2

STARAN S

Reference Manual

a new way of thinking

Staran

GOODYEAR AEROSPACE CORPORATION AKRON, OHIO 44315

16 January 1973

To: Distribution

Subject: STARAN S Reference Manual - GER-15636

The subject manual has been upgraded. The attached pages are forwarded for inclusion in your copy. Please substitute the changed pages for the existing pages.

L. P. Arkwright

LPA:mb

Price $25.00

GOODYEAR AEROSPACE
CORPORATION
AKRON, OHIO 44315

STARAN S REFERENCE MANUAL

GER 15636
Change 1

JUNE 1972
JANUARY 1973

NOTICE

This document contains material generated by Goodyear Aerospace Corporation and is transmitted for the purpose of aiding the transaction of business between Goodyear Aerospace Corporation and the recipient. It is understood that the material contained herein will not be used, copied, or disclosed to others, without specific written consent of Goodyear Aerospace Corporation.

DESIGN UPGRADING

The STARAN S Computer System will continue to be improved and upgraded. Interested parties should contact Goodyear Aerospace Corporation, Computer Division Marketing, Akron, Ohio 44315, Telephone: (216) 794-3631 for information regarding STARAN S design improvements.

LIST OF EFFECTIVE PAGES

Insert latest changed pages and dispose of superseded pages.

NOTE: On a changed page, the information affected by the latest change is indicated by a vertical line in the outer margin of the page. A zero in the change number column indicates an original page.

The total number of pages in this manual is 157, consisting of the following:

Page No.	Change Number
Title	1
A, B	1
i - ix	0
x	1
xi, xii	0
1-1 - 1-3	0
1-4	1
1-5, 1-6	0
1-7	1
1-8 - 1-10	0
1-11	1
1-12 - 1-14	0
1-15	1
1-16 - 1-30	0

LIST OF EFFECTIVE PAGES CONTINUED

Page No.	Change Number
2-1	0
2-2, 2-3	1
2-4	0
2-5	1
2-6 - 2-9	0
2-10	1
2-11, 2-12	0
2-13	1
2-14	0
2-15	1
2-16	0
2-17, 2-18	1
2-19, 2-20	0
2-21, 2-22	1
2-23 - 2-25	0
2-26	1
2-27 - 2-29	0
2-30	1
2-31 - 2-38	0
2-39	1
2-40 - 2-48	0
2-49	1
2-50 - 2-57	0
3-1 - 3-4	0
Ai	0
A1	1
A2 - A6	0
A7	1
A8, A9	0
A10	1
Bi, B1 - B11	0
Ci, C1 - C3	0
Di, D1 - D3	0
D4	1
D5	0
Ei, E1 - E4	0
Fi, F1	0
Gi, G1 - G2	0
X-1 - X-8	0

NOTE - This document supersedes GER-15386 and GER-15592.

TABLE OF CONTENTS

CHAPTER TITLE PAGE

TABLE OF CONTENTS

CHAPTER TITLE PAGE

1 (cont)

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

FOREWORD

MANUALS

The STARAN S Reference Manual is one of five standard manuals for the STARAN S computer system. As a composite group, the manuals provide the information necessary for programming, operating, and maintaining the STARAN S.

The titles and publication numbers of the manuals are as follows:

Title	Publication
STARAN S Reference Manual	GER-15636
STARAN S APPLE Programming Manual	GER-15637
STARAN S Operator's Guide	GER-15638
STARAN S System Programmer's Reference Manual	GER-15639
STARAN S Maintenance Manual	GER-15640

REFERENCE MANUAL

The STARAN S Reference Manual provides the user with a brief description of the STARAN S hardware characteristics and a detailed guide for machine language programming.

STARAN S COMPUTER SYSTEM

STARAN S, the first truly new architecture for computer systems to be implemented in the last 20 years, is the result of over a decade of intensive development effort in associative and parallel processing at Goodyear Aerospace. STARAN S, the first Associative Processor (AP) to go into production, can operate independently as a stand-alone system or in a hybrid system to complement a conventional computer (host computer).

STARAN S is most efficient when performing tasks requiring parallel operations on many sets of data simultaneously, and is less efficient when performing sequential operations. STARAN S is designed to interface easily with a host computer to provide highly efficient operations for both sequential and parallel processing operations.

FEATURES

Associative Arrays

The key element of the STARAN S computer system is the Associative Array, which provides content addressability and parallel processing capabilities.

Each array consists of 65,536 bits organized as a matrix of 256 words by 256 bits — with parallel access in either the word or bit direction.

FEATURES

Associative Arrays (cont)

Each array contains 256 response store elements that provide a means of operating on many sets of data simultaneously to perform search, logical, and arithmetic operations.

A basic STARAN S system contains one associative array. Up to 32 arrays can be included in a single STARAN S installation.

Associative Processor Control

The AP control performs data manipulations within the Associative Arrays as directed by instructions stored in AP control memory.

Associative Processor Control Memory

AP control memory contains high-speed page memories and a High-Speed Data Buffer (HSDB) to provide fast access to data and instructions that require frequent access and/or fast execution.

AP control memory also contains a slower bulk core memory for program storage.

A block of AP control memory addresses are reserved for Direct Memory Access (DMA) to a host computer.

A basic STARAN S Control Memory consists of:

1) Three page memories, each containing 512 32-bit words
2) One HSDB containing 512 32-bit words
3) A bulk core memory containing 16,384 32-bit words
4) A block of 30,720 addresses reserved for DMA

Program Pager

The ProgramPager moves program segments, which require fast execution, from bulk core to the page memories.

Sequential Controller And Memory

The Sequential Controller (SC) provides offline capabilities for assembling and debugging STARAN S programs, a communication link between STARAN S and the operator, and control for diagnostic and test programs.

The basic system contains 8192 16-bit words of Sequential Control Memory.

External Function Logic

External Function logic enables an element of STARAN S to control and interrogate the status of other elements. An external function code may be issued by AP control, the program pager, sequential control, and the host computer.

FOREWORD

FEATURES

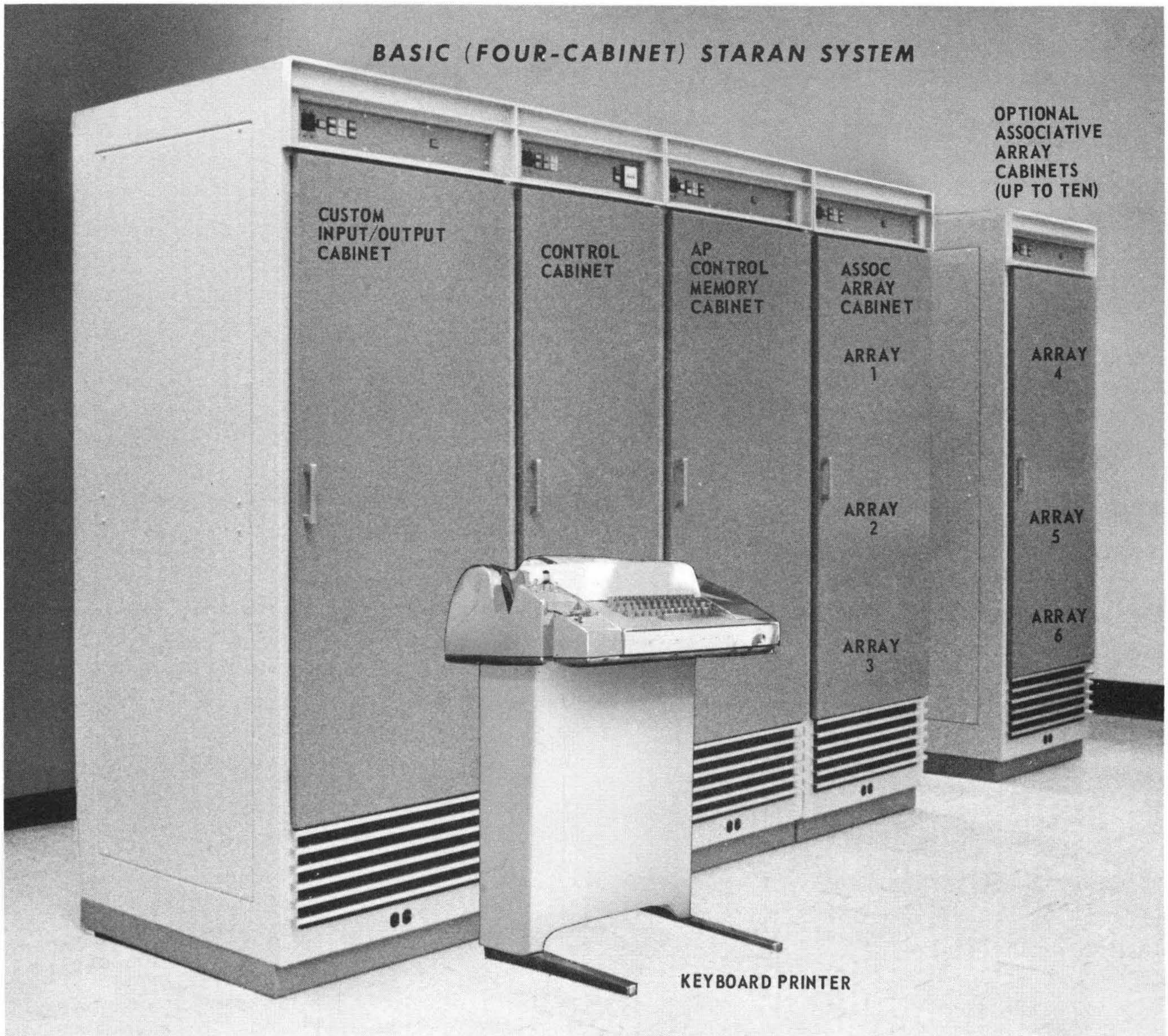
Input/Output

The following Input/Output variations are provided on the STARAN S system:

1) Direct Memory Access to a host computer
2) Buffered input/output channel 32 bits wide
3) External Function channel to pass function codes between the STARAN S control elements
4) Parallel input/output channel for each array that is 256 bits wide

PHYSICAL DESCRIPTION

The basic STARAN S computer system consists of four standard-size cabinets along with a free-standing keyboard printer. The four cabinets are: Custom input/output (I/O), Control, AP control memory, and Associative array. Up to three array modules can be included in the associative array cabinet. Since an expanded STARAN S configuration can contain up to 32 associative array modules, a fully expanded system consists of 10 array cabinets in addition to the four cabinets in the basic system, 14 cabinets in all.


BASIC (FOUR-CABINET) STARAN SYSTEM
OPTIONAL ASSOCIATIVE ARRAY CABINETS (UP TO TEN)
CUSTOM INPUT/OUTPUT CABINET
CONTROL CABINET
AP CONTROL MEMORY CABINET
ASSOC ARRAY CABINET
ARRAY 1
ARRAY 2
ARRAY 3
ARRAY 4
ARRAY 5
ARRAY 6
KEYBOARD PRINTER

STARAN S COMPUTER SYSTEM

CHAPTER I

STARAN ARCHITECTURE

SYSTEM ARCHITECTURE

The STARAN*S system introduces a new concept in computers, designed to achieve very high processing rates economically. Figure 1-1 shows a block diagram of the STARAN S computer. Each block of the diagram is discussed briefly in the following paragraphs. More detailed discussions are presented in subsequent areas of this manual.

ASSOCIATIVE PROCESSOR CONTROL MEMORY

The main function of the Associative Processor (AP) control memory is to store the assembled AP application programs. The control memory can also store items of data and act as a buffer between AP control and other elements of STARAN S. The AP control memory is divided into several memory blocks. Some memory blocks are small and fast to enable AP control to fetch its instructions rapidly (page memories and the high-speed data buffer); others are large and slower to ecomically store the entire control program (bulk core memory).

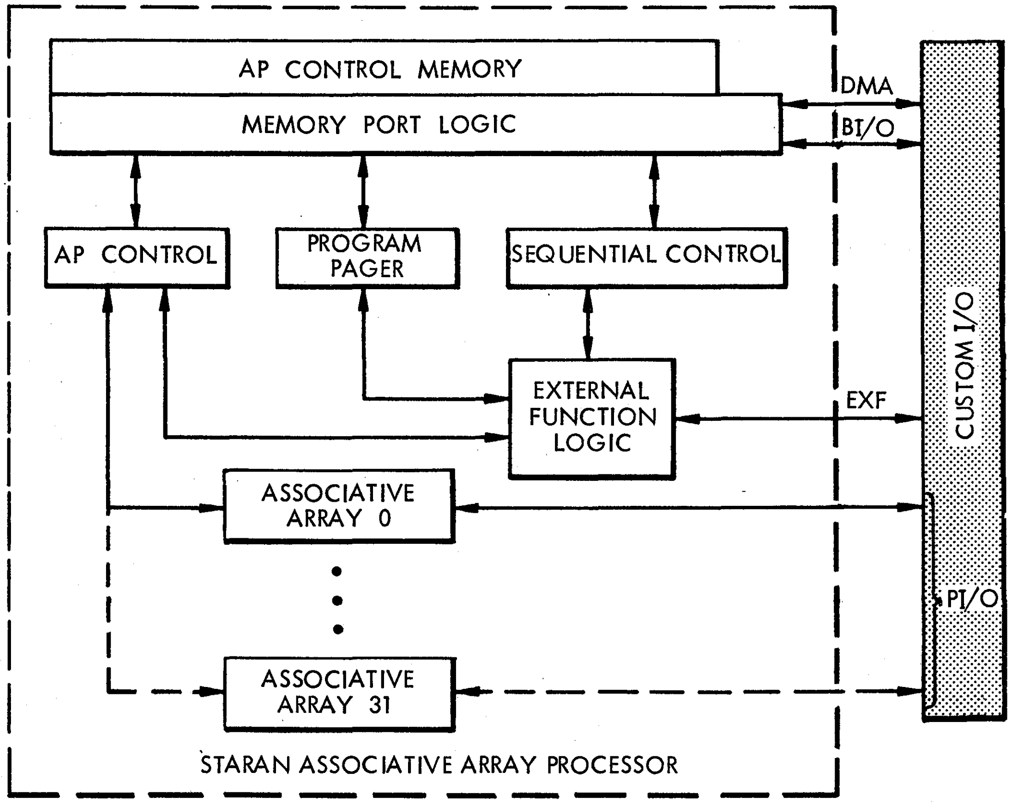


Figure 1-1. STARAN S Block Diagram

*TM, Goodyear Aerospace Corporation, Akron, Ohio 44315

ASSOCIATIVE PROCESSOR CONTROL

AP control is directly responsible for manipulation of data within the associative arrays under control of the program contained within the AP control memory. Associative operations are coordinated and controlled by AP control. Extensive interrupt and interlock capability in AP control allow the processing capability of the arrays to be applied quickly to an immediate requirement.

