reg),(reg)
Bits 12-15 of AKR			MVBI	byte,reg
Bits 12-15 of AKR			B	longaddr*

*Indirect addressing.

Figure 2-2. Data Movement in Address Spaces (ISK≠OP1K, OP1K≠OP2K)

Address Key Values After Interrupts

When priority or class interrupts occur, certain values are set in the address keys of the affected AKR. These values anticipate the address spaces that the programmer might need for interrupt processing. Figure 2-3 shows the resulting AKR values for each type of interrupt:

Interrupt	EOS	OP1K	OP2K	ISK
Priority	0	0	0	0
Supervisor call	0	Note 1	0	0
Machine check	0	Note 2	0	0
Program check	0	Note 2	0	0
Soft-exception trap	0	Note 1	0	0
Trace	0	Note 3	0	0
Console	0	0	0	0
PTW	0	0	0	0

Notes:

1. *OP1K is set to the preceding key contained in OP2K.*
2. *OP1K is set to the last active processor address key.*
3. *OP1K is set to the preceding key contained in the ISK.*

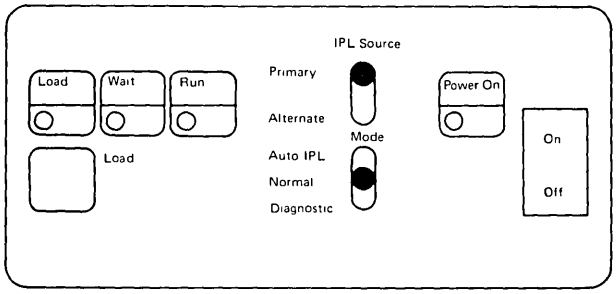
Figure 2-3. Resulting AKR Values

All interrupt service routines reside in address space 0; therefore, the ISK and OP2K are set to 0's when an interrupt occurs. Necessary information for processing a specific interrupt may reside in an address space other than 0. The address key related to the particular interrupt is placed in OP1K. The OP1K is set in anticipation of a storage-to-storage move of information from the interrupting address space to address space 0.

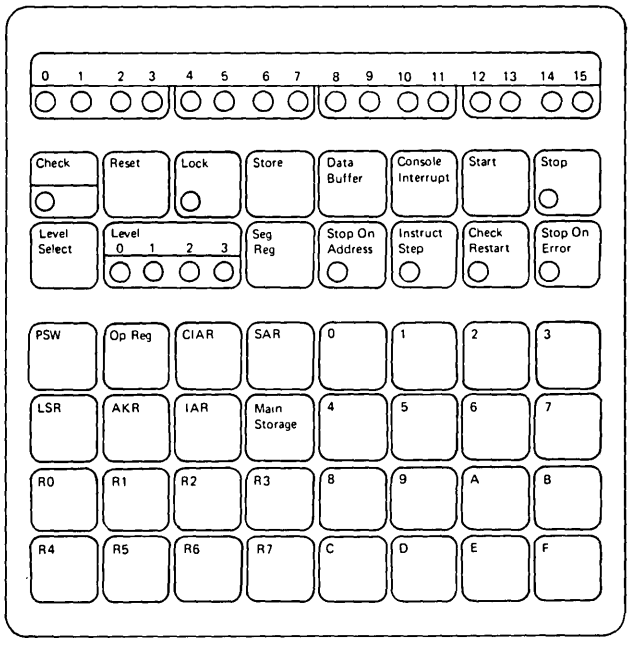
Note: Class interrupts cause a hardware-controlled storing of a level status block. This operation uses address key 0.

Chapter 3. Console

The basic console is a standard feature of the processor; the programmer console is an optional feature.



Basic console



Programmer console

The basic console is intended primarily for those systems that are totally dedicated to a particular application, where operator intervention is not needed during the execution of the application.

The programmer console is intended for operator-oriented systems where various programs are entered and executed. This type of environment requires a more versatile console arrangement for program and machine problem determination, and for manual alteration of data and programs in storage.

Basic Console

Each 4956 comes equipped with a basic console, which provides the following:

- Power On/Off switch for the processor unit
- IPL Source switch to select a primary or alternate IPL device
- Load key for initial program load (IPL)
- Mode switch to select: Auto IPL, Normal, or Diagnostic mode
- Load, Wait, Run, and Power On indicators.

Switches and Keys

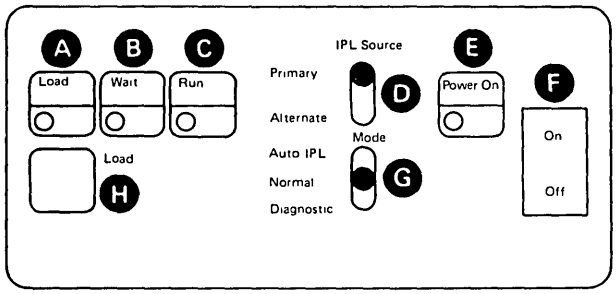
(F) **Power On/Off:** When this switch is set to the On position, power is applied to the processor unit. After all power levels are up, the Power On indicator is turned on. When this switch is set to the Off position, power is removed from the processor unit and the Power On indicator is turned off.

(D) **IPL Source:** This switch selects the I/O device to be used for program loading. In the Primary position, the device that was pre-wired as the primary IPL device is selected. In the Alternate position, the device that was pre-wired as the alternate IPL device is selected.

(H) **Load:** Pressing this key causes a system reset, and the initial program load (IPL) sequence is started. The Load indicator is turned on and remains on until the IPL sequence is completed. When the IPL sequence is completed, instruction execution begins at location 0 on priority level 0.

G **Mode:** This switch has the following positions:

- Auto IPL—In this position, an IPL is initiated after a successful power-on sequence. Bit 13 of the PSW is set to indicate to the software that an automatic IPL was performed. In this mode, Stop instructions are treated as no-ops.
- Normal—In this position, Stop instructions are treated as no-ops.
- Diagnostic—This position has no function without the programmer console. This position places the processor in diagnostic mode if the programmer console is attached. When the processor is in diagnostic mode, Stop instructions cause the processor to enter the stop state.



Indicators

A **Load:** On when the machine is performing an initial program load (IPL).

B **Wait:** On when an instruction that exits the active level has been executed and no other priority interrupts or levels are pending.

C **Run:** On when the machine is executing instructions.

E **Power On:** On when the proper power levels are available to the system.

Programmer Console

The programmer console is an optional feature that can be ordered with the 4956 or field-installed at a later date. The programmer console provides the following:

- Start and stop of the processor.
- Ability to display or alter any storage location.
- System reset.
- Selection of any one of the four interrupt levels for the purpose of displaying or altering data.
- Displaying or altering of the storage address register (SAR), instruction address register (IAR), SAR address key register (AKR), stop-on-address address key register (AKR), level address key register (AKR), segmentation registers, console data buffer, or any general purpose register.
- Displaying, but not altering, the level status register (LSR), current instruction address register (CIAR), op register, or processor status word (PSW). Note that the following bits of the PSW and LSR may be altered: PSW bit 14 (translator enabled), PSW bit 7 (4-bit mode), LSR bit 8 (supervisor state), and LSR bit 11 (summary mask).
- Address stop on instruction stream addresses.
- Stop on error.
- Instruction stepping.
- Check restart.
- Request for a console interrupt.
- Check indicator. The Check indicator is a light emitting diode (LED) that lights when a machine check or program check class interrupt occurs.
- Lock console.

The programmer console is touch-sensitive, with an audio-tone generator providing an audio response tone whenever a key is pressed and the information has been accepted and serviced by the processor.

Console Display

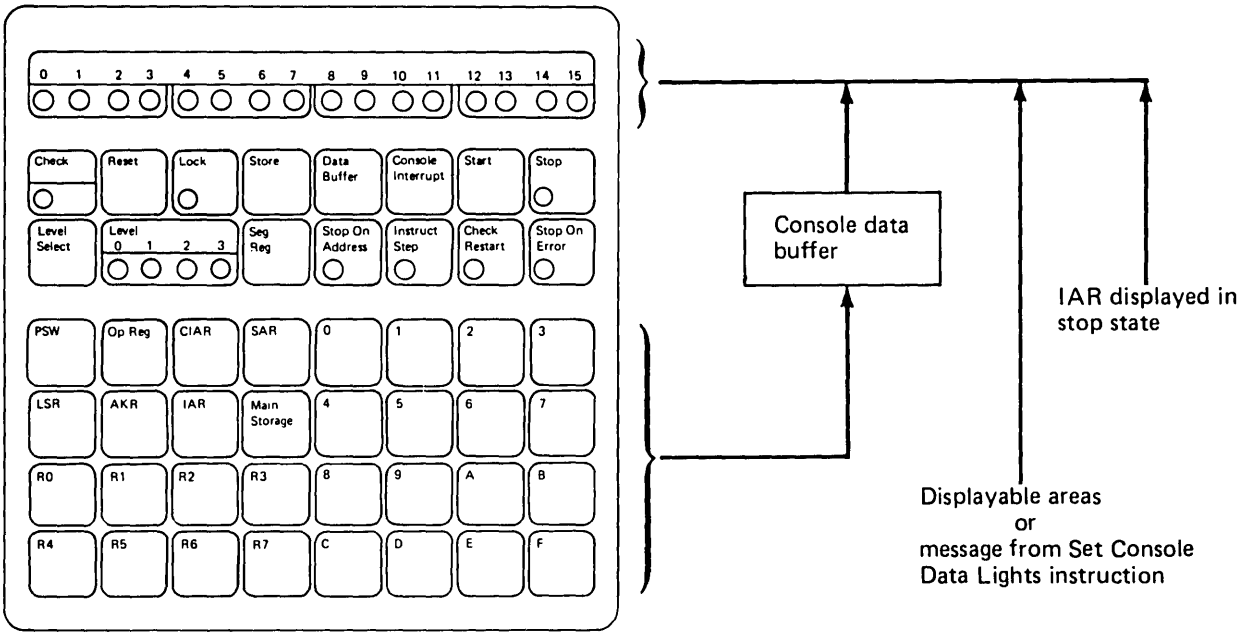
Run or Wait State

When the processor is in run or wait state, the console data buffer is displayed in the data display indicators. An exception to this is when a Set Console Data Lights (SECON) instruction writes a message to the data lights and does not change the buffer. When the Data Buffer key is pressed, the console data buffer is again displayed in the indicators.

When the console data buffer is being displayed, the console data buffer and the display are changed by entering new data with the data entry keys.

Stop State

When the processor enters stop state, the IAR is displayed in the data display indicators. Any system resource that has a corresponding select key on the console can be displayed. For example, the console data buffer can be displayed by pressing the Data Buffer key.



Power-On Reset

After a successful power-on reset, the data display indicators are set on, and the Stop indicator is set on (if the Mode switch is not positioned for Auto IPL).

Indicators

A Data Display: When the processor is in run state, the console data buffer is displayed in the data display indicators.

The Set Console Data Lights (SECON) instruction can write a message to the data display.