ASSOCIATIVE ARRAY MEMORY

The associative arrays are the heart of the STARAN S system. There may be a maximum of 32 arrays in one system. Each associative array contains a multidimensional-access (MDA) memory with 65,536(2^{16}) bits of storage. A response-store element in each associative array allows array data to be searched, restructured, and processed at a fast rate, 256 bits at a time. The array memory is arranged in a 256-bit by 256-bit square, with loading and storing permitted in either the vertical or horizontal direction and with a maximum of 256 bits transferred in a single operation. The multidimensional-access feature allows accessing of data to take place in either dimension by simply changing a bit in the associative instruction format.

PROGRAM PAGER

The program pager loads the high-speed page memories from the lower speed bulk core memory of AP control memory. The pager performs these transfer functions independently of AP control, so that while the AP control is executing a program segment out of one page memory, the pager can be loading another page memory with a future program segment.

EXTERNAL FUNCTION LOGIC

External function (EXF) logic enables the AP control, sequential control, or an external device to control the STARAN S operation. The external function logic facilitates coordination among the different STARAN S elements, provides for special functions, and simplifies housekeeping, maintenance, and test functions. By issuing external function codes to the EXF logic, a STARAN S element can interrogate and control the status of the other elements.

SEQUENTIAL CONTROL

The sequential control portion of STARAN S consists of a sequential processor (SP) having an 8K memory, a keyboard-printer, a perforated tape reader/punch unit, and logic capability to interface the sequential processor with other STARAN S elements. Sequential control is used for system software programs such as assembler, operating system, diagnostic programs, debugging, and housekeeping routines.

INPUT/OUTPUT VARIATIONS

STARAN S has a variety of input/output (I/O) options available. A custom I/O cabinet containing buffered and/or unbuffered I/O channels to data gathering, data receiving, and data-storing devices may be obtained as part of the basic STARAN S system. STARAN S can also be integrated with a variety of other computer systems. A direct memory access (DMA) channel to a host-computer memory enables a rapid interchange of data between the systems in the common memory bank. A buffered I/O channel provides an alternate means of exchanging data, while an external function channel permits interrupts and/or other control information to be passed between the two systems.

An optional parallel input/output (PI/O) channel, with a width of up to 256 bits per array, can also be implemented in STARAN S. The extreme width of this channel (up to 8, 192 bits), plus its submicrosecond cycle time, gives STARAN S an I/O bandwidth many times wider than that of a conventional computer. This PI/Ochannel can easily accommodate the high data rates that arise in many real-time applications. Also, it is possible for STARAN S to connect with special high-bandwidth mass-storage devices, permitting rapid retrieval, restructuring, and processing of data in a large data base.

AP CONTROL MEMORY

DESCRIPTION

General

The main function of the AP control memory is to store the assembled AP application programs. (See figure 1-2) AP control memory can also be used for data storage and to act as a buffer between AP control and other elements of STARAN S. Since the AP control memory is not an integral part of the associative arrays, AP control can overlap the AP control memory cycle time with the associative array cycle time.

AP control memory is divided into several memory blocks. Three small, fast memory blocks, called pages, contain the current (active) AP program segments; the larger and slower bulk core memory block contains the remainder of the AP program. A program pager is included in STARAN S to facilitate transfer from the slow to the fast memory blocks.

Each word of AP control memory contains 32 bits of either data or instructions. Bit 0 is the left (most-significant) bit, and bit 31 is the right (least-significant) bit of each word. Each word is given a 16-bit address expressed in hexadecimal notation.

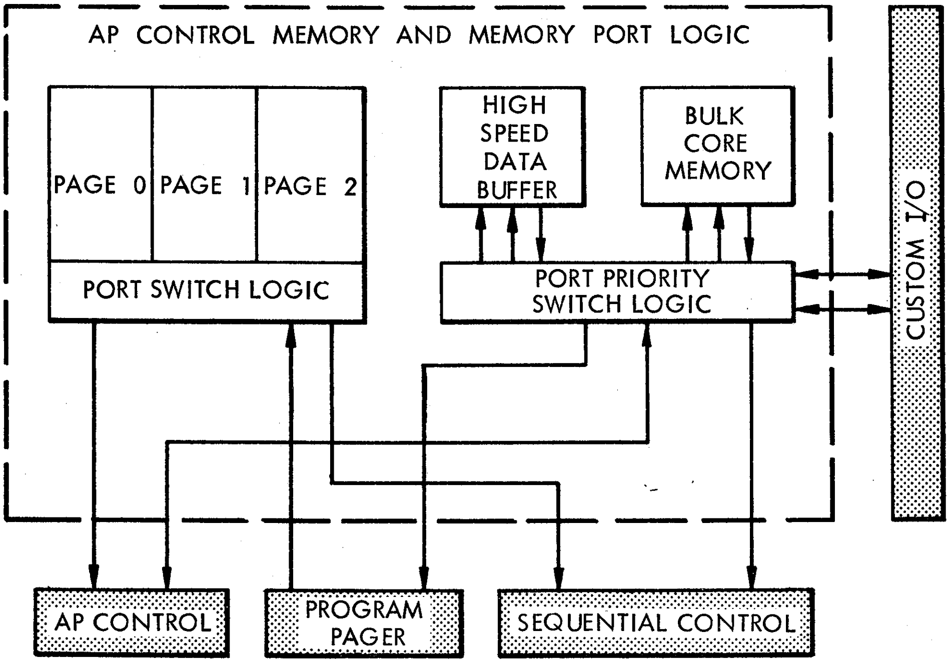


Figure 1 - 2. AP Control Memory Block Diagram

Page Memories

Three page memories are included in the AP control memory: page 0, page 1, and page 2. Page memories use fast, bipolar, solid-state elements that are volatile. Each page contains 512 words in the standard STARAN S configuration. The page memories can be doubled to 1024 words each on an optional basis. In the standard configuration, page 0 contains hexadecimal addresses 000 through 1FF; page 1 contains hexadecimal addresses 200 through 3FF; and page 2 contains hexadecimal addresses 400 through 5FF.

Page 0 may contain a library of microprograms that require fast execution, such as arithmetic subroutines. Pages 1 and 2 can be used in ping-pong fashion, with the AP control reading instructions out of one page while the other is being loaded by the program pager. This permits the programmer to use the faster memory for certain segments of the program or the entire program if fast execution is required.

Each page memory has a port switch that connects it to one of three buses. The port switch is controlled by external function codes. At any given time, a page memory is connected to 1) the instruction bus, which allows AP control to read instructions from the page; 2) the pager bus, which allows the program pager to load the page; or 3) the sequential control bus, which allows sequential control to read items from the page. If one of these buses should try to access a word in the page memory while the port switch is set to another bus, a hangup results. Hangups, which are detected by error detectors, cause interrupts in sequential control.

High-Speed Data Buffer

The high-speed data buffer (HSDB), like the page memories, uses fast, bipolar, solid-state elements and is volatile. In the standard configuration of STARAN S it contains 512 words. As an option, its size can be doubled to 1024 words. In the standard configuration of STARAN S, the high-speed data buffer contains the hexadecimal addresses 600 through 7FF.

All buses that can access AP control memory can access the HSDB making it a convenient place to store data and instruction items that need to be accessed quickly by the different STARAN elements.

A priority port switch on the HSDB resolves any conflict among buses - each memory cycle is given to the highest priority bus requesting an HSDB address at that time, while other buses requesting HSDB addresses wait for the next memory cycle. Priorities among buses are as follows:

1) Buffered I/O bus to I/O cabinet (highest priority)
2) AP control data bus
3) AP control instruction bus
4) Program pager bus
5) Sequential control bus (lowest priority)

DESCRIPTION

Bulk Core Memory

The bulk core memory uses nonvolatile core storage. In the standard configuration it contains 16,384 words (hexadecimal address 8000 through BFFF). It is expandable to 32,768 words.

Like the high-speed data buffer (HSDB), the bulk core is accessible to all buses that can access AP control memory (through a priority port switch that gives each memory cycle to the highest priority bus requesting a bulk core address). The priorities of the buses are the same as those for the high-speed data buffer.

The bulk core memory, which is large and nonvolatile, is used for storing the AP control programs. Because the bulk core memory is slower than the page memories, it is recommended that program segments be moved into the page memories for execution. Also, since the bulk core memory is accessible to all buses having access to AP control memory, it is also useful as a buffer for data items that do not require the high-speed of the HSDB.

Direct Memory Access

A block of AP control memory addresses is reserved for the direct memory access (DMA) channel to external memory. In the standard configuration this block can contain up to 30,720 addresses (hexadecimal addresses 0800 through 7FFF). If any page memory or the high-speed data buffer is expanded, the DMA block may be reduced.

All buses accessing AP control memory can access the DMA block. A priority port switch resolves any interbus conflicts, giving each access cycle to the highest priority bus requesting a DMA address at the time. Priorities among buses are the same as those for the high-speed data buffer.

ADDRESSING

General

Each AP control memory word contains either 32 bits of data or instructions. Each word is given a 16-bit address.

Bulk Core Memory

The bulk core memory contains 16, 384 words of memory in the standard configuration and is expandable to 32, 768 words. These words are assigned hexadecimal addresses 8000 through BFFF.

Certain words in the bulk core memory are dedicated to special purposes. For the standard configuration of STARAN S, these locations, are as follows:

Hexadecimal Address	Use
8000	First AP control instruction when activated
8001	AP control interrupt 1
.	.
.	.
.	.
800F	AP control interrupt 15

When the AP control becomes active, the instruction at 8000 is the first instruction executed. It is usually a branch instruction to the beginning of the first program segment to be executed.

Page Memories

The three page memories, 512 words each, that are included in the AP control memory are designated page 0, page 1, and page 2. In the standard configuration, page 0 contains hexadecimal addresses 000 through 1FF; page 1 contains hexadecimal addresses 200 through 3FF; and page 2 contains hexadecimal addresses 400 through 5FF.

High-Speed Data Buffer

The high-speed data buffer contains 512 words of memory in the standard configuration, but can be doubled to 1024 words on an optional basis. In the standard configuration, the high-speed data buffer contains hexadecimal addresses 600 through 7FF.

- **Branch and Link Registers**

Certain words in the high-speed data buffer are dedicated to special purposes. These locations are used as branch and link registers to facilitate subroutine linkage. For the standard configuration of STARAN S, the locations are as follows:

Hexadecimal Address	Use
600	Branch and Link Register 0
601	Branch and Link Register 1
602	Branch and Link Register 2
603	Branch and Link Register 3
604	Branch and Link Register 4
605	Branch and Link Register 5
606	Branch and Link Register 6
607	Branch and Link Register 7

Direct Memory Access

A block of AP control memory addresses is reserved for the direct memory access (DMA) channel to access the memory of a host computer. This block can currently contain up to 30,720 addresses, which are assigned hexadecimal addresses 0800 through 7FFF. This block may be reduced if page memories or the high-speed data buffer is increased in size.

ASSOCIATIVE PROCESSOR CONTROL MEMORY SUMMARY

Table 1-1 summarizes the AP control memory characteristics of the standard configuration. The characteristics of each memory and the connection of each bus to each section are given.

Table 1-1. AP Control Memory Characteristics

ITEM		MEMORY BLOCKS					
		CORE	PAGE 0	PAGE 1	PAGE 2	HSDB	DMA
Implementation		Mag. Core	Bipolar	Bipolar	Bipolar	Bipolar	?
Volatile		No	Yes	Yes	Yes	Yes	?
Number of Words		16,384	512	512	512	512	30,720
Bits Per Word		32	32	32	32	32	
First Octal Address		100000	000000	001000	002000	003000	004000
Last Octal Address		137777	000777	001777	002777	003777	077777
First Hex Address		8000	000	200	400	600	0800
Last Hex Address		BFFF	1FF	3FF	5FF	7FF	7FFF
Port Switch Control		Priority	Ext Fcn	Ext Fcn	Ext Fcn	Priority	Priority
BUS	Buffered I/O	RW	-	-	-	RW	RW
	AP Control Data	RW	-	-	-	RW	RW
	AP Control Instr	R	R	R	R	R	R
	Program Pager	R	W	W	W	R	R
	Sequential Control	RW	R	R	R	RW	RW

-	Bus cannot access memory.
R	Bus can only read from memory.
W	Bus can only write into memory.
RW	Bus can both read and write into memory.
?	Depends on customized use of DMA.

ASSOCIATIVE PROCESSOR CONTROL

GENERAL

The major function of the AP control is to control the STARAN S associative arrays. AP control fetches instructions from the AP control memory. A 16-bit program counter contains the address of the instruction, while a 32-bit instruction register contains the instruction itself. Some instructions perform array operations, while others perform AP control functions. Internal registers affect control operations and other elements of the hardware.

AP control consists of eight basic elements (see figure 1-3) as follows:

1) Instruction Register
2) Program Control
3) Bus Logic
4) Common Register
5) Field Pointers and Length Counters
6) Response Store Control
7) Array Control
8) Resolver

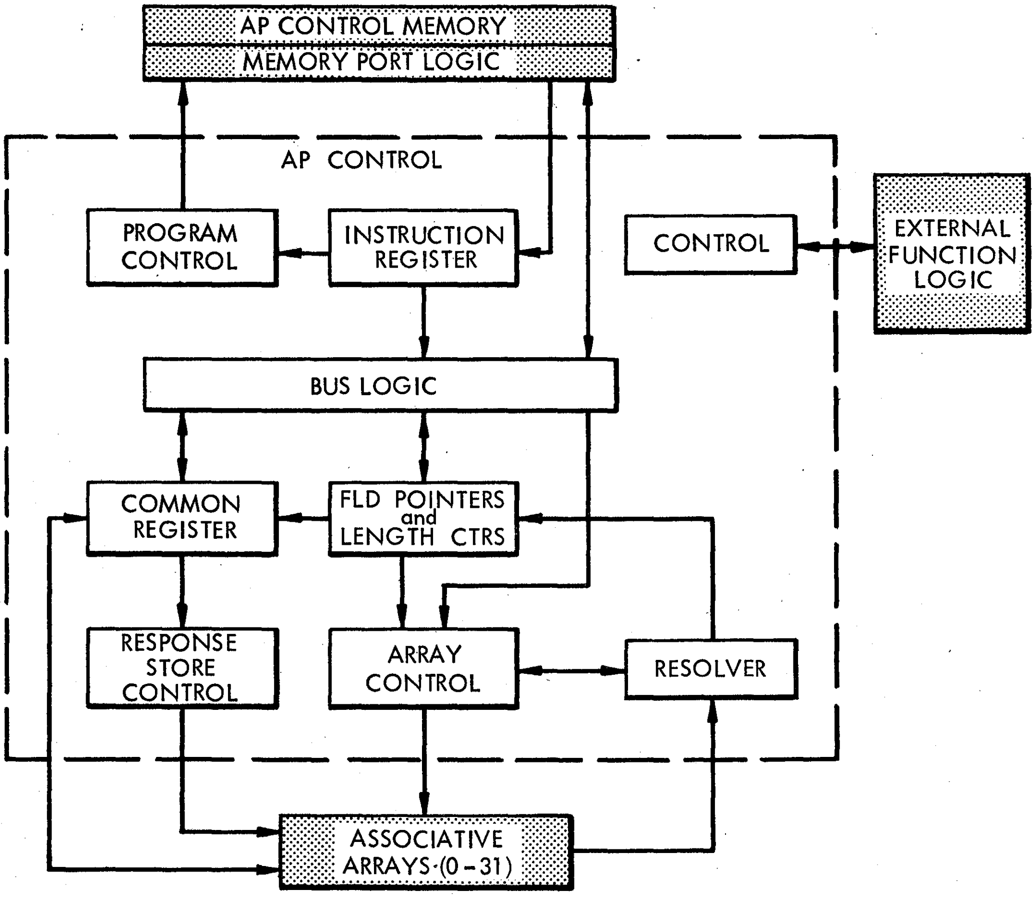


Figure 1-3 AP Control Block Diagram

INSTRUCTION REGISTER

The instruction register contains the instruction being executed. The instruction loaded into the instruction register is received from AP control memory via the instruction bus. Parity is checked at the instruction register. The instruction register contains 32 bits, which are numbered from 0 to 31 with bit 0 at the left. Portions of the instruction register are used as a direct source of data or addresses as a function of the instruction being executed.

PROGRAM CONTROL

The sequence in which instructions are obtained from AP control memory is controlled directly by the program control logic. The program control logic consists of the following: the program counter, the start loop marker, the end loop marker, the comparator, and the status register.

Program Counter

The program counter contains the address of the instruction being read from control memory. It is a 16-bit counter incremented by AP control. The program counter may be loaded from the bus logic; e.g., a branch instruction loads an address. The contents of the program counter form bits 0 through 15 of the program status word.

Start Loop Marker

The start loop marker is used to store the address of the first instruction immediately following a loop instruction. The start loop marker is a 16-bit register loaded directly from the program counter at the start of an instruction loop. It is loaded into the program counter when the last instruction of the loop has been executed and the loop is to be repeated.

End Loop Marker

The end loop marker is used to store the address of the last instruction of a loop. The end loop marker is a 16-bit register loaded from the rightmost 16 bits of the loop instruction.

Comparator

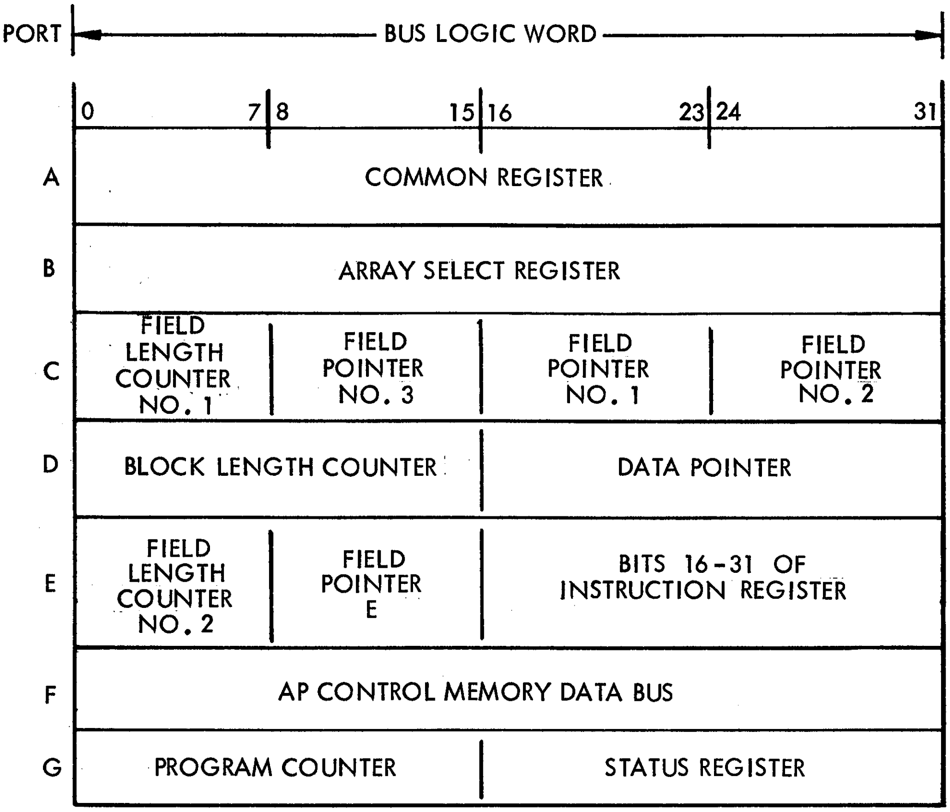
The comparator compares the address contained in the end loop marker with the address in the program counter. The comparator is a full 16-bit comparator, the output of which is transmitted to control as an indication that the end of the loop has been reached. The control then loads the start loop marker contents into the program counter if the loop is to be repeated.

Status Register

The status register contains the interrupt mask (IMASK) for the 15 AP control interrupts. All interrupts with numbers higher than the mask are accepted. The status register makes up bits 28 through 31 of the program status word.

BUS LOGIC

The bus logic provides a common data path for all pertinent registers of AP control and the data bus from control memory. The bus is 32 bits wide. Registers of less than 32 bits are grouped to form a 32-bit word. Details of registers connected to the bus and register grouping are shown in figure 1-4.



PORT	0–7	8–15	16–23	24–31
A	COMMON REGISTER			
B	ARRAY SELECT REGISTER			
C	FIELD LENGTH COUNTER NO. 1	FIELD POINTER NO. 3	FIELD POINTER NO. 1	FIELD POINTER NO. 2
D	BLOCK LENGTH COUNTER		DATA POINTER	
E	FIELD LENGTH COUNTER NO. 2	FIELD POINTER E	BITS 16-31 OF INSTRUCTION REGISTER	
F	AP CONTROL MEMORY DATA BUS			
G	PROGRAM COUNTER		STATUS REGISTER	

Figure 1-4. AP Control Bus Logic Port and Bit Allocations

Shift Logic

Data transmitted via the bus logic passes through the bus shift logic. The bus shift logic shifts the bus word left end around by either 0-, 8-, 16-, or 24-bit positions. The amount of shift is controlled by the instruction moving the data. Data received from AP control memory is checked for correct parity as it passes the bus shift logic. Data stored in the control memory has a parity bit generated by the bus shift logic.

Data Pointer Register

The data pointer register contains the control address for the data bus for block transfers. The data pointer is a 16-bit counter. The data pointer can be stepped with each transfer within a data block.

Block Length Counter

The block length counter controls the length of a data block transfer. The block length counter is a 16-bit decrementing counter.

COMMON REGISTER

The Common Register contains the argument for a search operation performed upon the associative arrays, the input data to be stored into an array, or the output data loaded from an array. Output data from an array is loaded into the common register through a mask generated by the mask generator. The Common Register contains 32 bits which are numbered from 0 to 31. Bit 0 is the left (most-significant) bit and bit 31 the right (least-significant) bit. The search argument or array input data is loaded via the bus logic. The array output data is loaded through a mask. The use of the mask allows formatting of an output word from noncontiguous data in an array.

COMMON REGISTER

Mask Generator

The mask generator generates a mask pattern to be used in loading array output data into the common register. The mask enables data to be loaded for a number of contiguous bits. The mask generator requires the bit addresses of the most- and least-significant bits to be loaded. All bits between and including these limits are loaded, while those outside these limits are unaltered.

FIELD POINTERS AND LENGTH COUNTERS

Field pointers generally contain bit slice or word addresses for indirect addressing mode in the associative array operations. The field length counters control the number of bits to be operated on in sequence. There are three field pointers and two field length counters. In addition, one register for temporary pointer storage is required for the multiply-and-divide associative instructions. A selector is used to route either field pointers 1, 2, or 3 or the address field of the instruction register to the array. These registers are 8 bits in length and their contents range from 0 to 255. When attempting to increment above 255, the register will return to zero; when attempting to decrement below zero, the register contents become 255. The field length counters can be decremented only.

Field Pointer 1

Field pointer 1 may contain an array address for an associative array operation or the address of a selected bit of the common register to be used for a search instruction. Field pointer 1 (like field pointers 2 and 3) is an 8-bit counter. As a result of resolve operation, field pointer 1 will be loaded with the number of the array module containing the first responder (i.e., first selected word whose Y flip-flop is set to one).

Field Pointer 2

Field pointer 2 may contain an array address for an associative array operation. As a result of a resolve operation, FP2 will be loaded with the word address of the first responder in the array specified in FP1.

Field Pointer 3

Field pointer 3 contains an array bit or word address.

Field Pointer E

Field pointer E is an 8-bit counter used for temporary storage of an array bit or word address. The register is used in the execution of the associative multiply-and-divide subroutines to hold the successive starting points of the partial products or intermediate dividends. It also may contain a shift constant for certain associative operations.

Field Length Counter 1

Field length counter 1 and field length counter 2 are 8-bit counters. The length counters can be decremented only. When the contents of a field counter become zero, a signal is sent to AP control for test purposes. This permits the program sequence to be altered by a branch if a field length

Field Length Counter 1 (cont)

counter becomes zero. Field length counter 1 is typically used to control the number of cycles made for a minor instruction loop, such as search or add fields.

Field Length Counter 2

Field length counter 2 may be used to control the cycles of a major instruction loop, suchas multiply fields.

RESPONSE STORE CONTROL

The response store control logic generates the control signals required by the associative arrays and buffers them to insure correct timing at the response store. The response store control consists of the control line conditioner and the control line buffer.

Control Line Conditioner

The control line conditioner generates the control lines required to manipulate the response store. Control line signals are generated as a function of the instruction register, a selected bit of the Common Register and the inclusive OR output from the resolver.

Control Line Buffer

The control line buffer controls the timing of the control lines transmitted to the associative arrays.

ARRAY CONTROL

Control lines to the associative arrays not generated by the response store control are generated by the array control. The array control logic selects which arrays are to be used and controls such things as bit/word mode, store mask, and shifting.

Array Select Register

The array select register establishes which array modules are to be enabled for an operation. The array select register is 32 bits wide. Each bit position controls one array. Bit 0 corresponds to array 0, and a 1 in a bit position enables the corresponding array. The array select register contents are also used by the resolver logic.

Array Address Select

The array address select logic selects either the array select register or field pointer 1 to generate the array enable signals. When field pointer 1 is selected, the five right-most bits of the pointer specify the one array to be enabled. This is done without modifying the contents of the array select register. Such operations as loading one item of data from an array or storing one item of data into an array enables only one of the associative arrays. When more than one array is involved in an operation, the array select register is used to select the arrays to participate.

ARRAY CONTROL

Array Mode

The array mode logic controls the addressing mode. Either a bit slice or a word slice in the arrays is selected for loading or storing.

Shift Control

The shift control logic generates the control signals required by the array to perform shifting and mirroring operations.

RESOLVER

The resolver logic finds the array address and word address of the first (most-significant) responder. A responder is a response store element in an enabled array whose Y flip-flop is set to one. Generally, responders indicate those words satisfying some search criteria. The array address is loaded into field pointer 1 and the word address is loaded into field pointer 2. This allows subsequent operations to affect only the first responder of a search.

GENERAL DESCRIPTION

Each STARAN array(see figure 1-5) includes a 256-word by 256-bit square array of solid-state multidimensional access (MDA) storage. Associated with each word of an array is a three flip-flop response store element (M,X, and Y registers). Also included in each array is a network to facilitate data manipulation and interword communication, a 256-bit wide PI/O port communicating with the custom I/O unit, and dual control ports to allow processing in some arrays concurrently with input/output in other arrays.

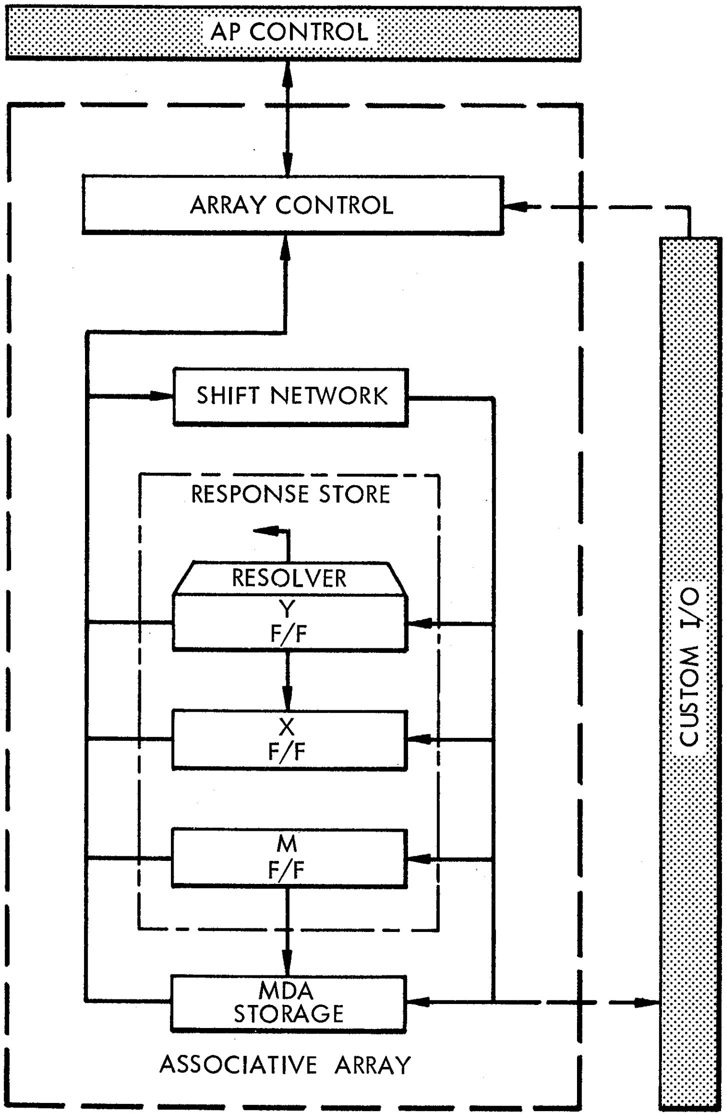


Figure 1-5. Associative Array Block Diagram

GENERAL DESCRIPTION (cont)

The associative array consists of two basic components, the array storage and the response store. As many as 32 associative arrays may be contained within one system. Arrays may be selected and operated in parallel or one at a time.