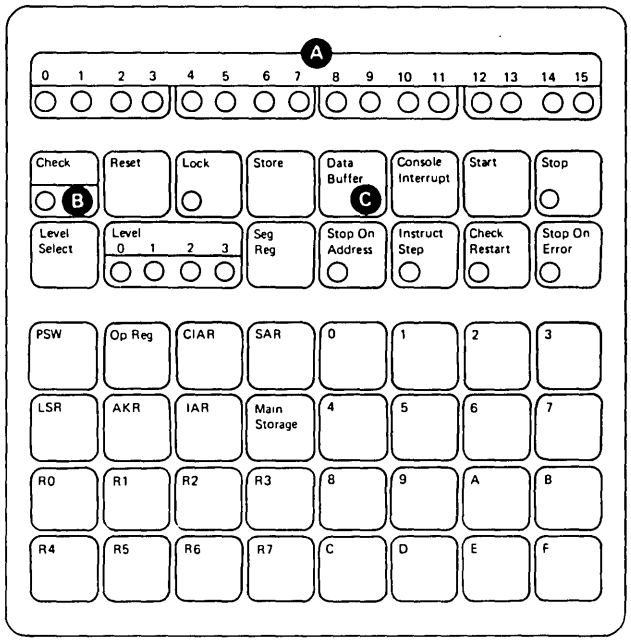
When the processor enters stop state, the IAR is displayed until another system resource is selected.

To display the contents of the console data buffer after a system resource has been displayed, press the Data Buffer key C.

B Check: On when a machine-check or program-check has been recognized. The Check indicator is turned off by:

- Clearing the check condition.
  - Reset key.
  - Load key.
  - Executing a Copy Processor Status and Reset (CPPSR) instruction. This instruction resets bits 0–6 and bits 8–12 of the PSW.
- Pressing any console key while in the stop state. The check condition is not cleared unless the Reset key or the Load key is pressed.

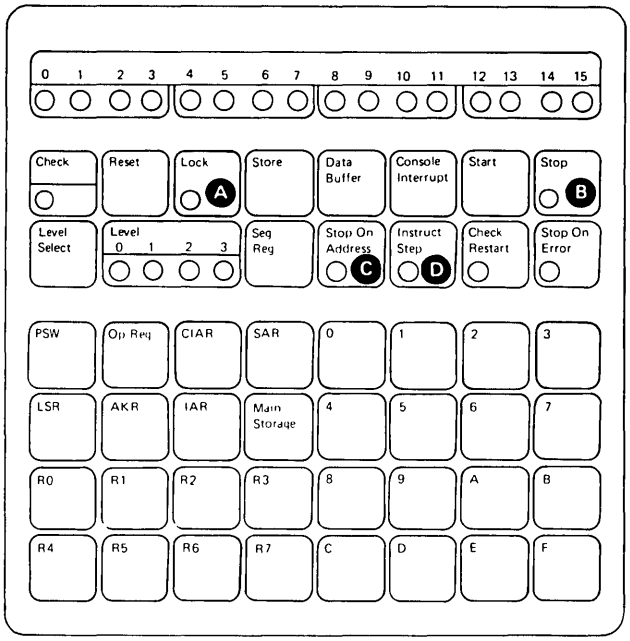
While in the stop state, the Check indicator is used to indicate main storage parity errors or invalid storage addresses during display operations. Refer to "Displaying Main Storage Locations" in this chapter.



Combination Keys/Indicators

There are six combination keys/indicators:

- Lock
- Stop
- Stop On Address
- Instruct Step
- Check Restart
- Stop On Error.



A Lock: Pressing the Lock key first, then pressing four hex keys and the Store key locks the console. A locked console is indicated by an illuminated indicator on the Lock key. The data LEDs are automatically set to the previous value. Displays or alterations cannot be performed with the programmer console keys while in the lock mode. The console remains locked until the same sequence of hex keys that locked the console is repeated and then followed by pressing the Store key.

The only data displayed during the lock mode is data set by the program.

Lock mode is automatically reset during a power-on sequence. If the console is locked and an auto IPL occurs after a power failure, the console will not be locked after the power-on.

If the console is locked in the stop mode, the only active switches are Lock, Store, and the hex keys. At this time, the run mode cannot be entered; therefore, for a normal lock function, lock mode should only be set during the run mode.

Ⓑ **Stop:** This indicator is on when the processor is in the stop state. Stop state is entered in the following ways:

- By pressing the Stop key.
  - In run state, the current instruction is completed.
  - In wait state, stop state is entered directly.
  - In stop state, the contents of the instruction address register (IAR) prior to entering the present stop state are restored to the IAR and displayed in the data display indicators. The level that was active upon entering stop state is reselected (becomes active).
- By execution of the Stop instruction (diagnostic mode only).
- When an address compare occurs in stop-on-address mode.
- When an error occurs in stop-on-error mode.
- By pressing the Reset key.
- When a power-on reset occurs.
- By selecting instruction step mode while in run state.

The Stop On Address key and the Instruct Step key are mutually exclusive. When one is pressed, the other is reset if it is on.

Ⓒ **Stop On Address:** Pressing this key places the processor in stop-on-address (SOA) mode and turns on the Stop On Address indicator. Pressing this key a second time resets stop-on-address mode and turns off the indicator. SOA allows access into instruction stream addresses only.

Ⓓ **Instruct Step:** Pressing this key places the processor in instruction step mode and turns on the Instruct Step indicator. Pressing this key a second time resets instruction step mode and turns off the indicator.

If the processor is in run or wait state, pressing this key causes the processor to enter stop state. Pressing the Instruct Step key a second time resets instruction step mode; the processor remains in stop state.

To operate in instruction step mode:

1. Key the desired starting address and store into the IAR.
2. Press the Instruct Step key.
3. Press the Start key. The instruction located at the selected address is executed, and the processor returns to stop state. The IAR is updated to the next instruction address; this address is displayed in the data display indicators.

Each time the Start key is pressed, one instruction is executed and the IAR is updated to the next instruction address.

Note: Priority and class interrupts are not inhibited during execution of the instruction.

Stop-On-Address Mode

The processor must be in stop state to set the compare address.

Stop On Address (Relocation Translator Disabled)

1. Press the Stop On Address key.

   Contents of the stop-on-address register are indicated by the display LEDs (instruction stream addresses only).

2. Enter the selected stop-on-address address by pressing the hex entry keys for a four-digit hex address.

3. Press the Store key.

   Contents of the updated stop-on-address register are indicated by the display LEDs.

4. Press the Start key.

   Execution begins at the current IAR address on the level that was active prior to entering the stop state.

   When the selected address is loaded into the IAR, the processor enters the stop state (IAR stop only). When the compare occurs, the stop state is entered with the IAR displayed in the data display indicators.

Stop On Address (Relocation Translator Enabled)

1. Press the Stop On Address key.

   Contents of the stop-on-address (SOA) register are indicated by the display LEDs.

2. Press the AKR (address key register) key.

   Contents of the stop-on-address address key register are displayed.

3. Enter the desired address key by pressing one hex entry key for a digit value (hex 0 through 7) for 3-bit mode, or (hex 0 through F) 4-bit mode.

4. Press the Store key.

   Contents of the updated stop-on-address key register are displayed.

5. Press the Stop On Address key.

   Contents of the stop-on-address register are indicated by the display LEDs.

6. Enter the selected compare address by pressing the hex entry keys for a four-digit hex address.

7. Press the Store key.

   Contents of the updated stop-on-address register are indicated by the display LEDs.

   The selected stop-on-address key register and stop-on-address register are used to compute a 21-bit physical address. Whenever the value in the segmentation register is changed, the physical address is recomputed.

   Note: The contents of the stop-on-address key register and the stop-on-address register may be displayed on the console; however, the 21-bit physical address cannot be displayed.

8. Press the Stop On Address key.

   The processor is now in stop-on-address mode.

9. Press the Start key.

   Execution begins at the current IAR address on the level that was active prior to entering the stop state.

   When the selected physical address is computed using the IAR and the ISK, the processor enters the stop state. The logical IAR is indicated by the display LEDs. The stop on address only stops on an instruction address.

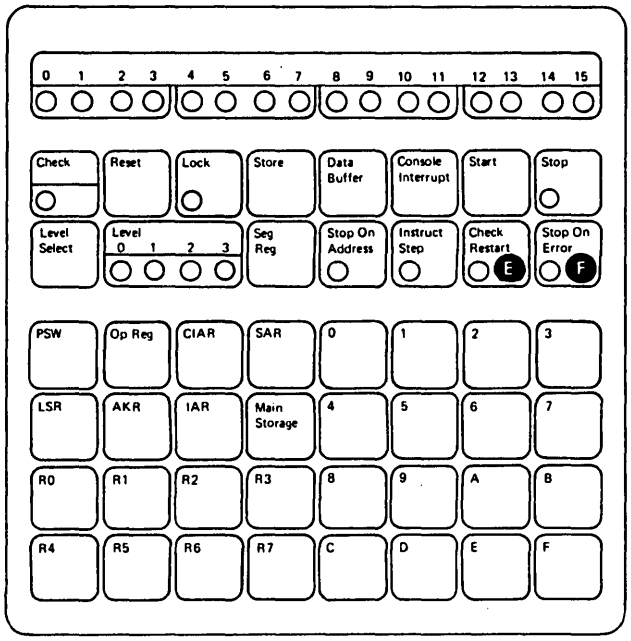
The Check Restart key and the Stop On Error key are mutually exclusive. When one is pressed, the other is reset if it is on.

E **Check Restart:** Pressing this key places the processor in check restart mode. While in this mode, a program-check, machine-check, or power thermal warning class interrupt causes the processor to be reset and execution to restart at address 0 on level 0.

Note: The power thermal warning stop-on-error condition is controlled by the summary mask.

F **Stop On Error:** Pressing this key places the processor in stop-on-error mode. Any program-check, machine-check, or power thermal warning class interrupt causes the processor to enter stop state. To determine the cause of the error, display the PSW. To restart the processor, press the Reset key and then the Start key. Pressing only the Start key allows the processor to proceed with the class interrupt as if stop mode had not occurred. Note that the Check indicator may have been turned off while in stop state. After the class interrupt routine is completed, control may be returned to the instruction that caused the error and an attempt to reexecute the instruction may be made. Some instructions are not reexecutable because operand registers or storage locations were changed before the instruction was terminated (because of the initial error). In these cases, the operator must be familiar with the program because manual restoration of affected locations must be made before restart is attempted.

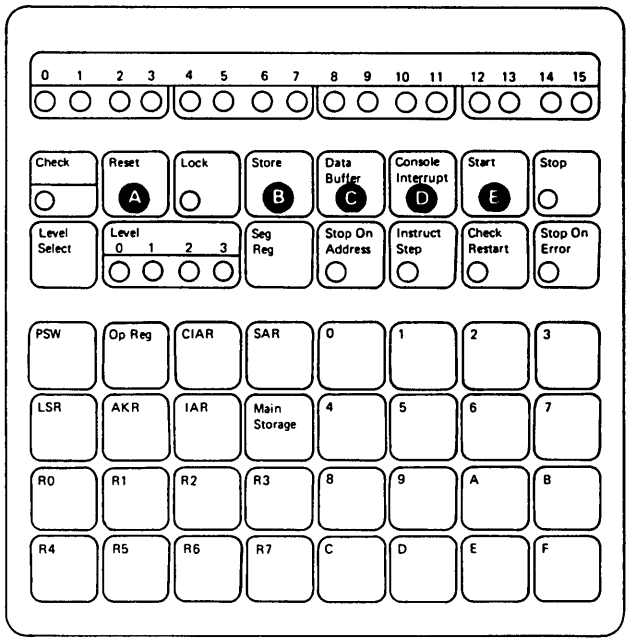
Note: The power thermal warning class interrupt is controlled by the summary mask.