Each array contains 65,536 bits, organized as a square 256 words by 256 bits of solid-state storage. By use of a special organization within the array, access may be made in either the bit or word direction. An entire word of 256 bits or a bit slice, bit n of all 256 words, may be accessed.

The response store portion of the array consists of 256 response store elements. The M, X, and Y response store registers (256 bits each) may be used as temporary storage of data loaded from the array or may contain the data to be stored into the array. The X and Y registers may also logically combine array data simultaneously with the load response store operations. The M (MASK) register is used as a MASK to select which words of array memory participate in an array Store Operation.
The Y register is used as a responder by the resolver in a search type operation.

Associative array input and output may be either 32 bits (via the common register) or 256 bits (via the response store registers or Parallel I/O if the PI/O option is selected).

ADDRESSING

Addressing of and within the associative arrays is carried in hexadecimal notation. The four basic areas to be addressed are arrays, words, bit columns, and fields. (See figure 1-6)

Arrays

Addressing of arrays is accomplished by an address from 0, the first array, to 31, the last array. The number of arrays within a system, therefore, ranges from 1 to 32 as dictated by the requirement placed on the system. The internal organization and addressing of all arrays are identical.

Words

Addressing of a word within an array is accomplished by an address from 0, the first word, to 255, the last word. An array word consists of 256 bits, all of which may be accessed in parallel or divided into eight fields of 32 bits each and accessed via the Common Register.

Bit Columns

Addressing of a bit position within an array is accomplished by an address from 0, the most-significant (left), to 255, the least-significant (right) bit position. Bit (n) of all words is accessed by using address (n).

Fields

Addressing of a particular field of an array word is accomplished by an an address from 0, the most-significant (left) field, to 7, the least-significant (right) field. Each field contains 32 contiguous bits within the word being addressed. The most-significant field starts at the most-significant bit position.

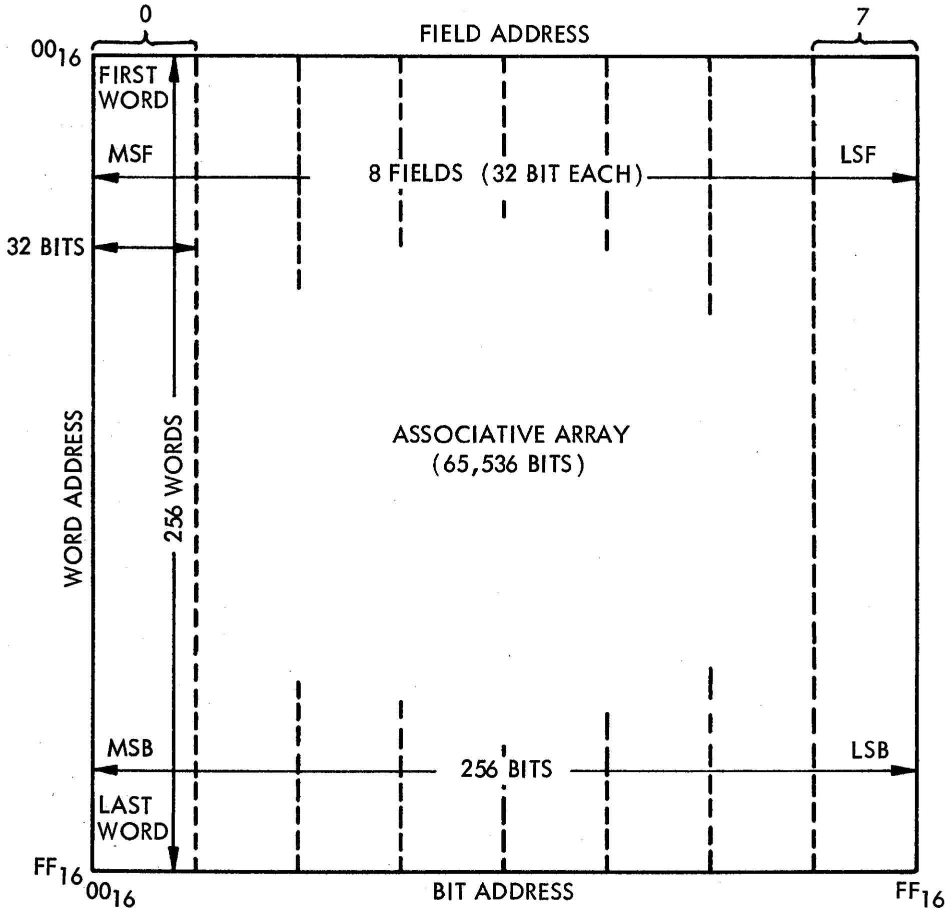


Figure 1-6. Associative Array Address Map

OPERATIONS

The operations performed within the associative array can be grouped into the following categories: load, store, logical, and resolve address.

Load

Data is loaded from the array storage in either the bit or word mode. Data loaded from the array storage can be sent to the 256-bit output bus or loaded into one of the response store registers. By specifying a field address (section), a field (section) of one word may be loaded from the array storage for output over the 32-bit output bus to the common register. Also, one response store register may be loaded with the contents of another response store register. All loads from the array storage are nondestructive. Logic may be performed between the X and Y registers and the input simultaneously with the load.

OPERATIONS

Masked Store

Data is stored into the array memoryin either the bit or word mode. The data, regardless of source, may be stored through a mask contained in the M response store register. Data to be stored may come from the response store registers, the 32-bit input bus, or the 256-bit input bus (parallel I/O).

Logical

To perform operations such as exact-match search and add fields, certain logical operations must be performed on the data in the response store registers. Data loaded from the array memory may be logically combined with the current contents of the response store to accomplish these operations. Data may also be transferred among the response store registers with logic functions applied to the X and/or Y registers.

Resolve

The array address and bit or word address of the first Y response store element set is continuously resolved. This address and the inclusive OR of all Y response store elements are made available to control. This address (which is stored in FP1 and FP2) may be used in succeeding operations.

PROGRAM PAGER

GENERAL

The function of the program pager is to transfer program segments from the bulk core memory to the page memories.

Under normal programming practice, the pager is activated by AP control when a new program segment is to be transferred to a page memory. The program pager transfers the segment one word at a time at a rate dictated by the source memory, while AP control executes instructions from previously loaded segments. When the pager completes the transfer, it restores the page memory port switch to the AP instruction bus and halts.

The program pager contains three registers. (See figure 1-7.) The GET address register contains a 16-bit AP control memory address. If the pager is in the midst of moving data, the GET address points to the memory location containing the next source word to be moved. At other times the GET address acts like a program counter, pointing to the location of the next page memory instruction to be executed.

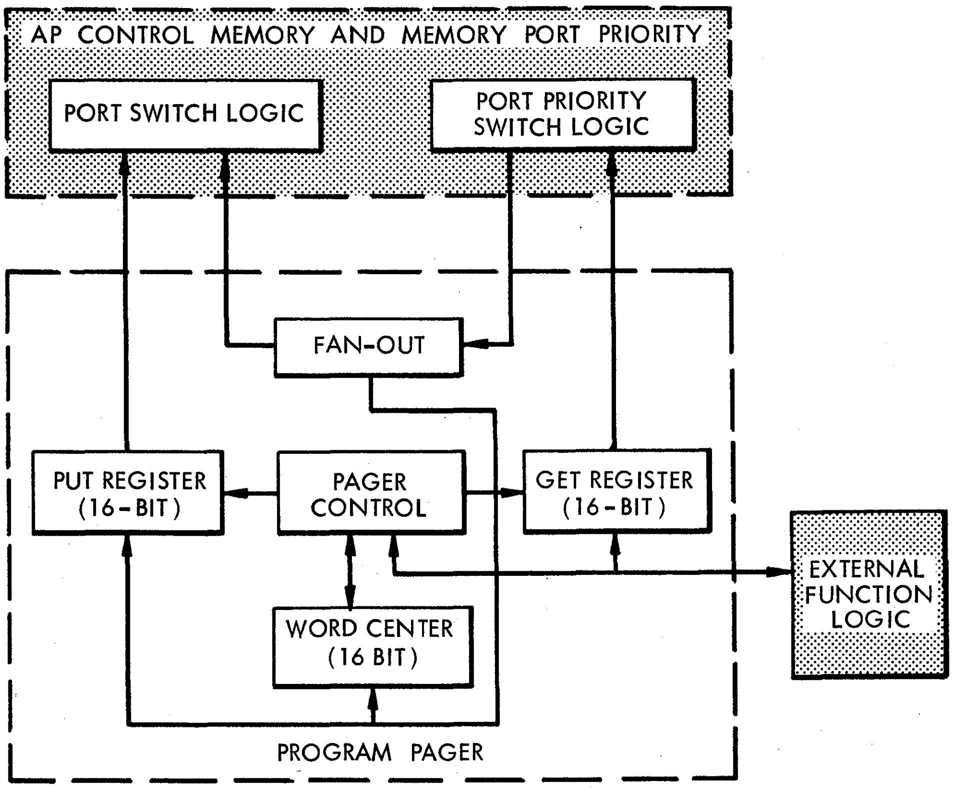


Figure 1-7. Program Pager Block Diagram

GENERAL (cont)

The PUT address register holds a 16-bit AP control address. It points to the memory location into which the next destination word is to be put during a move-data operation.

The 14-bit count register holds the number of words still to be transferred during a transfer operation.

OPERATION

Pager transfer speed is governed by the cycle time of the source memory, which can be slower than AP control execution speed. To keep STARAN from being "pager bound," AP programs should be segmented carefully. An ideal program segment will contain enough long instructions, subroutine calls and loops, etc., so that before AP control leaves the segment, the pager has had enough time to get the next segment loaded in a page memory. If all segments are ideal, AP control will never wait for the pager.

Each page memory has a port switch to prevent AP control from jumping into a segment before the pager has loaded it. A premature attempt to jump into a page memory that is still being loaded by the pager will be delayed until the pager has switched the port switch to the instruction bus.

Pager operation is initiated by an external function code that loads the GET address register. With one external function code the current pager operation can be stopped in midstream and a new operation started. For instance, suppose from program segment 1, AP control either jumps to program segment 2 or to program segment 3, depending on some condition. Most of the time it jumps to segment 2. In this case, AP control can initiate the loading of segment 2 as it begins executing segment 1, so that little or no time is lost in waiting for the pager. In the rarer case, when AP control jumps to segment 3, it can stop the loading of segment 2 in midstream, if necessary, and start the loading of segment 3.

When the program pager is running (busy), it fetches pager commands and source data from AP control memory. Its commands are sufficiently general to permit flexible pager programming.

EXTERNAL FUNCTION LOGIC

GENERAL

Numerous hardware functions are under the control of external function (EXF) logic. These include page memory port switches, AP and sequential control interrupts, AP control and program pager activity control, and resets and clears. Control and status sensing of these functions are accomplished by issuing 19-bit external function commands to EXF logic and receiving one-bit sense signals in return. Three elements of the main frame can issue EXF commands: AP control, program pager, and sequential control. (See figure 1-8.)

EXF logic is expandable to allow receipt of EXF commands from the custom I/O unit and control of other hardware functions in the custom I/O unit. A resolver in EXF logic allows only one EXF command to be treated at a time.

The resolver in EXF logic resolves conflicts among the four elements issuing function codes. One function code at a time is accepted by EXF

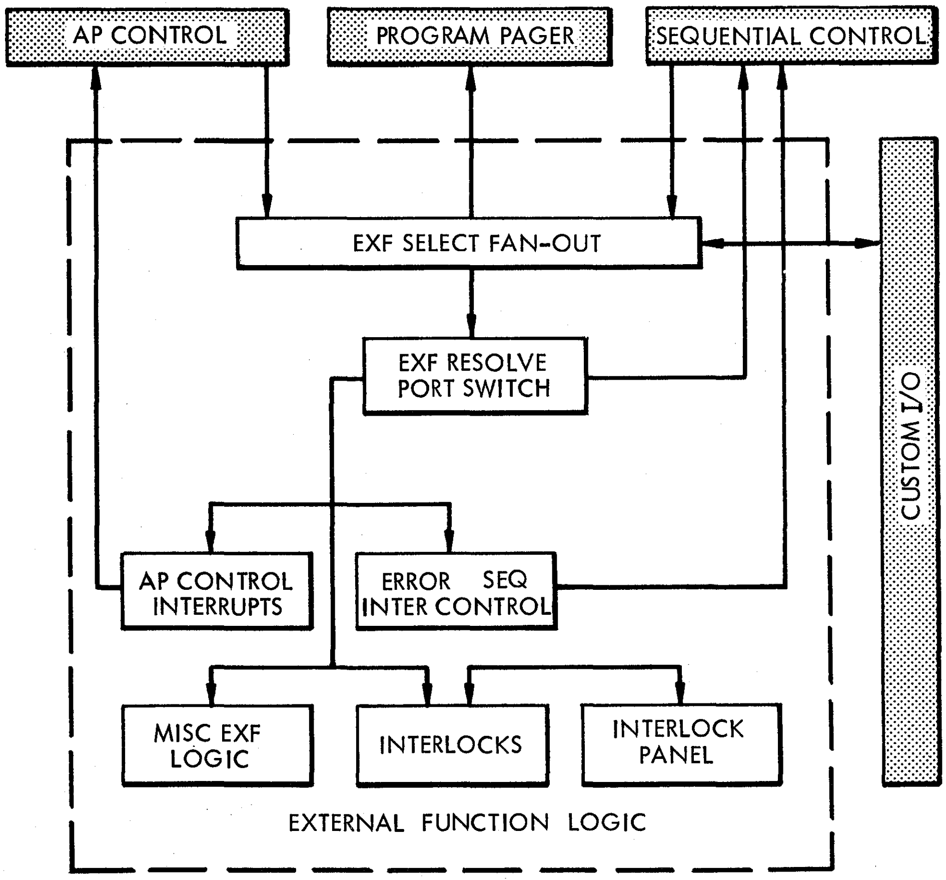


Figure 1-8. External Function Logic Block Diagram

GENERAL (cont)

logic. The interrogation and/or control called for is performed and then another function code is accepted if one is present. A function code can interrogate and control an element in one operation without interference from another function code.