0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
Check
Reset
Lock
Store
Data Buffer
Console Interrupt
Start
Stop
Level Select
Level 0 1 2 3
Seg Reg
Stop On Address
Instruct Step
Check Restart
E
Stop On Error
F
PSW
Op Reg
CIAR
SAR
0
1
2
3
LSR
AKR
IAR
Main Storage
4
5
6
7
R0
R1
R2
R3
8
9
A
B
R4
R5
R6
R7
C
D
E
F

A **Reset:** This key initiates a system reset that performs the following functions:

- IAR on level 0 set to 0
- AKR on level 0 set to 0
- Interrupt mask set to all levels enabled
- LSR on level 0—indicators set to 0's, summary mask enabled, supervisor state and in-process flag turned on, trace disabled
- LSRs for levels 1–3 set to 0's
- PSW bits 0–12 and 14 set to 0's (bit 14 set to 0 indicates translator disabled); bits 13 and 15 retain their state prior to system reset
- SAR set to 0
- CIAR set to 0
- Console display LEDs turned off
- Clock class interrupts disabled
- Return to 3-bit mode.

After the system reset is completed, the processor is placed in the stop state with the Stop indicator on.



The following are not affected by system reset:

- General registers (all levels)
- IARs (levels 1–3)
- AKRs (levels 1–3)
- Main storage
- Console data buffer
- Segmentation registers
- Stop-on-address register
- Clock
- Comparator.

B **Store:** This key is effective only when the processor is in stop state. Pressing this key causes the last data entry to be stored in the last selected resource.

C **Data Buffer:** Pressing this key causes the console data buffer to be selected. The contents of the console data buffer are displayed in the data display indicators.

D **Console Interrupt:** The effect of this key depends on the state of the processor. If the processor is in the stop or load state, this key has no effect. If the processor is in the run or wait state and the summary mask is enabled prior to the key action, a console-class interrupt occurs. The audio-response tone is generated when the interrupt is processed.

E **Start:** This key is effective in stop state only. Stop state is exited and the processor resumes execution at the address in the IAR on the current level. If stop state was entered from system reset, execution begins at address 0, level 0. If stop state was entered from wait state, the processor returns to wait state.

(H) PSW: Pressing this key selects the processor status word. The contents of the PSW are displayed in the data display indicators. Only PSW bit 14 (translator enabled) and PSW bit 7 (4-bit key enable) can be stored into the PSW from the programmer console.

(J) Op Reg: Pressing this key selects the op register and displays the contents in the data display indicators. Data cannot be stored into the op register from the console.

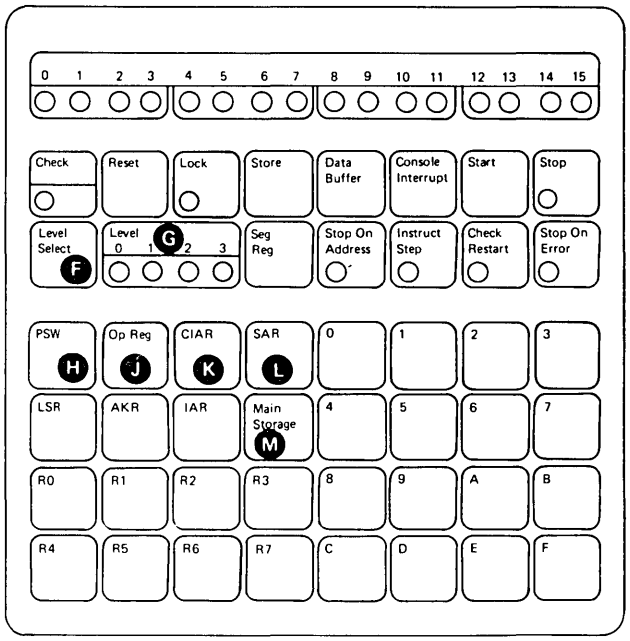
(K) CIAR: Pressing this key after entering stop state causes the address of the instruction just executed to be displayed. Data cannot be stored into the CIAR from the console.

(L) SAR: Pressing this key while in stop state displays the contents of the storage address register. An address can be stored into the SAR to address main storage or the segmentation registers for display or store operations. Bit 15 of the SAR cannot be set from the console.

(M) Main Storage: Pressing this key selects main storage as the facility to be accessed by the console. When this key is pressed, the contents of the main storage location addressed by the SAR are displayed in the data display indicators. Procedures for displaying and storing main storage are described in subsequent paragraphs in this chapter.

(F) Level Select: In the stop state, the Level Select key should be pressed first, before selecting a new level. The desired level may then be selected by pressing either the 0, 1, 2, or 3 hex key.

The current active level (level 0, 1, 2, or 3) is always displayed by one of the four level indicators at (G).



Level-Dependent Keys

The following keys select registers that are duplicated in hardware for each of the four interrupt levels:

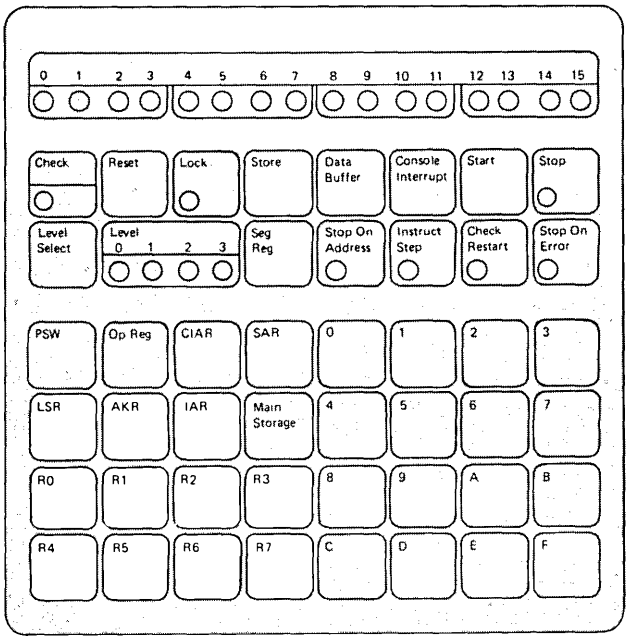
- LSR
- AKR
- IAR
- R0–R7 (General purpose registers 0–7).

Pressing any of these keys, once a level has been selected, causes the contents of that register to be displayed in the data display indicators.

The level status register (LSR) is displayable only, except bits 8 (supervisor state) and 11 (summary mask) can be stored into this register.

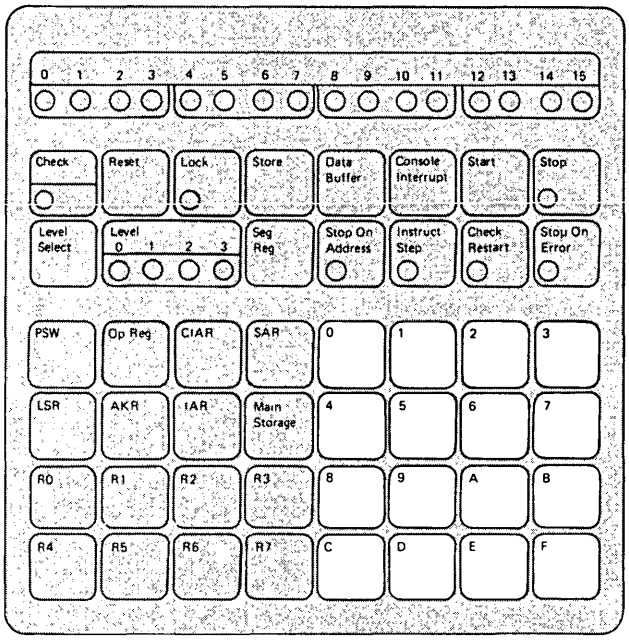
To display an AKR for a given level, enter the desired level, and then press the AKR key. The *level* AKR, bits 0, 5–7, 9–11, and 13–15 (EOS, OP1K, OP2K, and ISK) are displayed in the data display indicators for 3-bit mode. For 4-bit mode, EOS, OP1K, OP2K, and ISK are displayed in bits 0, 4–7, 8–11, and 12–15 of the data display indicators.

To display SAR AKR, first press SAR, then press AKR. To display the stop on address AKR, first press the Stop On Address key, then press AKR. To display CIAR AKR, first press CIAR, then press AKR (three or four bits, ISK). An AKR store is accomplished by first displaying the level AKR, then entering four hexadecimal digits, followed by pressing the Store key. When the Store key is pressed, the new level AKR is displayed. After the SOA AKR is displayed, enter one hexadecimal digit and press the Store key. After the SAR AKR is displayed, enter one hexadecimal digit and press the Store key for processor segmentation registers or enter the number 8 followed by one hexadecimal digit for the I/O segmentation registers. The CIAR AKR is displayable only.

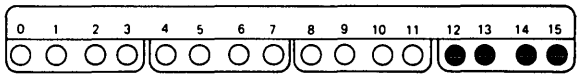
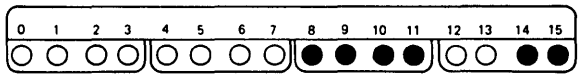
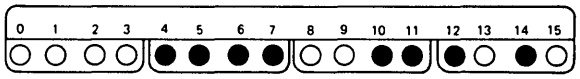
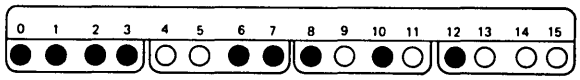


Data Entry Keys

The 16 data entry keys are used to enter data into a selected resource, such as main storage or a general register. When data is entered, it is shifted through the indicators, as shown in the following example:



Example: Data to be entered: F3A8

Action	Data display indicators
Press data entry key F	
Press data entry key 3	
Press data entry key A	
Press data entry key 8	

Legend:

● – Indicator on

○ – Indicator off

Displaying Registers

The processor must be in stop state.

1. Select the proper level by first pressing the Level Select key (C), and then the appropriate 0, 1, 2, or 3 hex data key.

   The contents of any register associated with the selected level can now be displayed by pressing a register key.

2. Press the desired register key. The contents of that register are displayed in the data display indicators (A).

Storing Into Registers

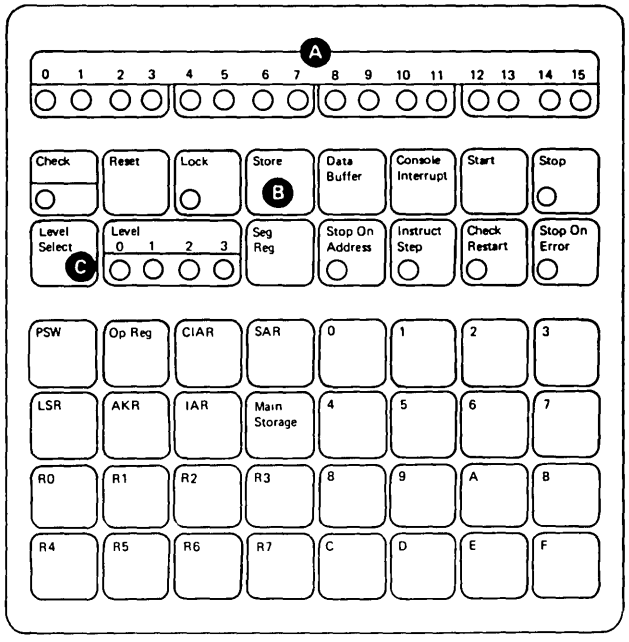
The processor must be in stop state.

1. Select the proper level by pressing the Level Select key (C), and then the appropriate 0, 1, 2, or 3 hex data key.

2. Press the key for the register where data is to be stored. The contents of that register are displayed in the data display indicators (A).