The classes of function codes are as follows: page memory port switches, interlocks, program pager, error control, AP control interrupts, sequential control interrupt, miscellaneous, and spare. These classes are described in the following sections.

PAGE MEMORY PORT SWITCHES

Each page memory has a port switch that connects it to one of three buses. The port switch is controlled by external function codes. At any given time, a page memory is connected to 1) the instruction bus, which allows AP control to read instructions from the page; 2) the pager bus, which allows the program pager to load the page; or 3) the sequential control bus, which allows sequential control to read items from the page. If one of these buses should try to access a word in the page memory while the port switch is set to another bus, a hangup results. Hangups, which are detected by error detectors, cause interrupts to sequential control.

This class of external function codes allows interrogation and control of the port switches. The current state of the selected port switch is interrogated, and depending on whether it is on the sequential bus, the instruction bus, or the pager bus, one of the sense bits of the function code is returned. The current status switch is also used to select a new switch setting of the function code.

INTERLOCKS

The EXF logic contains 64 stored bits called interlocks. These bits have no predetermined meaning; software can assign a function to the interlocks and use them for various purposes. They are useful for passing signals among STARAN S elements, such as what programs should be executed, etc. Function codes allow the current state of an interlock to be sensed and a new state to be entered in one operation.

Sixteen interlocks (hexadecimal address 00 through 0F) can be controlled and sensed manually by panel switches and lights on the interlock panel to facilitate communication with an operator. The other 48 interlocks (hexadecimal addresses 10 through 3F) can only be sensed and controlled by function codes.

Interlocks are volatile, and so their states are lost whenever power is lost.

PROGRAM PAGER

Certain external function codes allow program pager operation to be sensed, initiated, halted, and modified.

The program pager has two states: Off, and Busy. A function code allows the state to be sensed and changed. If the current pager state is off, the function code can either leave it in the off condition or turn it on, in which case (if it is operative) the pager will become busy.

If the current pager state is busy, the function code can either turn the pager off or leave it in the busy condition.

ERROR CONTROL

Error detectors are included in various elements of STARAN S to sense hardware faults and program errors. Each error detector sets an error indicator when an error is detected. Each error indicator is given a number by which it may be sensed, set, and/or reset by function codes. If any error indicator is set, an interrupt to sequential control is generated. Also, certain error indicators will make the program pager inoperative and others will make AP control inactive. Since an error indicator can be set by a function code, errors can be simulated in order to check out error handling routines, etc.

Error indicators in the standard STARAN S configuration are shown in table 1-2.

Table 1-2. External Function Logic Error Indicators

Number (Octal)	Error Description	Action When Set*
00	Illegal Instruction in AP Control	AP Control Inactive
01	AP Control Hung Up in a Loop	AP Control Inactive
02	Buffered I/O Bus Hung Up	- - - - - - - - - - -
03	AP Control Data Bus Hung Up	AP Control Inactive
04	AP Control Instruction Bus Hung Up	AP Control Inactive
05	Pager Get Bus Hung Up	Pager Inoperative
06	Pager Put Bus Hung Up	Pager Inoperative
10	Parity Error on AP Control Data Bus	AP Inactive
11	Parity Error on AP Control Instruction Bus	AP Inactive
13	Parity Error on Sequential Control Bus	- - - - - - - - - - -

* Every error also generates an interrupt to sequential control. Other errors may be added in the custom I/O cabinet.

ASSOCIATIVE PROCESSOR CONTROL INTERRUPTS

The AP control interrupt is a class of function codes used to sense, set, and reset the state of 15 programmable interrupts to AP control. AP control interrupts are given hex addresses 01 (lowest priority) to 0F (highest priority).

Bits 28 through 31 of the program status word (PSW) in AP control contain an interrupt mask. AP control accepts interrupt n if the following conditions are satisfied: 1) AP control is active and at an interruptable point; 2) the interrupt mask is less than n; 3) no interrupt of higher priority is set; and 4) interrupt n is set. When AP control accepts interrupt n, it fetches the next instruction from hex address 8000 + n (without disturbing the contents of its program counter). Normally, this instruction is a swap PSW instruction, which saves the old PSW and loads in a new PSW, causing AP control to enter an interrupt routine. The interrupt mask of the new PSW should be n or more to prevent AP control from accepting interrupt n again until the interrupt routine is complete.

SEQUENTIAL CONTROL INTERRUPTS

The sequential control interrupt class of function codes can sense, set, and reset the state of eight programmable interrupts to sequential control. Table 1-3 shows the vector addresses of the sequential control interrupts, together with the priority levels.

Table 1-3. Sequential Control Interrupt

Vector Address (Octal)	Priority Level	Relative Priority
334	7	High
330	7	↓
324	6	
320	6	
314	5	
310	5	
304	4	
300	4	Low

When the processor priority is set at n, all requests for interrupts at level n and below are ignored.

Bits 19 through 27 of the instruction select one of the sequential control interrupts (see above vector address assignments). The current state of the selected interrupt is sensed and a sense bit is returned. Also, a new state may be assigned to the interrupt.

MISCELLANEOUS EXTERNAL FUNCTION LOGIC

General

This class of function codes senses and controls the status of certain STARAN S elements. These codes are discussed in the following subsections.

AP Control Activity

An external function code senses and controls the AP control activity. AP control has two states: inactive and active. In the active state it fetches instructions from AP control memory and exercises the associative arrays. In the inactive state it does not fetch any instructions.

When switched from the inactive state to the active state, AP control fetches its first instruction from hexadecimal address 8000 without disturbing the program counter. It could be a No-op, which would cause AP control to continue with its previous program, or it could be a swap program status word instruction, which would cause the old status of AP control to be saved and a new program entered.

AP Control Loop Indicator

When AP control executes a loop-type instruction, a loop indicator is set in the loop indicator function code and remains set until all repetitions of the loop are completed. The indicator is neither disturbed by changes in activity nor by interrupts. Execution of a loop instruction when the loop indicator is still set from a previous loop is illegal. Function codes allow the loop indicator to be sensed and/or reset. Resetting of a set loop indicator causes AP control to forget the loop instruction that set it, even if all repetitions of the loop were not completed.

Resets and Clears

Other external function codes are reserved for selective resetting and clearing of various STARAN S registers and status flip-flops. They are used for clearing any hangup conditions that may arise, and also by the power-on program module (in sequential control) to cause a master clear in STARAN S when power is turned on.

SEQUENTIAL CONTROL

GENERAL

The sequential control device used in STARAN S consists of a sequential processor (SP) with 8K of memory, interface logic to connect the SP to other STARAN S elements, and peripheral units, which include a keyboard/printer and a perforated tape reader/punch unit. As described earlier, the sequential control provides:

1) A means to initially load the AP memory
2) A communication link between the operator and STARAN S for on-line control and monitoring
3) Off-line capabilities for assembling and debugging STARAN S programs
4) Control for STARAN S maintenance and diagnostic program routines
5) Housekeeping capabilities

SEQUENTIAL PROCESSOR DESCRIPTION

The sequential processor is a 16-bit, general-purpose, parallel-logic minicomputer using two's complement arithmetic. The 8,192 16-bit words (16,384 8-bit bytes of memory) have octal addresses 000000 through 037777. All communication between system components is performed on a single high-speed bus. There are eight general-purpose registers, which can be used as accumulators, index registers, or address pointers, and a multilevel automatic priority interrupt system.

The sequential processor features include:

1) Single and double operand addressing
2) 16-bit word and 8-bit byte addressing
3) Simplified list and stack processing through auto-address stepping (auto-incrementing and auto-decrementing)
4) Eight programmable general-purpose registers
5) Data manipulation directly within external device registers
6) Addressing of device registers using normal memory reference instructions
7) Asynchronous operation of SP memory, processor, and I/O devices
8) Hardware interrupt priority structure for devices peripheral to the SP
9) Automatic interrupt identification without device polling
10) Direct addressing of the SP 8K words or 16K bytes

A single common path connects the SP memory and all peripherals. Addresses, data, and control information are sent along the 56 lines of

SEQUENTIAL PROCESSOR DESCRIPTION (cont)

the bus. All instructions that can be applied to data in the SP core memory can be applied as well to data in peripheral device registers.

The common path lines are bidirectional. A peripheral device register can be either read or set by the SP or other peripheral devices; thus the same register can be used for both input and output functions.

Communication between two devices on the common path is in the form of a master-slave relationship. A controlling device(termed the bus master) controls the bus when communicating with another device on the bus (termed the slave).

Common path communication is interlocked so that for each control signal issued by the master device, there must be a response from the slave device in order to complete the transfer. The maximum transfer rate on the path is one 16-bit word every 750 nanoseconds or 1.3 million 16-bit words per second.

SEQUENTIAL CONTROL INTERFACE

General

Communication between sequential control and other STARAN S elements is accomplished using certain device-register addresses of the common path and certain interrupt-vector addresses. Four forms of communication are described below.

1) Direct Access To AP Control Memory: Words in the AP control memory are given sequential control bus addresses to facilitate transfer of data and instructions between AP control and sequential control.

2) Register Readout: Certain registers in STARAN S can be read by sequential control. This facilitates program debugging and hardware maintenance and test.

3) External Functions: External function codes can be transmitted to the external function logic and sense bits received. This state allows sequential control to activate and deactivate AP control, issue interrupts, and perform many housekeeping tasks.

4) Interrupt Acceptance: Other elements of STARAN S can issue interrupts to sequential control by issuing certain function codes to the external function logic. Also, when errors such as parity errors are detected, a sequential control interrupt is issued. This function allows real-time control of the resources in sequential control.

Detailed descriptions of these forms of communication are given in the next four subsections.

Direct Access To AP Control Memory

Sequential control can read and write AP control memory data. For this purpose AP control memory is considered to be divided into 12 groups. Each group contains 4096 32-bit words (16,384 bytes). Group 0 contains AP octal addresses 000000 through 007777. Group 1 contains addresses 010000 through 017777, etc. Group 11 contains addresses 130000 through 137777.

To access any of the 16,384 bytes of a group, a sequential control program should first store the group number in a special 4-bit register called the Group Register (GRP), whose sequential control octal address is 164064. This register can be read, written, and incremented by sequential control.

The 16,384 bytes of the selected group can be accessed using sequential control addresses 040000 through 077777. Addresses 040000 through 040003 access parts of the first AP control memory word in the group, addresses 040004 through 040007 access parts of the second memory word, and so on. Addresses 077774 through 077777 access parts of the last word in the group.

Addressing of bytes within an AP control memory word takes place from right to left; e.g., address 040000 accesses the rightmost byte and address 040003 accesses the leftmost byte of the first memory word in the group.

Addressing of 16-bit halves of AP control memory words also takes place from right to left, using even addresses; e.g., address 040000 accesses the right half and address 040002 accesses the left half of the first memory word in the group.

Words in the page 0, page 1, and page 2 memories can only be read by sequential control. Other AP control memory words can be both read and written by sequential control.

Register Readout

To assist personnel during program debugging, hardware maintenance, or test operations, certain STARAN S registers can be read by sequential control by reading certain bus addresses. The register sequential control addresses are shown in table 1-4. The octal addresses for these registers are in the range of 164000 to 167777. Load operations into these addresses are ignored. AP control should be inactive when reading AP registers or meaningless data will be read. Similarly, the pager should be in the off state when reading the pager registers.

External Functions

The external function (EXF) logic is used to control and sense various elements of STARAN S. Elements of STARAN S issue 19-bit external function codes to the EXF logic and receive 1-bit sense returns. Sequential control issues external functions using two device addresses, 164030 and 164032, that are

External Functions (cont)

given the mnemonics EFS and EFB respectively. To issue an external function, sequential control should put bits 13 through 15 of the external function into bits 2 through 0 respectively, of EFS, and then put bits 16 through 31 of the external function into bits 15 through 0 respectively, of EFB. These bits cannot be read by the bus. Loading of EFS or EFB clears bits 15 and 7 of EFS to zeros. When the external function is treated by the EXF logic, one of these bits is set to one. Bit 7 is set to one if the sense bit returned is zero. Bit 15 is set to one if the sense bit returned is one. Bits 7 and 15 of EFS cannot be set by the bus.

Table 1-4. Sequential Control Readout Registers

Octal Address	Register Abbreviation	Register	Length In Bits
164000	DAL	Seq. Bus Data Reg. (Low Order)	16
164002	DAH	Seq. Bus Data Reg. (High Order)	16
164004	PUT	Pager Put Address	16
164006	GET	Pager Get Address	16
164010	IRL	AP Instruction Reg. (Low Order)	16
164012	IRH	AP Instruction Reg. (High Order)	16
164014	ELM	End Loop Marker	16
164020	CL	Common Register (Low Order)	16
164022	CH	Common Register (High Order)	16
164026	CNT	Pager Word Count	16
164036	SLM	Start Loop Marker	16
164060	ASL	Array Select Register (Low Order)	16
164062	ASH	Array Select Register (High Order)	16
164064	GRP	Group Register	4
164120	DP	Data Pointer Register	16
164122	BL	Block Length Counter	16
164160	IMASK	Interrupt Mask	4
164162	PC	Program Counter	16
164222	FPE	Field Pointer E	8
164223	FL2	Field Length Counter No. 2	8
164360	FP2	Field Pointer No. 2	8
164361	FP1	Field Pointer No. 1	8
164362	FP3	Field Pointer No. 3	8
164363	FL1	Field Length Counter No. 1	8

Interrupt Acceptance

Sequential control can accept interrupts from other STARAN S elements. The interrupts arise from error detection, the panel-interrupt button, or external functions. Eight different interrupt vector addresses are provided. Table 1-3 shows the vector addresses and the priority levels.

Interrupt Acceptance (cont)

Bits 7, 6, and 5 in the processor status register (processor priority) of sequential control determine which interrupts are acceptable. When the processor priority is set at n, all requests for interrupts at priority level n and below are ignored.

Some sequential control peripherals may also generate interrupts to sequential control. Such interrupts do not use this interface.

Peripherals

STARAN S sequential processor has as standard peripherals a keyboard/ printer and a tape reader/punch.

The perforated tape reader and punch units handle 8-level tape with a reader speed of 300 frames per second and a punch speed of 50 characters per second. The keyboard/printer and tape reader/punch peripherals interface directly with sequential control. Communication between the remainder of the STARAN S and the peripherals is then handled through sequential control. The keyboard/printer provides a communication link between STARAN S and an operator. During operation STARAN S fault and error-alarm messages are output on the printer as detected by the on-line diagnostic program module in sequential control. The keyboard provides a means for the operator to input command statements to change operating modes of the STARAN S.

Optional Peripherals

Additional sequential control peripherals available are: card reader, line printer, disk, magnetic tapes, and communication to remote devices.

INPUT/ OUTPUT VARIATIONS

STARAN S provides many input/output options which can be customized to fit the requirements of a particular installation. A detailed discussion is provided in Chapter III.

Optional STARAN S Peripherals

The standard buffered I/O channal or the high bandwidth parallel I/O option can provide access to conventional disks and magnetic tape units or to a customized multihead disk for processing large data bases.

CHAPTER 2

STARAN S INSTRUCTION REPERTOIRE

GENERAL

PROGRAM SEQUENCE

The AP control fetches instructions from AP control memory sequentially unless a branch, loop, load program status word (PSW) or swap PSW is encountered. The sequence may also be altered by an interrupt request.

PROGRAM COUNTER

Normally, the program counter contains the address of the current instruction. After an instruction is executed the program counter is incremented to the next sequential instruction. However, when AP control moves from the inactive to the active state the next instruction will be fetched from a dedicated location (8000_{16}) in AP control memory without disturbing the program counter. Also,when an interrupt is accepted, AP control will fetch an instruction from dedicated locations (8001_{16} to $800F_{16}$) without modifying the program counter. This allows the program counter state to be preserved so that an interrupted program can be continued after an interrupt is satisfied.

INSTRUCTION LENGTH

STARAN S instructions are 32 bits long. The bits of an instruction are numbered from 0 to 31 with bit 0 representing the most-significant bit (left) and bit 31 the least-significant bit (right).

INSTRUCTION TYPES

STARAN S instructions are divided into 2 major types: Instructions for Associative Processor (AP) Control (described in Section I) and instructions for the Program Pager (described in Section II). Either of these elements may issue 19-bit External Function Codes (EXF) which are described in Section III.

GENERAL

Associative Processor (AP) control instructions are divided into four classes: register instructions, branch/loop instructions, associative array instructions, and external function instructions.

REGISTER INSTRUCTIONS

Register instructions allow AP control registers to be loaded with an immediate value in the instruction format or with the contents of an AP control memory address. Also, the AP control registers may be stored into AP control memory or into another AP control register.

REGISTER LOAD OPERATIONS

Three Load operations are possible:

1) Load low half will load right, or least-significant half of the register group (most-significant bit is 0; least-significant bit is 31).

2) Load high half will load the left, or most-significant half of the register group.

3) Load high and low half will load the entire register group specified.

AP CONTROL REGISTER GROUPS AND BUS POSITIONS

GROUP	MAP OF REGISTERS ON 32-BIT BUS			
	0 7	8 15	16 23	24 31
000	CH		CL	
001	ASH		ASL	
010	BL		DP	
011	PC			28—31 IMASK
100	FL2(2)	FPE(2)		
101	FL1			FP2
110		FP3	FP1	
111	FL1	FP3	FP1	FP2

(2) FL2 and FPE are reversed with respect to load operations; therefore, a load high (left) half will load FPE and a load low (right) half will load FL2.

INCREMENT MASK

LEFT SHIFT

The source data is shifted left-end-around, 0, 1, 2, or 3 bytes as specified in the shift field. The direction of the shift is from bit 31 toward bit 0.

EFFECTIVE ADDRESS FORMATION

Register operations that reference AP control memory for loading from memory or storing to memory allow the effective address to be a function of:

1) The address field of the instruction
2) The Data Pointer (DP), or
   Branch and Link Registers

The Tag Field of the instruction determines the usage of the Data Pointer Register and Branch and Link Registers. Decrementing and incrementing the Data Pointer Registers and decrementing the Block Length Counter are also permitted as specified in the Tag Field. This permits stepping through a block of data in AP control memory.

ASSOCIATIVE PROCESSOR CONTROL REGISTER ABBREVIATIONS

Following is a list of the AP control registers with their associated lengths and abbreviations:

	Abbr	Register		No. of Bits	Bus Position
Group 0	C	Common Register		32	0-31
	CH	Common Register	high order	16	0-15
	CL	Common Register	low order	16	16-31
	C0	Common Register	Byte 0	8	0-7
	C1	Common Register	Byte 1	8	8-15
	C2	Common Register	Byte 2	8	16-23
	C3	Common Register	Byte 3	8	24-31
Group 1	AS	Array Select Register		32	0-31
	ASH	Array Select Register	high order	16	0-15
	ASL	Array Select Register	low order	16	16-31
	AS0	Array Select Register	Byte 0	8	0-7
	AS1	Array Select Register	Byte 1	8	8-15
	AS2	Array Select Register	Byte 2	8	16-23
	AS3	Array Select Register	Byte 3	8	24-31
Group 2	BL	Block Length Counter		16	0-15
	BL0	Block Length Counter	Byte 0	8	0-7
	BL1	Block Length Counter	Byte 1	8	8-15
	DP	Data Pointer Register		16	16-31
	DP0	Data Pointer Register	Byte 0	8	16-23
	DP1	Data Pointer Register	Byte 1	8	24-31
Group 7	FL1	Field Length Counter 1		8	0-7
Group 4	FL2(1)	Field Length Counter 2		8	0-7
Group 7	FP1	Field Pointer 1		8	16-23
	FP2	Field Pointer 2		8	24-31
	FP3	Field Pointer 3		8	8-15
Group 4	FPE(1)	Field Pointer E		8	8-15
Group 3	PSW	Program Status Word		32	0-31
	PC	Program Counter		16	0-15
	IMASK	Interrupt Mask		4	28-31
	R0	Branch and Link Register 0		32(600)(2)	0-31
	R1	Branch and Link Register 1		32(601)	0-31
	R2	Branch and Link Register 2		32(602)	0-31
	R3	Branch and Link Register 3		32(603)	0-31
	R4	Branch and Link Register 4		32(604)	0-31
	R5	Branch and Link Register 5		32(605)	0-31
	R6	Branch and Link Register 6		32(606)	0-31
	R7	Branch and Link Register 7		32(607)	0-31

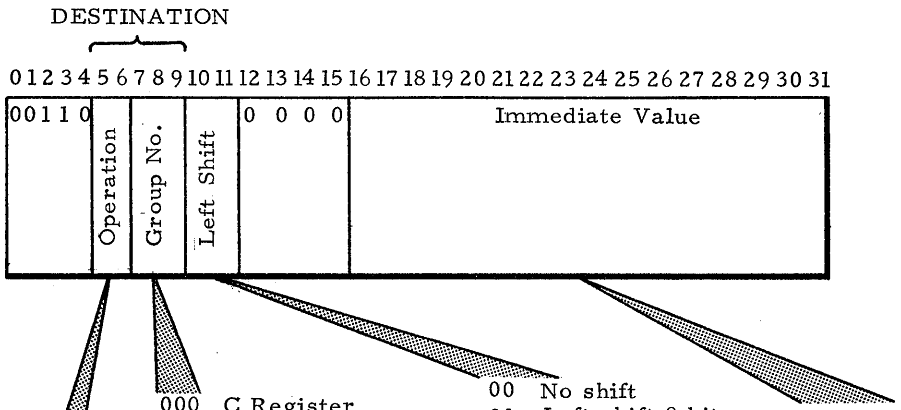
(1) FL2 and FPE are reversed with respect to load operations; therefore a load high (left)half will load FPE and a load low (right) half will load FL2.

(2) The branch and link registers occupy a dedicated area in the high-speed data buffer; the number in parentheses is the register hex address.

LOAD IMMEDIATE

The Load Immediate instruction loads either 4, 8, or 16 bits of the value field (bits 16 to 31 of the instruction) into a register, a pair of registers, or part of a register. This permits initialization of registers without a data fetch. The register loaded is selected by the destination field. The 16-bit immediate value from the instruction format is extended to a 32-bit quantity by appending 16 zeroes. The 32-bit quantity is put on the bus and then shifted left by 0, 8, 16 or 24 places end-around according to the left shift field. Parts of the shifted quantity are then loaded into the group or part of a group according to the destination field.

Instruction Format



000 C Register
001 AS Register
010 BL and DP
011 PC and IMASK
100 FL2 and FPE(2)
101 FP2 and FL1
110 FP3 and FP1
111 FL1, FP3, FP1, FP2

00 No shift
01 Left shift 8 bits
10 Left shift 16 bits
11 Left shift 24 bits

These 16 bits enabled to bus bits 16-31

01 Load low half of group
10 Load high half of group
11 Load high and low half of group

(2) FL2 and FPE are reversed with respect to the load operation; therefore, a load high half will load FPE, and a load low half will load FL2.

Sample Coding

- Load Low Half of Register Group

Instruction Hex Code (Bits 0-31)	REGISTERS BELOW ARE CLEARED		IMMEDIATE VALUE	
			Bits 16-23	Bits 24-31
3200nnnn(2)	▨	▨	CL	
3210nnnn	C3	▨	▨	C2
3220nnnn	CL		▨	▨
3230nnnn	▨	CL		▨
3240nnnn	▨	▨	ASL	
3250nnnn	AS3	▨	▨	AS2
3260nnnn	ASL		▨	▨
3270nnnn	▨	ASL		▨
3280nnnn	▨	▨	DP	
3290nnnn	DP1	▨	▨	DP0
32A0nnnn	DP		▨	▨
32B0nnnn	▨	DP		▨
32C0nnnn	▨	▨	▨	IMASK
32D0nnnn	IMASK	▨	▨	▨
32E0nnnn	▨	IMASK	▨	▨
32F0nnnn	▨	▨	IMASK	▨
3300nnnn	FL2	▨	▨	▨
3310nnnn	▨	FL2	▨	▨
3320nnnn	▨	▨	FL2	▨
3330nnnn	▨	▨	▨	FL2
3340nnnn	▨	▨	▨	FP2
3350nnnn	FP2	▨	▨	▨
3360nnnn	▨	FP2	▨	▨
3370nnnn	▨	▨	FP2	▨
3380nnnn	▨	▨	FP1	▨
3390nnnn	▨	▨	▨	FP1
33A0nnnn	FP1	▨	▨	▨
33B0nnnn	▨	FP1	▨	▨
33C0nnnn	▨	▨	FP1	FP2
33D0nnnn	FP2	▨	▨	FP1
33E0nnnn	FP1	FP2	▨	▨
33F0nnnn	▨	FP1	FP2	▨

REGISTER LOADED IS INDICATED IN BLOCKS

(2) nnnn = effective address

LOAD
IMMEDIATE

Sample
Coding
(cont)