3. Key in the data that is to be stored. This data is displayed in the data display indicators (A).

4. Press the Store key (B). The data that is displayed is stored into the selected register.

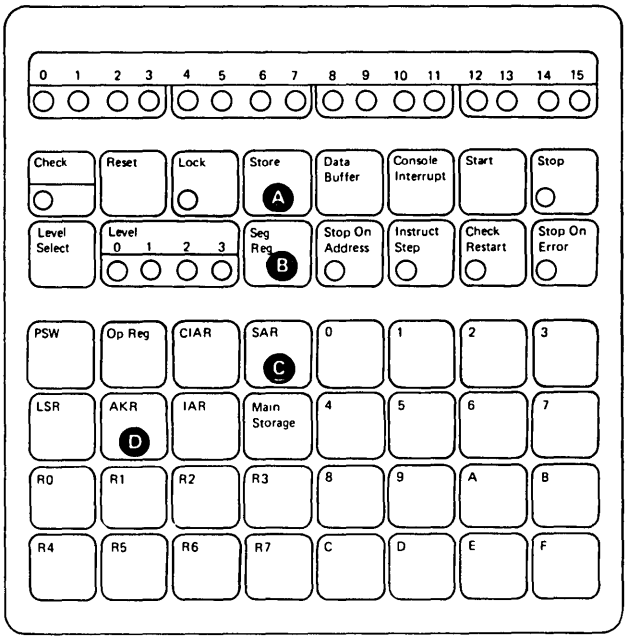


Displaying Segmentation Registers

The address relocation translator provides 16 stacks (0–15) of 32 segmentation registers (0–31) in each stack, for a total of 512 CPU segmentation registers. Refer to "Relocation Addressing" in Chapter 2. In addition, the address relocation translator provides eight stacks (0–7) of 32 segmentation registers (0–31) for a total of 256 I/O segmentation registers.

The processor must be in the stop state.

1. Press the SAR key (C). The contents of the SAR are displayed in the data display indicators.
2. Key in a hexadecimal four-digit number with the five high-order bits equal to the binary address (bits 0–31) of the desired segmentation register.
3. Press the Store key (A). The address is stored in SAR.
4. Press the SAR key (C). The selected address is displayed in the data display indicators.



5. Press the AKR key (D). The contents of the SAR address key register (AKR) are displayed in the data display indicators.
6. Key in one hexadecimal character to select the desired segmentation stack (0–F).
7. To select the I/O segmentation stack, key in hexadecimal 008X, where X is the desired stack (0–7).
8. Press the Store key (A). The value is stored in the SAR AKR.

9. Press the Seg Reg key (B). The contents of the selected segmentation register (defined by the five high-order bits of the SAR and the three/four SAR AKR bits) are displayed in the data display indicators.

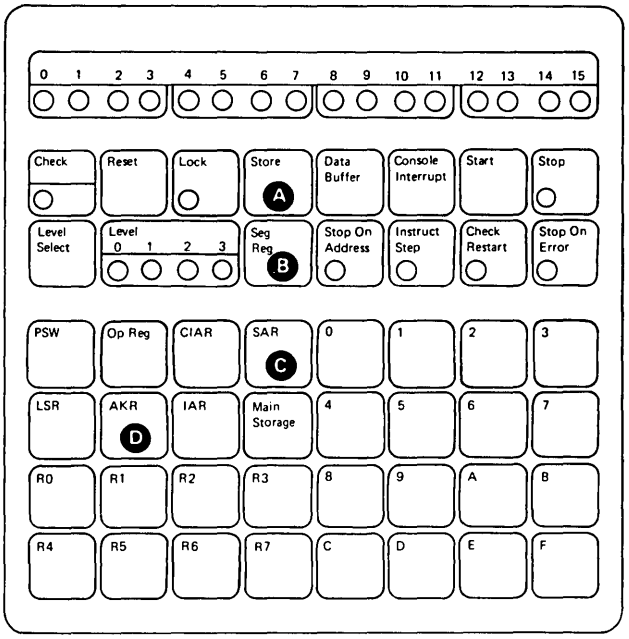
   Note: Each time the Seg Reg key is pressed, the segmentation- selection address is incremented by 1 until the last segmentation register in the stack is selected. Then, the segmentation-selection address wraps from 31 to 0. When the segmentation-selection address wraps from 31 to 0, the SAR AKR is incremented by 1 (a new segmentation-register stack is selected); the new segmentation-register contents are displayed in the data display indicators.

Storing Into a Segmentation Register

The address relocation translator provides 16 stacks (0–15) of 32 segmentation registers (0–31) for a total of 512 CPU segmentation registers. Refer to "Relocation Addressing" in Chapter 2. In addition, the address relocation translator provides eight stacks (0–7) of 32 segmentation registers (0–31) for a total of 256 I/O segmentation registers.

The processor must be in the stop state.

1. Press the SAR key (C). The contents of the SAR are displayed in the data display indicators.



2. Key in the value that selects the desired segmentation register within a stack (four hex characters entered with the data entry keys).

3. Press the Store key (A). The selected address is stored in the SAR.

4. Press the SAR key (C). The selected address is displayed in the data display indicators.

5. Press the AKR key (D). The contents of the SAR address key register (AKR) are displayed in the data display indicators.

6. Key in one hex character with a data entry key (any value from 0 through F, which is the new address key that selects a segmentation-register stack). This character is displayed in bits 12–15 of the data display indicators.

7. To select the I/O segmentation stack, key in hexadecimal 008X, where X is the desired stack (0–7).

8. Press the Store key (A). The contents of the SAR address key register (AKR) are updated to the value entered from the data entry keys.

9. Press the Seg Reg key (B). The contents of the selected segmentation register (defined by the five high-order bits of the SAR and the three/four SAR AKR bits) are displayed in the data display indicators.

10. Key in the value (four hex characters entered at the data entry keys) that provide both the desired ten high-order bits of the 21-bit physical main storage address (select a 2K-byte block of main storage) and that contain the correct value for the valid bit and the read-only bit.

11. Press the Store key (A). The selected segmentation register is updated to the value in the data display indicators.

    Note: Each time the Store key is pressed, the last value keyed is entered into the selected segmentation register and the segmentation selection address is incremented by 1 until the last segmentation register in the stack is selected. Then, the segmentation selection address wraps from 31 to 0. When the segmentation selection address wraps from 31 to 0, the SAR AKR is incremented by 1 (a new segmentation-register stack is selected); the new segmentation-register contents are displayed in the data display indicators.

The segmentation registers can be written into by the Set Segmentation Register (SESR) instruction and can be displayed by the Copy Segmentation Register (CPSR) instruction. Refer to *IBM Series/1 Principles of Operation*, GA34-0152, for additional information.

Displaying Main Storage Locations

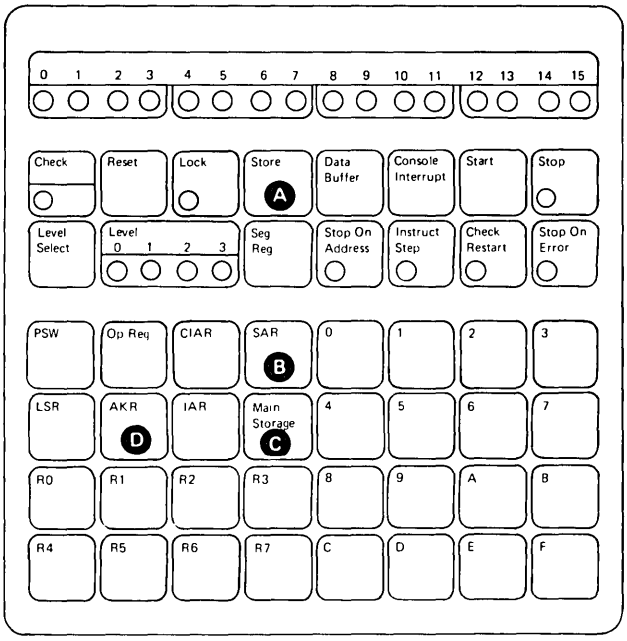
The processor must be in stop state.

If the storage address relocation translator is enabled, start at step 1; otherwise, start at step 5.

Note: If steps 1 through 4 of the procedure are used, it is assumed that the operator has a thorough knowledge of the relocation translator and the storage mapping assigned by the program.

1. Press the SAR key (B). The contents of SAR are displayed in the data display indicators.

2. Press the AKR key (D). The contents of the SAR AKR are displayed in the data display indicators.



3. Key in one hex character (value of 0 through F, which is the new address key). This character is displayed in bits 12–15 of the data display indicators.

4. Press the Store key (A). The new address key is stored into the SAR AKR.

5. Press the SAR key (B). The contents of the SAR are displayed in the data display indicators.

6. Key in the selected address (four hex characters). This address is displayed in the data display indicators.

7. Press the Store key (A). The address that is displayed is stored into the SAR.

8. Press the Main Storage key (C). The contents of the addressed storage location are displayed in the data display indicators and SAR is incremented by 2. Each time the Main Storage key is pressed, the location addressed by SAR is displayed in the data display indicators and then SAR is incremented by 2.

Notes:

1. If an invalid storage address occurs:

   a. The program check is suppressed.

   b. PSW bit 1 is set to 1.

   c. The Check indicator is turned on.

   d. PSW bit 1 set does not cause a class interrupt to occur upon entering the run state unless the check indicator is not reset. The bit is only an indication, to the operator, of an error while displaying main storage.

2. If a storage location with bad storage parity occurs:

   a. The program check is suppressed.

   b. PSW bit 8 is set to 1.

   c. The Check indicator is turned on.

   d. PSW bit 8 set does not cause a class interrupt to occur upon entering the run state unless the check indicator is not reset. The bit is only an indication, to the operator, of an error while displaying main storage.

Storing Into Main Storage

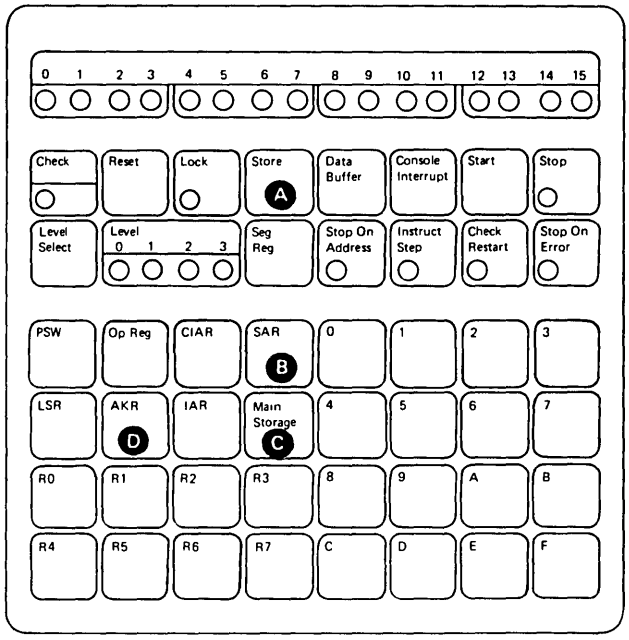
The processor must be in stop state.

If the storage address relocation translator is enabled, start at step 1; otherwise, start at step 4.

Note: If steps 1 through 3 of the procedure are used, it is assumed that the operator has a thorough knowledge of the relocation translator and the storage mapping assigned by the program.

1. Press the SAR key (B). The contents of SAR are displayed in the data display indicators.

2. Press the AKR key (D). The contents of the SAR AKR are displayed in the data display indicators.



3. Key in one hex character (a value of 0–F, which is the new address key). This character is displayed in bits 12–15 of the data display indicators.