- Load High Half of Register Group
- Load High and Low Half of Register Group

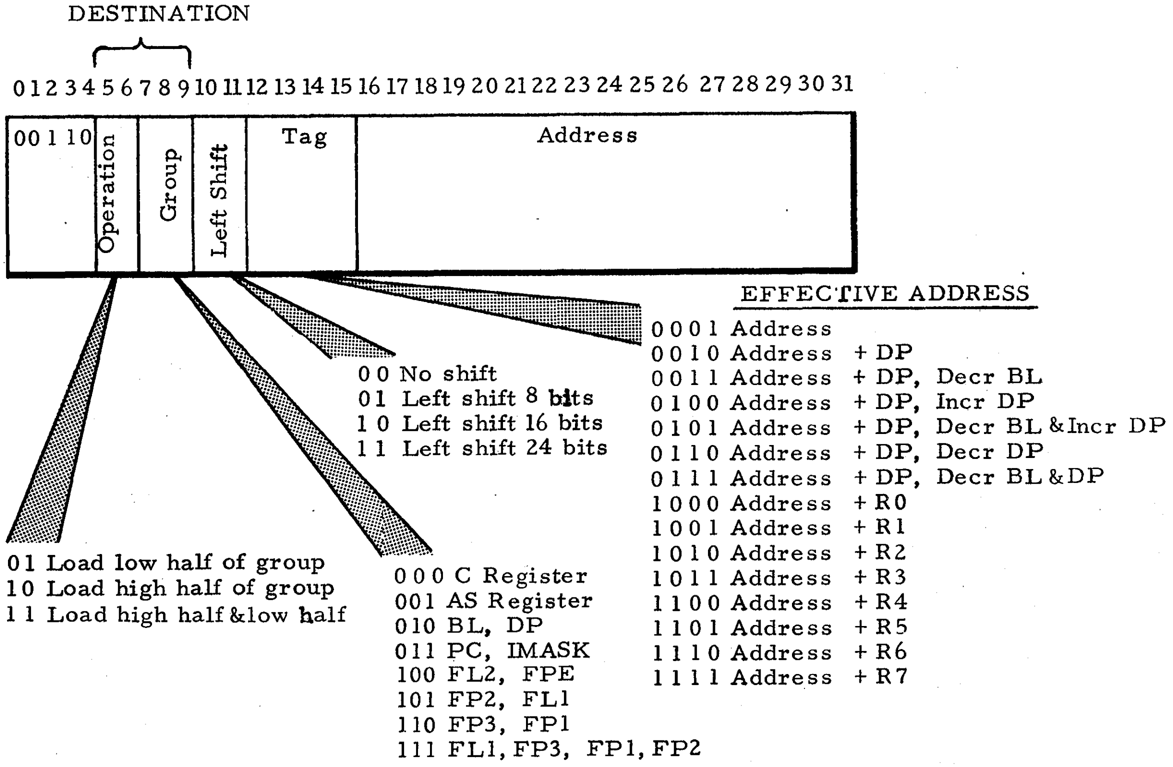
Instruction Hex Code (Bits 0-31)	REGISTERS BELOW		IMMEDIATE VALUE	
	ARE CLEARED		Bits 16-23	Bits 24-31
3400nnnn	CH			
3410nnnn		CH		
3420nnnn			CH	
3430nnnn	C1			C0
3440nnnn	ASH			
3450nnnn		ASH		
3460nnnn			ASH	
3470nnnn	AS1			AS0
3480nnnn	BL			
3490nnnn		BL		
34A0nnnn			BL	
34B0nnnn	BL1			BL0
34C0nnnn	PC			
34D0nnnn		PC		
34E0nnnn			PC	
34F0nnnn	PC1			PC0
3500nnnn		FPE		
3510nnnn			FPE	
3520nnnn				FPE
3530nnnn	FPE			
3540nnnn	FL1			
3550nnnn		FL1		
3560nnnn			FL1	
3570nnnn				FL1
3580nnnn		FP3		
3590nnnn			FP3	
35A0nnnn				FP3
35B0nnnn	FP3			
35C0nnnn	FL1	FP3		
35D0nnnn		FL1	FP3	
35E0nnnn			FL1	FP3
35F0nnnn	FP3			FL1
3600nnnn	C			
3610nnnn	C3	CH		C2
3620nnnn	CL		CH	
3630nnnn	C1	CL		C0
3640nnnn	AS			
3650nnnn	AS3	ASH		AS2
3660nnnn	ASL		ASH	
3670nnnn	AS1	ASL		AS0
3680nnnn	BL		DP	
3690nnnn	DP1	BL		DP0
36A0nnnn	DP		BL	
36B0nnnn	BL1	DP		BL0
36C0nnnn	PC			IMASK
36D0nnnn	IMASK	PC		
36E0nnnn		IMASK	PC	
36F0nnnn	PC1		IMASK	PC0
3700nnnn	FL2	FPE		
3710nnnn		FL2	FPE	
3720nnnn			FL2	FPE
3730nnnn	FPE			FL2
3740nnnn	FL1			FP2
3750nnnn	FP2	FL1		
3760nnnn		FP2	FL1	
3770nnnn			FP2	FL1
3780nnnn		FP3	FP1	
3790nnnn			FP3	FP1
37A0nnnn	FP1			FP3
37B0nnnn	FP3	FP1		
37C0nnnn	FL1	FP3	FP1	FP2
37D0nnnn	FP2	FL1	FP3	FP1
37E0nnnn	FP1	FP2	FL1	FP3
37F0nnnn	FP3	FP1	FP2	FL1

REGISTER LOADED IS INDICATED IN BLOCKS

LOAD REGISTER FROM MEMORY

The Load Register instruction loads a byte, a half word, or a word of data from control memory into a register or a group of registers. The effective address of the source is formed using the tag and address fields of the instruction. A hangup will result if the effective address points to a nonexistent location or to a word in a page memory. The register loaded is selected by the destination field, and in the case of byte and half-word loads, the part of the memory word is selected by the shift field. The memory word is shifted left end-around (shifted from bit 31 toward bit 0).

Instruction Format



01 Load low half of group
10 Load high half of group
11 Load high half &low half

000 C Register
001 AS Register
010 BL, DP
011 PC, IMASK
100 FL2, FPE
101 FP2, FL1
110 FP3, FP1
111 FL1, FP3, FP1, FP2

00 No shift
01 Left shift 8 bits
10 Left shift 16 bits
11 Left shift 24 bits

EFFECTIVE ADDRESS

0001	Address
0010	Address + DP
0011	Address + DP, Decr BL
0100	Address + DP, Incr DP
0101	Address + DP, Decr BL &Incr DP
0110	Address + DP, Decr DP
0111	Address + DP, Decr BL &DP
1000	Address + R0
1001	Address + R1
1010	Address + R2
1011	Address + R3
1100	Address + R4
1101	Address + R5
1110	Address + R6
1111	Address + R7

Sample Coding

- Load Low Half of Register Group

Instruction Hex Code (Bits 0-31)	Memory Word (Source)			
	Bits 0-7	Bits 8-15	Bits 16-23	Bits 24-31
320nnnnn(2)	▨	▨	CL	
321nnnnn	C3	▨	▨	C2
322nnnnn	CL		▨	▨
323nnnnn	▨	CL		▨
324nnnnn	▨	▨	ASL	
325nnnnn	AS3	▨	▨	AS2
326nnnnn	ASL		▨	▨
327nnnnn	▨	ASL		▨
328nnnnn	▨	▨	DP	
329nnnnn	DP1	▨	▨	DP0
32Annnnn	DP		▨	▨
32Bnnnnn	▨	DP		▨
32Cnnnnn	▨	▨	▨	IMASK
32Dnnnnn	IMASK	▨	▨	▨
32Ennnnn	▨	IMASK	▨	▨
32Fnnnnn	▨	▨	IMASK	▨
330nnnnn	FL2	▨	▨	▨
331nnnnn	▨	FL2	▨	▨
332nnnnn	▨	▨	FL2	▨
333nnnnn	▨	▨	▨	FL2
334nnnnn	▨	▨	▨	FP2
335nnnnn	FP2	▨	▨	▨
336nnnnn	▨	FP2	▨	▨
337nnnnn	▨	▨	FP2	▨
338nnnnn	▨	▨	FP1	▨
339nnnnn	▨	▨	▨	FP1
33Annnnn	FP1	▨	▨	▨
33Bnnnnn	▨	FP1	▨	▨
33Cnnnnn	▨	▨	FP1	FP2
33Dnnnnn	FP2	▨	▨	FP1
33Ennnnn	FP1	FP2	▨	▨
33Fnnnnn	▨	FP1	FP2	▨

REGISTER LOADED IS INDICATED IN BLOCKS

(2) nnnnn = effective address

LOAD REGISTER FROM MEMORY

Sample Coding (cont)

- Load High Half of Register Group

<table>
<tr><th rowspan="2">Instruction Hex Code (Bits 0-31)</th><th colspan="4">MEMORY WORD (SOURCE)</th></tr>
<tr><th>Bits 0-7</th><th>Bits 8-15</th><th>Bits 16-23</th><th>Bits 24-31</th></tr>
<tr><td>340nnnnn</td><td colspan="2">CH</td><td>▨</td><td>▨</td></tr>
<tr><td>341nnnnn</td><td>▨</td><td colspan="2">CH</td><td>▨</td></tr>
<tr><td>342nnnnn</td><td>▨</td><td>▨</td><td colspan="2">CH</td></tr>
<tr><td>343nnnnn</td><td>C1</td><td>▨</td><td>▨</td><td>C0</td></tr>
<tr><td>344nnnnn</td><td colspan="2">ASH</td><td>▨</td><td>▨</td></tr>
<tr><td>345nnnnn</td><td>▨</td><td colspan="2">ASH</td><td>▨</td></tr>
<tr><td>346nnnnn</td><td>▨</td><td>▨</td><td colspan="2">ASH</td></tr>
<tr><td>347nnnnn</td><td>AS1</td><td>▨</td><td>▨</td><td>AS0</td></tr>
<tr><td>348nnnnn</td><td colspan="2">BL</td><td>▨</td><td>▨</td></tr>
<tr><td>349nnnnn</td><td>▨</td><td colspan="2">BL</td><td>▨</td></tr>
<tr><td>34Annnnn</td><td>▨</td><td>▨</td><td colspan="2">BL</td></tr>
<tr><td>34Bnnnnn</td><td>BL1</td><td>▨</td><td>▨</td><td>BL0</td></tr>
<tr><td>34Cnnnnn</td><td colspan="2">PC</td><td>▨</td><td>▨</td></tr>
<tr><td>34Dnnnnn</td><td>▨</td><td colspan="2">PC</td><td>▨</td></tr>
<tr><td>34Ennnnn</td><td>▨</td><td>▨</td><td colspan="2">PC</td></tr>
<tr><td>34Fnnnnn</td><td>PC1</td><td>▨</td><td>▨</td><td>PC0</td></tr>
<tr><td>350nnnnn</td><td>▨</td><td>FPE</td><td>▨</td><td>▨</td></tr>
<tr><td>351nnnnn</td><td>▨</td><td>▨</td><td>FPE</td><td>▨</td></tr>
<tr><td>352nnnnn</td><td>▨</td><td>▨</td><td>▨</td><td>FPE</td></tr>
<tr><td>353nnnnn</td><td>FPE</td><td>▨</td><td>▨</td><td>▨</td></tr>
<tr><td>354nnnnn</td><td>FL1</td><td>▨</td><td>▨</td><td>▨</td></tr>
<tr><td>355nnnnn</td><td>▨</td><td>FL1</td><td>▨</td><td>▨</td></tr>
<tr><td>356nnnnn</td><td>▨</td><td>▨</td><td>FL1</td><td>▨</td></tr>
<tr><td>357nnnnn</td><td>▨</td><td>▨</td><td>▨</td><td>FL1</td></tr>
<tr><td>358nnnnn</td><td>▨</td><td>FP3</td><td>▨</td><td>▨</td></tr>
<tr><td>359nnnnn</td><td>▨</td><td>▨</td><td>FP3</td><td>▨</td></tr>
<tr><td>35Annnnn</td><td>▨</td><td>▨</td><td>▨</td><td>FP3</td></tr>
<tr><td>35Bnnnnn</td><td>FP3</td><td>▨</td><td>▨</td><td>▨</td></tr>
<tr><td>35Cnnnnn</td><td>FL1</td><td>FP3</td><td>▨</td><td>▨</td></tr>
<tr><td>35Dnnnnn</td><td>▨</td><td>FL1</td><td>FP3</td><td>▨</td></tr>
<tr><td>35Ennnnn</td><td>▨</td><td>▨</td><td>FL1</td><td>FP3</td></tr>
<tr><td>35Fnnnnn</td><td>FP3</td><td>▨</td><td>▨</td><td>FL1</td></tr>
</table>

- Load High and Low Half of Register Group

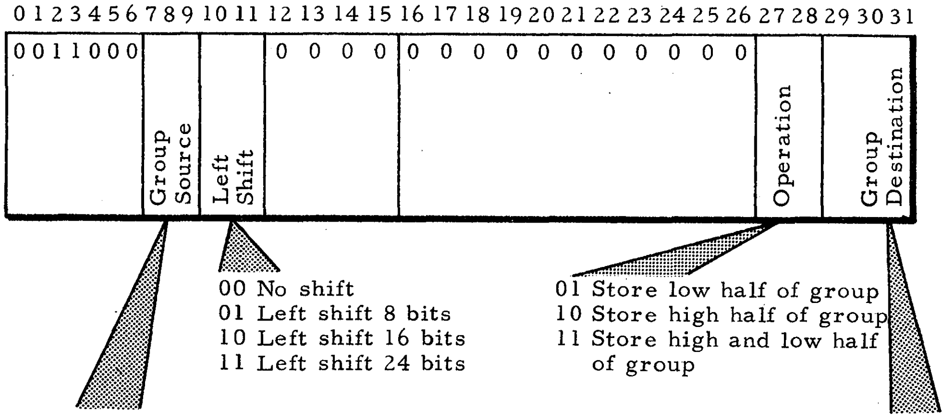
<table>
<tr><th rowspan="2">Instruction Hex Code (Bits 0-31)</th><th colspan="4">MEMORY WORD (SOURCE)</th></tr>
<tr><th>Bits 0-7</th><th>Bits 8-15</th><th>Bits 16-23</th><th>Bits 24-31</th></tr>
<tr><td>360nnnnn</td><td colspan="4">C</td></tr>
<tr><td>361nnnnn</td><td>C3</td><td colspan="2">CH</td><td>C2</td></tr>
<tr><td>362nnnnn</td><td colspan="2">CL</td><td colspan="2">CH</td></tr>
<tr><td>363nnnnn</td><td>C1</td><td colspan="2">CL</td><td>C0</td></tr>
<tr><td>364nnnnn</td><td colspan="4">AS</td></tr>
<tr><td>365nnnnn</td><td>AS3</td><td colspan="2">ASH</td><td>AS2</td></tr>
<tr><td>366nnnnn</td><td colspan="2">ASL</td><td colspan="2">ASH</td></tr>
<tr><td>367nnnnn</td><td>AS1</td><td colspan="2">ASL</td><td>AS0</td></tr>
<tr><td>368nnnnn</td><td colspan="2">BL</td><td colspan="2">DP</td></tr>
<tr><td>369nnnnn</td><td>DP1</td><td colspan="2">BL</td><td>DP0</td></tr>
<tr><td>36Annnnn</td><td colspan="2">DP</td><td colspan="2">BL</td></tr>
<tr><td>36Bnnnnn</td><td>BL1</td><td colspan="2">DP</td><td>BL0</td></tr>
<tr><td>36Cnnnnn</td><td colspan="2">PC</td><td>▨</td><td>▨ IMASK</td></tr>
<tr><td>36Dnnnnn</td><td>▨ IMASK</td><td colspan="2">PC</td><td>▨</td></tr>
<tr><td>36Ennnnn</td><td>▨</td><td>▨ IMASK</td><td colspan="2">PC</td></tr>
<tr><td>36Fnnnnn</td><td>PC1</td><td>▨</td><td>▨ IMASK</td><td>PC0</td></tr>
<tr><td>370nnnnn</td><td>FL2</td><td>FPE</td><td>▨</td><td>▨</td></tr>
<tr><td>371nnnnn</td><td>▨</td><td>FL2</td><td>FPE</td><td>▨</td></tr>
<tr><td>372nnnnn</td><td>▨</td><td>▨</td><td>FL2</td><td>FPE</td></tr>
<tr><td>373nnnnn</td><td>FPE</td><td>▨</td><td>▨</td><td>FL2</td></tr>
<tr><td>374nnnnn</td><td>FL1</td><td>▨</td><td>▨</td><td>FP2</td></tr>
<tr><td>375nnnnn</td><td>FP2</td><td>FL1</td><td>▨</td><td>▨</td></tr>
<tr><td>376nnnnn</td><td>▨</td><td>FP2</td><td>FL1</td><td>▨</td></tr>
<tr><td>377nnnnn</td><td>▨</td><td>▨</td><td>FP2</td><td>FL1</td></tr>
<tr><td>378nnnnn</td><td>▨</td><td>FP3</td><td>FP1</td><td>▨</td></tr>
<tr><td>379nnnnn</td><td>▨</td><td>▨</td><td>FP3</td><td>FL1</td></tr>
<tr><td>37Annnnn</td><td>FP1</td><td>▨</td><td>▨</td><td>FP3</td></tr>
<tr><td>37Bnnnnn</td><td>FP3</td><td>FP1</td><td>▨</td><td>▨</td></tr>
<tr><td>37Cnnnnn</td><td>FL1</td><td>FP3</td><td>FP1</td><td>FP2</td></tr>
<tr><td>37Dnnnnn</td><td>FP2</td><td>FL1</td><td>FP3</td><td>FP1</td></tr>
<tr><td>37Ennnnn</td><td>FP1</td><td>FP2</td><td>FL1</td><td>FP3</td></tr>
<tr><td>37Fnnnnn</td><td>FP3</td><td>FP1</td><td>FP2</td><td>FL1</td></tr>
</table>

REGISTER LOADED IS INDICATED IN BLOCKS

STORE REGISTER TO REGISTER

The Store Register to Register instruction allows the transfer of data between AP control registers. Registers may be transferred individually or in various group combinations.