4. Press the Store key (A). The new address key is stored into the SAR AKR.

5. Press the SAR key (B). The current contents of the SAR are displayed in the data display indicators.

6. Key in the selected address (four hex characters). The address is displayed in the data display indicators.

7. Press the Store key (A). The address displayed in the data display indicators is stored into the SAR.

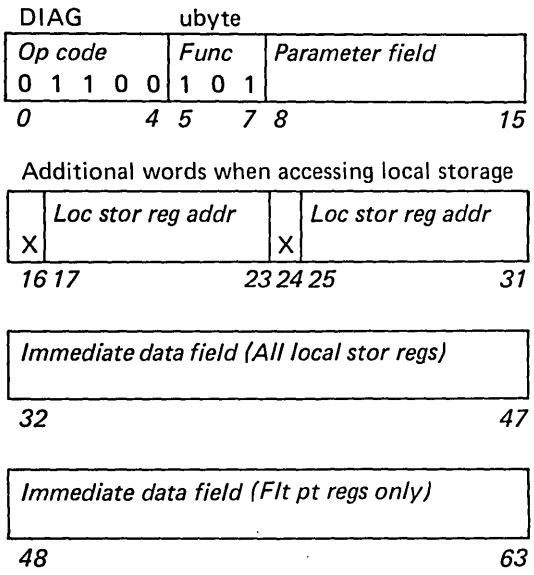
8. Press the Main Storage key (C). The contents of the addressed storage location are displayed in the data display indicators.

9. Key in the data that is to be stored into main storage. This data is displayed in the data display indicators.

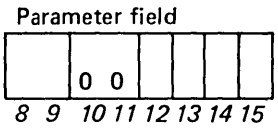
10. Press the Store key (A). The data that is displayed is stored at the selected storage location and SAR is incremented by 2. Repeat steps 9 and 10 to store in sequential storage word addresses, or repeat steps 8, 9, and 10 if sequential storage words are to be displayed before alteration.

Chapter 4. Diagnose (DIAG) Instruction

The DIAG instruction is used for controlling or testing various hardware functions.



The parameter field is used to define and select the functions of the DIAG instruction. The bits in the parameter field are as follows:



Note: Bits 10 and 11 are set to 0's.

Bits	Value	Function
8–9	00	Storage select (word)
	01	Set or reset self-modifying code detector
	10	Local storage register select
	11	Channel select
10–11	00	Reserved; must be set to 0
12	0	Storage-to-register data transfer
	1	Register-to-storage data transfer
13	0	Enable all other parameter bit functions
	1	Set system ID (All other parameter bit functions disabled)
14	0	Disable (ECC, channel interrupt requests, channel cycle-steal requests, or self-modifying code detector)
	1	Enable (ECC, channel interrupt requests, channel cycle-steal requests, and self-modifying code detector)
15	0	Enable all other parameter bit functions
	1	Microdiagnostic test function; disable all other parameter bit functions except set system ID

Storage Select

The storage select function provides for testing of the error correction code (ECC) generation, single-error correction, and double error detection in fitted storage.

Note: During a write storage cycle, ECC generates five code bits for each 8-bit data byte written to create a 26-bit word in storage. These code bits provide for the single-error correction/double error detection capability in the read storage cycle. When a double-error in storage is detected on a processor read, a machine-check interrupt occurs with PSW bit 8 set to 1 (storage parity error).

When parameter-field bits 8 and 9 = 00 (storage select), the other parameter-field bit definitions are described in the following:

Bits 10, 11: Must be set to 0's.

Bit 12: Specifies the direction of the data transfer.

- Bit 12=0. Transfer is from main storage to a register (read storage).
- Bit 12=1. Transfer is from a register to main storage (write storage).

Bit 13: Must be set to 0.

Bit 14: Specifies ECC generation, single-error correction, and double-error detection.

- Bit 14=0—Disable. ECC is not generated, single errors are not corrected, and double errors are not detected.
- Bit 14=1—Enable. ECC is generated, single errors are corrected, and double errors are detected.

Bit 15: Must be set to 0.

When self-modifying code is detected and the mode switch is in the diagnostic mode position, the processor enters the stop state. The console display indicators contain the IAR at which the self-modifying code was detected.

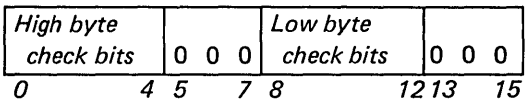
The self-modifying code detector can only be reset by the DIAG instruction which resets the self-modifying code detector, or a power-on reset.

Storage Select Word

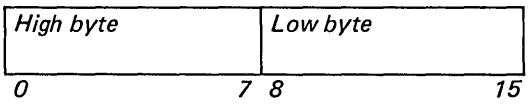
Parameter-field bits 8 and 9 are set to 0's.

The storage address for this data transfer cycle is contained in register 7 of the current priority level; the data is contained in registers 0 and 1 of the current priority level. Two bytes of data plus check bits are transferred as follows:

Register 0



Register 1



Bit 14: Disable/enable.

- Bit 14=0—Disable.

  Read storage. Single errors are not corrected, and double errors are not detected. Registers 0 and 1 contain the data and code bits of a storage read cycle.

  Write storage. ECC is not generated. The data word in register 1 and the check bits in register 0 are written into storage.

- Bit 14=1—Enable.

  Read storage. Single errors are corrected, and double errors are detected. Register 1 is the destination for the data bits of a storage read cycle.

  Write storage. ECC is generated.

When bit 14=1 (enable), a normal read or write word (bit 12 set to 0 or 1) occurs in storage.

Self-Modifying Code Detector Select

Storage select is by word. When parameter-field bits 8 and 9 = 01 (self-modifying code detector select). the other parameter-field bit definitions are described as follows:

Bits 10, 11: Must be set to 0's.

Bit 12: This bit is ignored for self-modifying code detector select.

Bit 13: Must be set to 0.

Bit 14: Specifies whether the self-modifying code detector is set or reset.

- 0—reset; self-modifying code is undetected.
- 1—set; self-modifying code is detected.

Bit 15: Must be set to 0.

Note: The self-modifying code detector attempts to detect self-modifying code beyond the pre-fetch limits of the 4956 model E processors. Modifications of the three words following the current instruction (4956E buffer size) may not be detected.

Local Storage Register Select

The local storage register select function permits the tranfer of data between main storage and any other local storage register. To select this function, parameter-field bit 8 is set to 1 and bit 9 is set to 0.

Note: The processor uses the local storage registers to store various machine parameters. Changing these parameters is not recommended because the results cannot be predicted.

Parameter-field bits that are used with this function are defined as follows:

Bits 10, 11: Must be set to 0's.

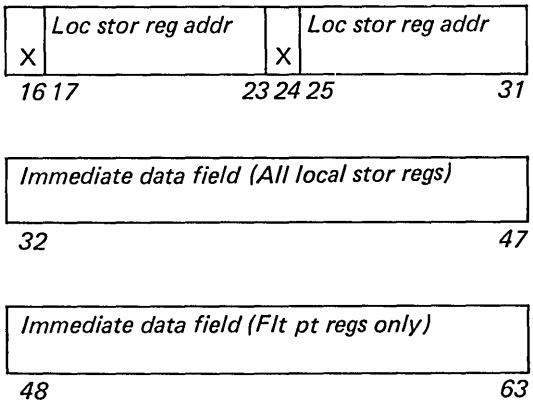
Bit 12: Specifies the direction of the data transfer.

- Bit 12=0. The transfer is from the immediate data field to the specified local storage register.
- Bit 12=1. The transfer is from the specified local storage register to the immediate data field.

Bits 13–15: Must be set to 0's.

The DIAG instruction has two additional words appended when this function is specified.

Additional words when accessing local storage



The bits of the two appended words are defined as follows:

Bits 16 and 24:

00—Access general purpose registers (GPR), floating-point registers, and extended registers.

01—Access level status registers on the current level only. The appended level status address word must be set to the address of GPR 0 of the desired level.

10—Access the IAR word of the IAR/AKR doubleword. Bits 16–20, 24, and bits 16, 24–28 of the appended level status address word determine the LS word selection for locations accessed by the IAR doubleword (IARD).

11—Access the AKR word of the IAR/AKR doubleword. Bits 16–20, 24 and bits 16, 24–28 of the appended level status address word determine the LS word selection for locations accessed by the IAR doubleword (IARD).

Bits 17–23: Local storage register address (00–7F). The data in bits 17–23 and 25–31 of the appended level status address word must be equal.

Bits 25–31: Local storage register address (00–7F). The data in bits 17–23 and 25–31 of the appended level status address word must be equal.

Bits 32–47: Immediate data to be transferred; all local storage locations.

Bits 48–63: Immediate data to be transferred; floating-point registers only.

Channel Select

The channel select function is determined by parameter-field bits 8 and 9 being set to 1's.

This function with bit 14 of the parameter field set to 0 (disable), inhibits and logically isolates I/O interrupts and cycle-steal operations. With bit 14 of the parameter field set to 1 (enable), this function allows I/O interrupts under the control of summary mask, and cycle-steal operations are enabled.

The parameter-field bit functions are defined as follows:

Bits 10, 11: Must be set to 0's.

Bit 12: This bit is ignored for the channel select function.

Bit 13: Must be set to 0.

Bit 14: Specifies whether channel priority interrupts and cycle-steal requests are enabled or disabled.

- Bit 14=0. Disabled.
- Bit 14=1. Enabled.

Bit 15: Must be set to 0.

Note: Pressing the Start push-button while in the stop state or executing any instruction that causes a level status block to be loaded (LEX instruction, SELB instruction, class interrupt, and so on) returns priority interrupt masking to program control. Also, the following operations cause cycle-stealing to be resumed:

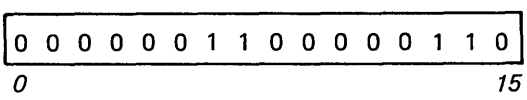
- *Setting or resetting stop-on-address mode.*
- *Performing the EN, DIS, SESR, or CPSR instruction.*

Set System ID

The set system ID function is selected by parameter field bit 13 being set to 1.

This function sets the system ID into register 0 of the current priority level. The system ID for the processor is hex 0306.

Register 0 is set as follows:



Note: When this function is selected, all other parameter-field functions are ignored.

Indicators

Indicators are not changed by the DIAG instruction; however, the data in local storage registers may be changed. Refer to "Local Storage Register Select" earlier in this chapter.

Program-Check Condition

The DIAG instruction is a privileged instruction. If this instruction is encountered during the problem state, the instruction is suppressed, a program-check interrupt occurs, and the privilege violate bit (bit 2) in the PSW is set to 1.

Machine Status Changes

The contents of general purpose registers 0 and 1 may change when the Diagnose instruction is executed with the storage select function active (Storage Select Read). The contents of a local storage register may change when the local store or storage-to-register function is selected.

Microdiagnostic Test Function

Parameter-field bit 15 is set to 1 and is used in conjunction with the microdiagnostic flag bit and the code source bits in the microcode flag register. Use of this function allows execution of the Series/1 instruction under the control of microdiagnostics. Series/1 instructions are placed in main storage by the microdiagnostic code with a microdiagnostic instruction at the end of the instruction sequence. The parameter-field bit 15 is set to 1 to allow a return to the microdiagnostic code to continue testing.

Appendix A. Instruction Execution Times

Several factors have an effect on instruction throughput. Some of these factors are:

- Instruction mix
- Storage refresh
- Channel load interference
- Wait state
- Translator enabled or disabled
- Synchronization of the channel interface to the storage interface.

Execution times for individual instructions can be calculated from the following information:

- Figure A-1—additional time for register/storage instructions
- Figure A-2—assembler syntax for address mode
- Figure A-3—additional time for storage/storage instructions
- Figure A-4—non-floating-point instructions
- Figure A-5—floating-point instructions.

The symbols used in the remainder of this appendix are defined as follows:

Symbol	Meaning
N	Number of bytes moved, filled, scanned or compared.
NS	Number of shifts
RS	Additional addressing-mode time for register/storage instructions
SS	Additional addressing-mode time for storage/storage instructions
*	Indirect address

Minimum times are shown for doubleword operations where the effective address is on an even-word boundary. Effective addresses on odd-word boundaries are from 0.12—0.36 microsecond slower.

- RS—the additional time for register/storage instructions

AM	RB	Time (usec)	Floating-point time (usec)
00	Any value	0.00	0.00
01	Any value	0.00	0.00
10	0	0.00	0.00
10	Not 0	0.12	0.12
11	0	0.60	0.36
11	Not 0	0.72	0.72

Figure A-1. Register/Storage Instructions

- Assembler syntax for address modes

  Register/storage instructions use assembler syntax *addr4* for address mode (AM). Storage/storage instructions use assembler syntax *addr5* for operand 1 (AM1) and *addr4* for operand 2 (AM2).

addr4	addr5	AM, AM1, or AM2
(reg^{0-3})	(reg)	00
(reg^{0-3})+	(reg)+	01
addr	addr	10
(reg^{1-3},waddr)	(reg^{1-7},waddr)	10
addr*	addr*	11 RB=0
displ(reg^{1-3},disp2)*	displ(reg^{1-7},disp2)*	11 RB≠0
disp(reg^{1-3})*	disp(reg^{1-7})	11 RB≠0
(reg^{1-3})*	(reg^{1-7})*	11 RB≠0
(reg^{1-3},disp)*	(reg^{1-7},disp)*	11 RB≠0

Figure A-2. Assembler Syntax for Address Mode

- SS—the additional time for storage/storage instructions

AM bits 10–13	Generation time (usec)	
0 0 0 0	0.00	
0 0 0 1	0.00	
0 0 1 0	0.00	
0 0 1 1	0.48	RB2=0
0 0 1 1	0.96	RB2≠0
0 1 0 0	0.00	
0 1 0 1	0.00	
0 1 1 0	0.00	
0 1 1 1	0.48	RB2=0
0 1 1 1	0.96	RB2≠0
1 0 0 0	0.00	RB1=0
1 0 0 0	0.12	RB1≠0
1 0 0 1	0.00	RB1=0
1 0 0 1	0.12	RB1≠0
1 0 1 0	0.24	RB1=0, RB2=0
1 0 1 0	0.36	RB1≠0, RB2=0
1 0 1 0	0.36	RB1=0, RB2≠0
1 0 1 0	0.48	RB1≠0, RB2≠0
1 0 1 1	0.84	RB1=0, RB2=0
1 0 1 1	0.96	RB1≠0, RB2=0
1 0 1 1	0.96	RB1=0, RB2≠0
1 0 1 1	1.14	RB1≠0, RB2≠0
1 1 0 0	0.36	RB1=0
1 1 0 0	1.02	RB1≠0
1 1 0 1	0.36	RB1=0
1 1 0 1	1.02	RB1≠0
1 1 1 0	0.72	RB1=0, RB2=0
1 1 1 0	1.14	RB1≠0, RB2=0
1 1 1 0	0.84	RB1=0, RB2≠0
1 1 1 0	1.14	RB1≠0, RB2≠0
1 1 1 1	1.32	RB1=0, RB2=0
1 1 1 1	1.62	RB1≠0, RB2=0
1 1 1 1	1.44	RB1=0, RB2≠0
1 1 1 1	1.68	RB1≠0, RB2≠0

Figure A-3. Storage/Storage Instructions

Figure A-4 lists the assembler instructions, except floating-point, in alphabetical sequence by mnemonic:

Mnemonic	Instruction name	Syntax	Execution time (usec)
AA	Add Address	raddr,reg[,reg] raddr,addr4	0.24 0.90 + RS
AB	Add Byte	reg,addr4 addr4,reg	0.90 + RS 0.72 + RS
ABI	Add Byte Immediate	byte,reg	0.24
ACY	Add Carry Register	reg	0.24
AD	Add Doubleword	reg,addr4 addr4,reg addr5,addr4	0.90 + RS See Note 1. 0.72 + RS See Note 1. 1.38 + SS
ARIB	Address Resolution with Indirect Branch	disp1,disp2, Tbladdr addr,addr index,addr addr,index index,index addr,immed index,immed	 5.40 6.42 6.42 7.50 5.52 6.60
ARIBOFF	Address Resolution with Indirect Branch Off		0.78
ARIBON	Address Resolution with Indirect Branch On		0.78
AW	Add Word	reg,reg reg,addr4 addr4,reg longaddr,reg longaddr*,reg addr5,addr4	0.24 0.90 + RS 0.60 + RS 0.60 if RB=0; 0.72 if RB≠0 1.08 1.38 + SS
AWCY	Add Word with Carry	reg,reg	0.24
AWI	Add Word Immediate	word,reg[,reg] word,addr4	0.24 0.90 + RS
B	Branch Unconditional	longaddr longaddr*	0.60 if R2=0; 0.72 if R2≠0 1.08 if R2=0; 1.56 if R2≠0
BAL	Branch and Link	longaddr,reg longaddr*,reg	0.60 if R2=0; 0.72 if R2≠0 1.08 if R2=0; 1.56 if R2≠0
BALS	Branch and Link Short	(reg,jdisp)* (reg)* addr*	1.08 1.08 1.08

Figure A-4 (Part 1 of 14). Instruction Execution Times

Mnemonic	Instruction name	Syntax	Execution time (usec)
BALX	Branch and Link External	vcon,reg	Same as BAL
BC	Branch on Condition	cond,longaddr cond,longaddr*	0.18 no branch; 0.66 if R2=0 or 0.78 if R2≠0 branch 0.18 no branch; 1.14 if R2=0 or 1.62 if R2≠0 branch
BCC	Branch on Condition Code	cond,longaddr cond,longaddr*	0.18 no branch; 0.66 if R2=0 or 0.78 if R2≠0 branch 0.18 no branch; 1.14 if R2=0 or 1.62 if R2≠0 branch
BCY	Branch on Carry	longaddr	Same as BC
BE	Branch on Equal	longaddr	Same as BC
BER	Branch on Error	longaddr	Same as BC
BEV	Branch on Even	longaddr	Same as BC
BGE	Branch on Arithmetically Greater Than or Equal	longaddr	Same as BC
BGT	Branch on Arithmetically Greater Than	longaddr	Same as BC
BLE	Branch on Arithmetically Less Than or Equal	longaddr	Same as BC
BLGE	Branch on Logically Greater Than or Equal	longaddr	Same as BC
BLGT	Branch on Logically Greater Than	longaddr	Same as BC
BLLE	Branch on Logically Less Than or Equal	longaddr	Same as BC
BLLT	Branch on Logically Less Than	longaddr	Same as BC
BLT	Branch on Arithmetically Less Than	longaddr	Same as BC
BMIX	Branch if Mixed	longaddr	Same as BC
BN	Branch on Negative	longaddr	Same as BC
BNC	Branch on Not Condition	cond,longaddr cond,longaddr*	Same as BC
BNCC	Branch on Not Condition Code	cond,longaddr cond,longaddr*	Same as BC
BNCY	Branch on No Carry	longaddr	Same as BC

Figure A-4 (Part 2 of 14). Instruction Execution Times

Mnemonic	Instruction name	Syntax	Execution time (usec)
BNE	Branch on Not Equal	longaddr	Same as BC
BNER	Branch if Not Error	longaddr	Same as BC
BNEV	Branch if Not Even	longaddr	Same as BC
BNMIX	Branch if Not Mixed	longaddr	Same as BC
BNN	Branch if Not Negative	longaddr	Same as BC
BNOFF	Branch if Not Off	longaddr	Same as BC
BNON	Branch if Not On	longaddr	Same as BC
BNOV	Branch on Not Overflow	cond,longaddr cond,longaddr*	Same as BC
BNP	Branch on Not Positive	longaddr	Same as BC
BNZ	Branch on Not Zero	longaddr	Same as BC
BOFF	Branch if Off	longaddr	Same as BC
BON	Branch if On	longaddr	Same as BC
BOV	Branch on Overflow	cond,longaddr cond,longaddr*	Same as BC
BP	Branch on Positive	longaddr	Same as BC
BX	Branch External	vcon	Same as B
BXS	Branch Indexed Short	(reg^{1-7},jdisp) (reg^{1-7}) addr	0.72 0.72 0.72
BZ	Branch on Zero	longaddr	Same as BC
CA	Compare Address	raddr,reg raddr,addr4	0.24 0.66 + RS
CB	Compare Byte	addr4,reg addr5,addr4	0.60 + RS 1.08 + SS
CBI	Compare Byte Immediate	byte,reg	0.36
CD	Compare Doubleword	addr4,reg addr5,addr4	0.72 + RS See Note 1. 1.08 + SS
CFED	Compare Byte Field Equal and Decrement	(reg),(reg)	0.66 if count is 0. 2.88 + (N-1)(1.68) if count≠0 and is not terminated before R=0.
CFEN	Compare Byte Field Equal and Increment	(reg),(reg)	Same as CFED

Figure A-4 (Part 3 of 14). Instruction Execution Times

Mnemonic	Instruction name	Syntax	Execution time (usec)
CFNED	Compare Byte Field Not Equal and Decrement	(reg),(reg)	Same as CFED
CFNEN	Compare Byte Field Not Equal and Increment	(reg),(reg)	Same as CFED
CMR	Complement Register	reg[,reg]	0.36
CPAKR	Copy Address Key Register	addr4 reg	0.90 + RS 0.24
CPCL	Copy Current Level	reg	0.24
CPCLK	Copy Clock	reg	0.60
CPCMP	Copy Comparator	reg	0.60
CPCON	Copy Console Data Buffer	reg	0.72
CPIMR	Copy Interrupt Mask Register	addr4	0.66 + RS
CPIPF	Copy In-Process Flags	addr4	0.66 + RS
CPISK	Copy Instruction Space Key	addr4 reg	Same as CPAKR
CPLB	Copy Level Block	reg,addr4	4.92 + RS maximum
CPLSR	Copy Level Status Register	reg	0.24
CPOOK	Copy Operand 1 Key	addr4 reg	Same as CPAKR
CPOTK	Copy Operand 2 Key	addr4 reg	Same as CPAKR
CPPSR	Copy Processor Status and Reset	addr4	1.92 + RS
CPSK	Copy Storage Key	reg,addr4	0.30
CPSR	Copy Segmentation Register	reg,addr4	1.14 + RS + (0.78)N See Note 2.
CW	Compare Word	reg,reg addr4,reg addr5,addr4	0.24 0.60 + RS 1.08 + SS
CWI	Compare Word Immediate	word,reg word,addr4	0.24 0.60 + RS
DB	Divide Byte	addr4,reg	3.24 + RS minimum 6.84 + RS maximum

Figure A-4 (Part 4 of 14). Instruction Execution Times

Mnemonic	Instruction name	Syntax	Execution time (usec)
DD	Divide Doubleword	addr4,reg	4.08 + RS minimum 11.52 + RS maximum
DIAG	Diagnose	Word Word Dblwd Dblwd (The check bits are in the high word of the doubleword and data is in the low word.)	1.20 Stg Write with ECC 1.38 Stg Read with ECC 2.70 Stg Write without ECC 2.70 Stg Read without ECC 1.98 GPR or FPR Write 1.80 GPR or FPR Read 1.80 LSR Write 1.62 LSR Read 1.92 IAR or AKR Write 1.98 IAR or AKR Read 1.62 Enable I/O Interface 1.98 Disable I/O Interface 1.02 Set Self-Modifying Code Detector 1.02 Reset Self-Modifying Code Detector 0.90 Set System ID
DIS	Disable	ubyte	1.26 when op bit 10 = 1 0.54 when op bit 12 = 1 0.54 when op bit 13 = 1 1.32 when op bit 14 = 1 0.54 when op bit 15 = 1
DW	Divide Word	addr4,reg	2.88 + RS minimum 6.48 + RS maximum
EN	Enable	ubyte	1.38 when op bit 10 = 1 1.56 when op bit 12 = 1 0.54 when op bit 13 = 1 1.38 when op bit 14 = 1 0.54 when op bit 15 = 1

Figure A-4 (Part 5 of 14). Instruction Execution Times

Mnemonic	Instruction name	Syntax	Execution time (usec)
FFD	Fill Byte Field and Decrement	reg,(reg)	0.66 when count=0. 2.64 + (N-1)(0.66) when count=1, 2, or 3. 4.38 + (N-4)(0.27) when the count is an even number greater than 3, and the address is even. 5.04 + (N-5)(0.27) when count is an odd number greater than 3, and the address is even. 4.80 + (N-4)(0.27) when count is an even number greater than 3, and the address is odd. 5.46 + (N-5)(0.27) when count is an odd number greater than 3, and the address is odd.
FFN	Fill Byte Field and Increment	reg,(reg)	0.66 when count=0. 2.64 + (N-1)(0.66) when count=1, 2, or 3. 4.56 + (N-4)(0.27) when count is an even number greater than 3, and the address is even. 5.22 + (N-5)(0.27) when count is an odd number greater than 3, and the address is even. 4.26 + (N-4)(0.27) when count is an even number greater than 3, and the address is odd. 4.92 + (N-5)(0.27) when count is an odd number greater than 3, and the address is odd.
IO	Operate I/O	longaddr longaddr*	5.46 Read 7.11 Write 5.94 Read 7.59 Write
IOPK	Interchange Operand Keys		0.24
IR	Interchange Registers	reg,reg	0.36
J	Jump Unconditional	jdisp jaddr	0.72 0.72
JAL	Jump and Link	jdisp,reg jaddr,reg	0.72 0.72
JC	Jump on Condition	cond,jdisp cond,jaddr	0.18 no jump; 0.78 jump taken 0.18 no jump; 0.78 jump taken

Figure A-4 (Part 6 of 14). Instruction Execution Times

Mnemonic	Instruction name	Syntax	Execution time (usec)
JCT	Jump on Count	jdisp,reg jaddr,reg	1.08 if R=0 and no branch 0.78 if R=1 and no branch 1.02 branch taken 1.08 if R=0 and no branch 0.78 if R=1 and no branch 1.02 branch taken
JCY	Jump on Carry		Same as JC
JE	Jump on Equal		Same as JC
JEV	Jump on Even		Same as JC
JGE	Jump on Arithmetically Greater Than or Equal		Same as JNC
JGT	Jump on Arithmetically Greater Than		Same as JNC
JLE	Jump on Arithmetically Less Than or Equal		Same as JC
JLGE	Jump on Logically Greater Than or Equal		Same as JNC
JLGT	Jump on Logically Greater Than		Same as JNC
JLLE	Jump on Logically Less Than or Equal		Same as JC
JLLT	Jump on Logically Less Than		Same as JC
JLT	Jump on Arithmetically Less Than		Same as JC
JMIX	Jump if Mixed		Same as JC
JN	Jump on Negative		Same as JC
JNC	Jump on Not Condition	cond,jdisp cond,jaddr	0.18 no jump; 0.78 jump taken 0.18 no jump; 0.78 jump taken
JNCY	Jump on No Carry		Same as JNC
JNE	Jump on Not Equal		Same as JNC
JNEV	Jump on Not Even		Same as JNC
JNMIX	Jump if Not Mixed		Same as JNC
JNN	Jump on Not Negative		Same as JNC
JNOFF	Jump if Not Off		Same as JNC
JNON	Jump if Not On		Same as JNC

Figure A-4 (Part 7 of 14). Instruction Execution Times

Mnemonic	Instruction name	Syntax	Execution time (usec)
JNP	Jump on Not Positive		Same as JNC
JNZ	Jump on Not Zero		Same as JNC
JOFF	Jump if Off		Same as JC
JON	Jump if On		Same as JC
JP	Jump on Positive		Same as JC
JZ	Jump on Zero		Same as JC
LEX	Level Exit	[ubyte]	4.28 to wait state 5.16
LMB	Load Multiple and Branch (For each specified register from 0–6, add an additional 0.65 per register.)	R7 to R6 R7 to R5 R7 to R4 R7 to R3 R7 to R2 R7 to R1 R7 to R0 R7 only	7.02 + RS odd word address 6.42 + RS even word address 6.42 + RS odd 5.82 + RS even 5.88 + RS odd 6.18 + RS even 5.58 + RS odd 5.10 + RS even 5.10 + RS odd 5.46 + RS even 4.74 + RS odd 4.38 + RS even 4.14 + RS odd or even 3.54 + RS odd or even
MB	Multiply Byte	addr4,reg	1.56 + RS minimum 2.28 + RS maximum
MD	Multiply Doubleword	addr4,reg	1.68 + RS minimum 3.36 + RS maximum
MVA	Move Address	addr4,reg raddr,addr4	0.24 + RS 0.24 + RS
MVB	Move Byte	reg,addr4 addr4,reg addr5,addr4	0.54 + RS 0.60 + RS 0.90 + SS
MVBI	Move Byte Immediate	byte,reg	0.24
MVBZ	Move Byte and Zero	addr4,reg	1.02 + RS
MVD	Move Doubleword	reg,addr4 addr4,reg addr5,addr4	0.66 + RS See Note 1. 0.60 + RS See Note 1. 0.66 + SS
MVDZ	Move Doubleword and Zero	addr4,reg	1.02 + RS See Note 1.

Figure A-4 (Part 8 of 14). Instruction Execution Times

Mnemonic	Instruction name	Syntax	Execution time (usec)
MVFD	Move Byte Field and Decrement	(reg),(reg)	0.66 when count=0 2.64 + (N-1)(1.14) when count=1, 2, or 3. 5.10 + (N-4)(0.39) when count is even and greater than 3, and addresses are both even or both odd. 6.24 + (N-4)(0.39) when count is odd and greater than 3, and addresses are both even or both odd. 2.64 + (N-1)(1.14) when count is greater than 3, and one address is even and the other address is odd.
MVFN	Move Byte Field and Increment	(reg),(reg)	0.66 when count=0. 2.64 + (N-1)(1.14) when count=1, 2, or 3. 5.28 + (N-4)(0.39) when the count is even and greater than 3, and both addresses are even. 6.42 + (N-4)(0.39) when the count is odd and greater than 3, and both addresses are even. 5.04 + (N-4)(0.39) when the count is even and greater than 3, and both addresses are odd. 6.18 + (N-4)(0.39) when the count is odd and greater than 3, and both addresses are odd. 2.64 + (N-1)(1.14) when the count is greater than 3, and one address is even and the other address is odd.
MVW	Move Word	reg,reg reg,addr4 addr4,reg reg,longaddr reg,longaddr* longaddr,reg longaddr*,reg addr5,addr4	0.24 0.54 + RS 0.60 + RS 0.54 if RB=0; 0.96 if RB≠0 0.90 if RB=0; 1.26 if RB≠0 0.60 if RB=0; 0.72 if RB≠0 1.08 0.66 + SS
MVWI	Move Word Immediate	word,reg word,addr4	0.24 0.66 + RS
MVWS	Move Word Short	reg,shortaddr reg,shortaddr* shortaddr,reg shortaddr*,reg	0.54 1.02 0.72 1.08

Figure A-4 (Part 9 of 14). Instruction Execution Times

Mnemonic	Instruction name	Syntax	Execution time (usec)
MVWZ	Move Word and Zero	addr4,reg	1.02 + RS
MW	Multiply Word	addr4,reg	1.56 + RS minimum 3.24 + RS maximum
NOP	No Operation		0.54
NWI	AND Word Immediate	word,reg[,reg]	0.24
OB	OR Byte	reg,addr4 addr4,reg addr5,addr4	0.90 + RS 0.72 + RS 1.38 + SS
OD	OR Doubleword	reg,addr4 addr4,reg addr5,addr4	0.90 + RS See Note 1. 0.72 + RS See Note 1. 1.38 + SS
OW	OR Word	reg,reg reg,addr4 addr4,reg longaddr,reg longaddr*,reg addr5,addr4	0.24 0.90 + RS 0.60 + RS 0.60 if RB=0, 0.72 if RB≠0 1.08 1.38 + SS
OWI	OR Word Immediate	word,reg[,reg] word,addr4	0.24 0.90 + RS
PB	Pop Byte	addr4,reg	2.52 + RS
PD	Pop Doubleword	addr4,reg	2.52 + RS See Note 1.
PSB	Push Byte	reg,addr4	2.40 + RS
PSD	Push Doubleword	reg,addr4	2.40 + RS See Note 1.
PSW	Push Word	reg,addr4	2.28 + RS
PW	Pop Word	addr4,reg	2.40 + RS
RBTB	Reset Bits Byte	reg,addr4 addr4,reg addr5,addr4	0.90 + RS 0.72 + RS 1.92 + SS
RBTD	Reset Bits Doubleword	reg,addr4 addr4,reg addr5,addr4	0.90 + RS See Note 1. 0.72 + RS See Note 1. 1.62 + SS
RBTW	Reset Bits Word	reg,reg reg,addr4 addr4,reg longaddr,reg longaddr*,reg addr5,addr4	0.24 0.90 + RS 0.60 + RS 0.60 if RB=0; 0.72 if RB≠0 1.08 1.20 + SS

Figure A-4 (Part 10 of 14). Instruction Execution Times

Mnemonic	Instruction name	Syntax	Execution time (usec)
RBTWI	Reset Bits Word Immediate	word,reg[,reg] word,addr4	0.24 0.90 + RS
SA	Subtract Address	raddr,reg[,reg] raddr,addr4	0.24 1.20 + RS
SB	Subtract Byte	reg,addr4 addr4,reg	0.90 + RS 0.72 + RS
SBTB	Set Bits Byte	reg,addr4 addr4,reg addr5,addr4	0.90 + RS 0.72 + RS 1.92 + SS
SBTD	Set Bits Doubleword	reg,addr4 addr4,reg addr5,addr4	0.90 + RS See Note 1. 0.72 + RS See Note 1. 1.62 + SS
SBTW	Set Bits Word	reg,reg reg,addr4 addr4,reg longaddr,reg longaddr*,reg addr5,addr4	0.24 0.90 + RS 0.60 + RS 0.60 if RB=0; 0.72 if RB≠0 1.08 1.20 + SS
SBTWI	Set Bits Word Immediate	word,reg[,reg] word,addr4	0.24 0.90 + RS
SCY	Subtract Carry Indicator	reg	0.24
SD	Subtract Doubleword	reg,addr4 addr4,reg addr5,addr4	0.90 + RS See Note 1. 0.72 + RS See Note 1. 1.38 + SS
SEAKR	Set Address Key Register	addr4 reg	1.20 + RS 0.84
SECLK	Set Clock	reg	0.36
SECMP	Set Comparator	reg	0.36
SECON	Set Console Data Lights	reg	2.52
SEIMR	Set Interrupt Mask Register	addr4	0.60 + RS
SEIND	Set Indicators	reg	0.60
SEISK	Set Instruction Space Key	addr4 reg	Same as SEAKR
SELB	Set Level Block	reg,addr4	7.80 + RS minimum 8.82 + RS maximum

Figure A-4 (Part 11 of 14). Instruction Execution Times

Mnemonic	Instruction name	Syntax	Execution time (usec)
SEOOK	Set Operand 1 Key	addr4 reg	Same as SEAKR
SEOTK	Set Operand 2 Key	addr4 reg	Same as SEAKR
SESK	Set Storage Key	reg,addr4	0.24
SESR	Set Segmentation Register	reg,addr4	0.78 + N(0.90) + RS
SFED	Scan Byte Field Equal and Decrement	reg,(reg)	0.66 when count=0. 2.52 + (N-1)(1.32) when the count≠0 and is not terminated before R=0.
SFEN	Scan Byte Field Equal and Increment	reg,(reg)	Same as SFED
SFNED	Scan Byte Field Not Equal and Decrement	reg,(reg)	Same as SFED
SFNEN	Scan Byte Field Not Equal and Increment	reg,(reg)	Same as SFED
SLC	Shift Left Circular	cnt16,reg reg,reg	1.14 1.26
SLCD	Shift Left Circular Double	cnt31,reg reg,reg	0.36 1.26
SLL	Shift Left Logical	cnt16,reg reg,reg	1.92 1.62 if>16 2.04 1.74 if>15
SLLD	Shift Left Logical Double	cnt31,reg reg,reg	1.62 1.74 if>31 2.28 2.04 if>32
SLT	Shift Left and Test (number of shifts specified)	0 1 or 2 3 or 4 5 or 6 7 or 8 9 or 10 11 or 12 13 or 15 15 or 16 >16	0.66NS See Note 3. 3.48NS + A 3.66NS + A 3.90NS + A 4.08NS + A 4.32NS + A 4.50NS + A 4.44NS + A 4.50NS + A 1.10NS + A

Figure A-4 (Part 12 of 14). Instruction Execution Times

Mnemonic	Instruction name	Syntax	Execution time (usec)
SLTD	Shift Left and Test Double (number of shifts specified)	0 1 or 2 3 or 4 5 or 6 7 or 8 9 or 10 11 or 12 13 or 15 15 or 16 17 or 18 19 or 20 21 or 22 23 or 24 25 or 26 27 or 28 29 or 30 31 32 >32	0.66NS See Note 3. 4.38NS + A 4.56NS + A 4.80NS + A 4.98NS + A 5.22NS + A 5.40NS + A 5.36NS + A 5.40NS + A 4.80NS + A 4.98NS + A 5.22NS + A 5.40NS + A 5.64NS + A 5.82NS + A 5.76NS + A 5.82NS + A 6.12NS 1.62NS
SRA	Shift Right Arithmetic	cnt16,reg reg,reg	1.14 1.38 if>15 1.26 1.50 if>15
SRAD	Shift Right Arithmetic Double	cnt31,reg reg,reg	0.36 1.26 1.50 if>31
SRL	Shift Right Logical	cnt16,reg reg,reg	1.14 1.20 if>15 1.26 1.32 if>15
SRLD	Shift Right Logical Double	cnt31,reg reg,reg	0.36 1.26 1.32 if>31
STM	Store Multiple (For each specified register from 0–6, add an additional 0.65 per register.)	R7 to R6 R7 to R5 R7 to R4 R7 to R3 R7 to R2 R7 to R1 R7 to R0 R7 only	5.64 + RS odd word address 6.18 + RS even word address 5.64 + RS odd 5.28 + RS even 5.04 + RS odd 5.40 + RS even 4.86 + RS odd 4.68 + RS even 4.44 + RS odd 4.62 + RS even 4.02 + RS odd 3.96 + RS even 3.72 + RS odd or even 3.18 + RS odd or even
STOP	Stop	[ubyte]	0.54

Figure A-4 (Part 13 of 14). Instruction Execution Times

Mnemonic	Instruction name	Syntax	Execution time (usec)
SVC	Supervisor Call	ubyte	6.24
SW	Subtract Word	reg,reg reg,addr4 addr4,reg longaddr,reg longaddr*,reg addr5,addr4	0.24 0.90 + RS 0.60 + RS 0.60 if RB=0; 0.72 if RB≠0 1.08 1.38 + SS
SWCY	Subtract Word with Carry	reg,reg	0.24
SWI	Subtract Word Immediate	word,reg[,reg] word,addr4	0.24 0.90 + RS
TBT	Test Bit	(reg,bitdisp)	0.72
TBTR	Test Bit and Reset	(reg,bitdisp)	1.14
TBTS	Test Bit and Set	(reg,bitdisp)	1.14
TBTV	Test Bit and Invert	(reg,bitdisp)	1.14
TWI	Test Word Immediate	word,reg word,addr4	0.48 0.72 + RS
VR	Invert Register	reg[,reg]	0.24
XB	Exclusive OR Byte	reg,addr4 addr4,reg	0.90 + RS 0.72 + RS
XD	Exclusive OR Doubleword	reg,addr4 addr4,reg	0.90 + RS See Note 1. 0.72 + RS See Note 1.
XW	Exclusive OR Word	reg,reg reg,addr4 addr4,reg longaddr,reg longaddr*,reg	0.24 0.90 + RS 0.60 + RS 0.60 if RB=0; 0.72 if RB≠0 1.08
XWI	Exclusive OR Word Immediate	word,reg[,reg]	0.24

Figure A-4 (Part 14 of 14). Instruction Execution Times

Notes:

1. *Minimum times are shown for doubleword operations where the effective address is on as even-word boundary. Effective addresses on odd-word boundaries are from 0.12–0.36 microsecond slower.*

2. *N is the number of segmentation registers set or copied.*

3. *A = 0.12NS if the result high-order bit in the shifted register has a value of 1. Otherwise, A=0. NS = Number of shifts done.*

Figure A-5 lists all floating-point instructions in alphabetic sequence by mnemonic:

Mnemonic	Instruction name	Syntax	Execution time (usec)
CPFLB	Copy Floating Level Block	freg,addr4	11.10 + RS
FA	Floating Add	addr4,freg	3.78 + RS minimum 5.52 + RS typical 8.40 + RS maximum
		freg,freg	3.12 minimum 4.98 typical 7.74 maximum
FAD	Floating Add Double	addr4,freg	5.76 + RS minimum 8.10 + RS typical 12.06 + RS maximum
		freg,freg	5.10 minimum 7.56 typical 11.16 maximum
FC	Floating Compare	freg,freg	3.24 minimum 5.58 typical 8.10 maximum
FCD	Floating Compare Double	freg,freg	5.10 minimum 7.68 typical 11.52 maximum
FD	Floating Divide	addr4,freg	3.90 + RS minimum 29.30 + RS typical 30.85 + RS maximum
		freg,freg	3.36 minimum 28.80 typical 30.30 maximum
FDD	Floating Divide Double	addr4,freg	5.36 + RS minimum 103.40 + RS typical 106.80 + RS maximum
		freg,freg	4.80 minimum 102.90 typical 106.30 maximum
FM	Floating Multiply	addr4,freg	3.54 + RS minimum 12.78 + RS typical 16.50 + RS maximum
		freg,freg	3.42 minimum 12.66 typical 16.38 maximum

Figure A-5 (Part 1 of 2). Floating-Point Instruction Execution Times

Mnemonic	Instruction name	Syntax	Execution time (usec)
FMD	Floating Multiply Double	addr4,freg	4.38 + RS minimum 33.30 + RS typical 54.30 + RS maximum
		freg,freg	4.14 minimum 33.00 typical 54.00 maximum
FMV	Floating Move	addr4,freg freg,freg freg,addr4	0.96 + RS 0.96 0.90 + RS
FMVC	Floating Move and Convert	addr4,freg	1.26 + RS mimimum 2.10 + RS typical 2.24 + RS maximum
		freg,addr4	1.92 + RS minimum 3.12 + RS typical 3.72 + RS maximum
FMVCD	Floating Move and Convert Double	addr4,freg	1.38 + RS minimum 3.54 + RS typical 3.78 + RS maximum
		freg,addr4	2.64 + RS minimum 4.92 + RS typical 5.28 + RS maximum
FMVD	Floating Move Double	addr4,freg freg,freg freg,addr4	1.56 + RS 1.08 1.68 + RS
FS	Floating Subtract	addr4,freg	3.78 + RS minimum 6.00 + RS typical 8.64 + RS maximum
		freg,freg	3.12 minimum 5.46 typical 7.98 maximum
FSD	Floating Subtract Double	addr4,freg	5.76 + RS minimum 8.22 + RS typical 12.42 + RS maximum
		freg,freg	5.10 minimum 7.68 typical 11.52 maximum
SEFLB	Set Floating Level Block	addr4,freg	11.22 + RS minimum 12.66 + RS maximum

Figure A-5 (Part 2 of 2). Floating-Point Instruction Execution Times

Appendix B. Software Notes

1. Instruction streams that are self-modifying cannot be guaranteed. Refer to the following examples.

   For the 4956 Model J00, an executing instruction cannot modify the next sequential instruction stream word.

   Example:

```
        MVWI    hex 5000,LOC1
LOC1    J       THERE
```

   This example illustrates a self-modifying instruction stream. The Move Word Immediate (MVWI) instruction moves a hex 5000 to LOC1. A hex 5000 in machine code is a no-operation instruction (NOP). The program is attempting to execute a NOP instruction, instead of the Jump (J) instruction.

   For the 4956 Model K00, an executing instruction cannot modify the next sequential three instruction stream words.

   Example:

```
        R1=0000
        R2=LOC1
        R7=6

        FFN     R1,(R2)
LOC1    MVBI    2,R3
LOC2    SECON   R3
LOC3    LEX
```

   This example illustrates a self-modifying instruction stream. The Fill Field and Increment (FFN) instruction moves 0000 to LOC1 through LOC3. A 0000 in machine code is an Add Byte Zero to Register Zero (ABI). The program is attempting to execute the ABI instruction instead of the sequence MVBI, SECON and LEX. This type of programming is not permitted on the 4956 Model K00.

2. Four priority interrupt levels (0–3) are implemented in the processor. A Prepare command to levels 4–15 is executed so that condition code reporting occurs; however, the Prepare command is not executed at the addressed device and effectively results in a no-operation.

3. There is no storage-protect feature in the 4956 processor. Execution of the Set Storage Key (SESK) and Copy Storage Key (CPSK) instructions results in a program check.

4. Byte write operations to storage locations that contain two bit errors are not executed by the storage card. The storage location must first be corrected with a word write before the byte write operation can be executed correctly.

   Note: After a successful power-on reset, all storage locations are initialized to some value.

Index

A

B

C

D

E

IBM Series/1 4956 Processor Model K00
Description

Order No. GA34-0851-0

READER'S COMMENT FORM

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you. Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

Note: Staples can cause problems with automated mail sorting equipment. Please use pressure sensitive or other gummed tape to seal this form.

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)

GA34-0851-0
Printed in U.S.A.

Reader's Comment Form

Cut or Fold Along Line

Fold and tape | Please Do Not Staple | Fold and tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

International Business Machines Corporation
Information Development, Department 28E
5408 (Internal Zip)
P.O. Box 1328
Boca Raton, Florida 33429-9960

Fold and tape | Please Do Not Staple | Fold and tape

IBM®

International Business Machines Corporation

GA34-0851-00

GA34-0851-0
Printed in U.S.A.