Instruction Format



00 No shift
01 Left shift 8 bits
10 Left shift 16 bits
11 Left shift 24 bits

01 Store low half of group
10 Store high half of group
11 Store high and low half of group

Group Source:

000 C Register
001 AS Register
010 BL, DP
011 PC, IMASK
100 FL2, FPE
111 FL1, FP3, FP1, FP2

Group Destination:

000 C Register
001 AS Register
010 BL, DP
011 PC, IMASK
100 FL2, FPE
101 FP2, FL1
110 FP3, FP1
111 FL1, FP3, FP1, FP2

Note: To use the table below: 1) Select the Destination Register Option and find the bus positions of the destination (note the Instruction Hex Code for Bits 16-31); 2) Then select the Source Register Option which appears in the same bus position (note the Instruction Hex Code for Bits 0-15).

Sample Coding

Instruction Hex Code	SOURCE REGISTER OPTION			
(Bits 0-15)	Bits 0-7	Bits 8-15	Bits 16-23	Bits 24-31
3 0 0 0	C0	C1	C2	C3
3 0 1 0	C1	C2	C3	C0
3 0 2 0	C2	C3	C0	C1
3 0 3 0	C3	C0	C1	C2
3 0 4 0	AS0	AS1	AS2	AS3
3 0 5 0	AS1	AS2	AS3	AS0
3 0 6 0	AS2	AS3	AS0	AS1
3 0 7 0	AS3	AS0	AS1	AS2
3 0 8 0	BL0	BL1	DP0	DP1
3 0 9 0	BL1	DP0	DP1	BL0
3 0 A 0	DP0	DP1	BL0	BL1
3 0 B 0	DP1	BL0	BL1	DP0
3 0 C 0	PC0	PC1	0	▨ IMASK
3 0 D 0	PC1	0	▨ IMASK	PC0
3 0 E 0	0	▨ IMASK	PC0	PC1
3 0 F 0	▨ IMASK	PC0	PC1	0
3 1 0 0	FL2	FPE	0	0
3 1 1 0	FPE	0	0	FL2
3 1 2 0	0	0	FL2	FPE
3 1 3 0	0	FL2	FPE	0
3 1 C 0	FL1	FP3	FP1	FP2
3 1 D 0	FP3	FP1	FP2	FL1
3 1 E 0	FP1	FP2	FL1	FP3
3 1 F 0	FP2	FL1	FP3	FP1

Instruction Hex Code	DESTINATION REGISTER OPTION			
(Bits 16-31)	Bits 0-7	Bits 8-15	Bits 16-23	Bits 24-31
0 0 0 8	▨	▨	CL	
0 0 0 9	▨	▨	ASL	
0 0 0 A	▨	▨	DP	
0 0 0 B	▨	▨	▨	▨ IMASK
0 0 0 C	FL2	▨	▨	▨
0 0 0 D	▨	▨	▨	FP2
0 0 0 E	▨	▨	FP1	▨
0 0 0 F	▨	▨	FP1	FP2
0 0 1 0	CH		▨	▨
0 0 1 1	ASH		▨	▨
0 0 1 2	BL		▨	▨
0 0 1 3	PC		▨	▨
0 0 1 4	▨	FPE	▨	▨
0 0 1 5	FL1	▨	▨	▨
0 0 1 6	▨	FP3	▨	▨
0 0 1 7	FL1	FP3	▨	▨
0 0 1 8	C			
0 0 1 9	AS			
0 0 1 A	BL		DP	
0 0 1 B	PC		▨	▨ IMASK
0 0 1 C	FL2	FPE	▨	▨
0 0 1 D	FL1	▨	▨	FP2
0 0 1 E	▨	FP3	FP1	▨
0 0 1 F	FL1	FP3	FP1	FP2

STORE REGISTER TO MEMORY

The Store Register stores AP control registers into Bulk core and High Speed data buffer. The effective address is formed by using the tag and address fields. Stores into page memories or nonexistent addresses generate hang-ups. The source may be a register or group of registers. If the group of registers is less than 32 bits, all 32 bits will be stored, with zeros in the unoccupied bit positions.

Instruction Format

0 1 2 3 4 5 6	7 8 9	10 11	12 13 14 15	16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
0 0 1 1 0 0 0	Group (Source)	Left Shift	Tag	Address

Group (Source)	
000	C Register
001	AS Register
010	BL, DP
011	PC, IMASK
100	FL2, FPE
111	FL1, FP3, FP1, FP2

Left Shift	
00	No Shift
01	Left Shift 8 Bits
10	Left Shift 16 Bits
11	Left Shift 24 Bits

EFFECTIVE ADDRESS

Tag	
0001	Address
0010	Address + DP
0011	Address + DP, Decr BL
0100	Address + DP, Incr DP
0101	Address + DP, Decr BL & Incr DP
0110	Address + DP, Decr DP
0111	Address + DP, Decr BL & DP
1000	Address + R0
1001	Address + R1
1010	Address + R2
1011	Address + R3
1100	Address + R4
1101	Address + R5
1110	Address + R6
1111	Address + R7

Sample Coding

	Instruction Hex Code (Bits 0-31)	MEMORY WORD (DESTINATION)				
		Bits 0-7	Bits 8-15	Bits 16-23	Bits 24-31	
	300nnnnn(2)	C0	C1	C2	C3	REGISTER STORED
	301nnnnn	C1	C2	C3	C0	
	302nnnnn	C2	C3	C0	C1	
	303nnnnn	C3	C0	C1	C2	
	304nnnnn	AS0	AS1	AS2	AS3	
	305nnnnn	AS1	AS2	AS3	AS0	
	306nnnnn	AS2	AS3	AS0	AS1	
	307nnnnn	AS3	AS0	AS1	AS2	
	308nnnnn	BLC	BL1	DP0	DP1	
	309nnnnn	BL1	DP0	DP1	BL0	
	30Annnnn	DP0	DP1	BL0	BL1	
	30Bnnnnn	DP1	BL0	BL1	DP0	
SWAP PSW	30Cnnnnn	PC0	PC1	0	▨IMASK	
	30Dnnnnn	PC1	0	▨IMASK	PC0	
	30Ennnnn	0	▨IMASK	PC0	PC1	
	30Fnnnnn	▨IMASK	PC0	PC1	0	
	310nnnnn	FL2	FPE	0	0	
	311nnnnn	FPE	0	0	FL2	
	312nnnnn	0	0	FL2	FPE	
	313nnnnn	0	FL2	FPE	0	
	31Cnnnnn	FL1	FP3	FP1	FP2	
	31Dnnnnn	FP3	FP1	FP2	FL1	
	31Ennnnn	FP1	FP2	FL1	FP3	
	31Fnnnnn	FP2	FL1	FP3	FP1	

(2) nnnnn = effective address

SWAP PSW (Special Case of Store Register Instruction)

The swap PSW operation causes the current program status word (PSW) to be stored in the memory location specified by the effective address and then a new PSW to be fetched from the succeeding location. (If shifting is selected, both the old and new PSW will be shifted.) The result is a change in the Interrupt Mask (IMASK) and the Program Counter (PC). The next instruction is fetched from the location whose address is in the left half of the new PSW. The location where the current PSW is stored should be even and not in a page memory. The new PSW is loaded from the next (odd) location.

Instruction Format

0 1 2 3 4 5 6 7 8 9	10 11	12 13 14 15	16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
0 0 1 1 0 0 0 0 1 1	Left Shift	Tag	Address

0 0	No Shift
0 1	Left Shift 8 Bits
1 0	Left Shift 16 Bits
1 1	Left Shift 24 Bits

EFFECTIVE ADDRESS

0001	Address
0010	Address + DP
0011	Address + DP, Decr
0100	Address + DP, Incr DP
0101	Address + DP, Decr BL & Incr DP
0110	Address + DP, Decr DP
0111	Address + DP, Decr BL & DP
1000	Address + R0
1011	Address + R1
1100	Address + R2
1100	Address + R3
1100	Address + R4
1100	Address + R5
1110	Address + R6
1111	Address + R7

Sample Coding

Instruction Hex Code (Bits 0-31)	MEMORY WORD (DESTINATION)			
	Bits 0-7	Bits 8-15	Bits 16-23	Bits 24-31
30Cnnnnn[2]	PC0	PC1	0	0 IMASK
30Dnnnnn	PC1	0	0 IMASK	PC0
30Ennnnn	0	0 IMASK	PC0	PC1
30Fnnnnn	0 IMASK	PC0	PC1	0

[2] nnnnn = effective address

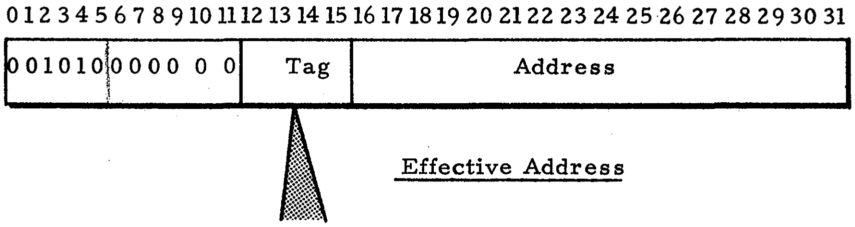
BRANCH/LOOP INSTRUCTIONS

The normal sequence of instruction execution can be modified by a branch operation, a branch and link operation, or a loop operation.

UNCONDITIONAL BRANCH INSTRUCTION

The unconditional branch instruction causes the effective address to be put into the program counter. The next operation is executed from the effective address.

Instruction Format



Effective Address

0001	Address
0010	Address + DP
0011	Address + DP, Decr BL
0100	Address + DP, Incr DP
0101	Address + DP, Decr BL & Incr DP
0110	Address + DP, Decr DP
0111	Address + DP, Decr BL & DP
1000	Address + R0
1001	Address + R1
1010	Address + R2
1011	Address + R3
1100	Address + R4
1101	Address + R5
1110	Address + R6
1111	Address + R7

Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
280nnnnn[2]	Branch unconditionally
281nnnnn[2]	No branch

[2] Effective Address

CONDITIONAL BRANCH INSTRUCTION

If the condition coded in the condition field is true the Conditional Branch operation acts like an unconditional branch. If the condition is not true, no change in control occurs and the next sequential instruction is executed. The effective branch address is a function of the address and tag fields of the instruction. This permits address modification by the DP register and the Branch and Link registers. If the tag field specifies modification of the DP or BL, these changes will occur regardless of the condition.

Instruction Format

0 1 2 3 4 5	6 7 8 9 10 11	12 13 14 15	16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
0 0 1 0 1 0	Condition Code	Tag	Address

	Condition
000001	(No Branch)
000010	FP1=0
000011	FP1≠0
000100	FP2=0
000101	FP2≠0
000110	FP3=0
000111	FP3≠0
001000	FL1=0
001001	FL1≠0
001010	FL2=0
001011	FL2≠0
001100	Selected bit of C register = 0(2)
001101	Selected bit of C register = 1(2)
001110	All Y register bits in enabled arrays(3)=0
001111	Any Y register bits in any enabled array(3)=0
010000	BL=0
010001	BL≠0
010010	FPE=0
010011	FPE≠0
010100	DP=0
010101	DP≠0

	Effective Address			
0 0 0 0	Address			
0 0 0 1	Address			
0 0 1 0	Address	+ DP		
0 0 1 1	Address	+ DP,	Decr BL	
0 1 0 0	Address	+ DP,	Incr DP	
0 1 0 1	Address	+ DP,	Decr BL	Incr DP
0 1 1 0	Address	+ DP,	Decr DP	
0 1 1 1	Address	+ DP,	Decr BL	DP
1 0 0 0	Address	+ R0		
1 0 0 1	Address	+ R1		
1 0 1 0	Address	+ R2		
1 0 1 1	Address	+ R3		
1 1 0 0	Address	+ R4		
1 1 0 1	Address	+ R5		
1 1 1 0	Address	+ R6		
1 1 1 1	Address	+ R7		

(2) Rightmost five bits of FP1 contain address of selected bit in Common Register..

(3) AS register selects enabled arrays.

Sample Coding

- Branch if Condition is Satisfied

Instruction Hex Code (Bits 0-31)	Operation Branch Condition
282nnnnn[3]	FP1=0
283nnnnn	FP1≠0
284nnnnn	FP2=0
285nnnnn	FP2≠0
286nnnnn	FP3=0
287nnnnn	FP3≠0
288nnnnn	FL1=0
289nnnnn	FL1≠0
28Annnnn	FL2=0
28Bnnnnn	FL2≠0
28Cnnnnn	Common Register Bit = 0[1]
28Dnnnnn	Common Register Bit = 1[1]
28Ennnnn	If all Y register bits in all selected arrays[2] =0
28Fnnnnn	If any Y register bit in any selected array[2] =1
290nnnnn	BL=0
291nnnnn	BL≠0
292nnnnn	FPE=0
293nnnnn	FPE≠0
294nnnnn	DP=0
295nnnnn	DP≠0

[1] Rightmost five bits of FP1 contain address of selected bit in Common Register.

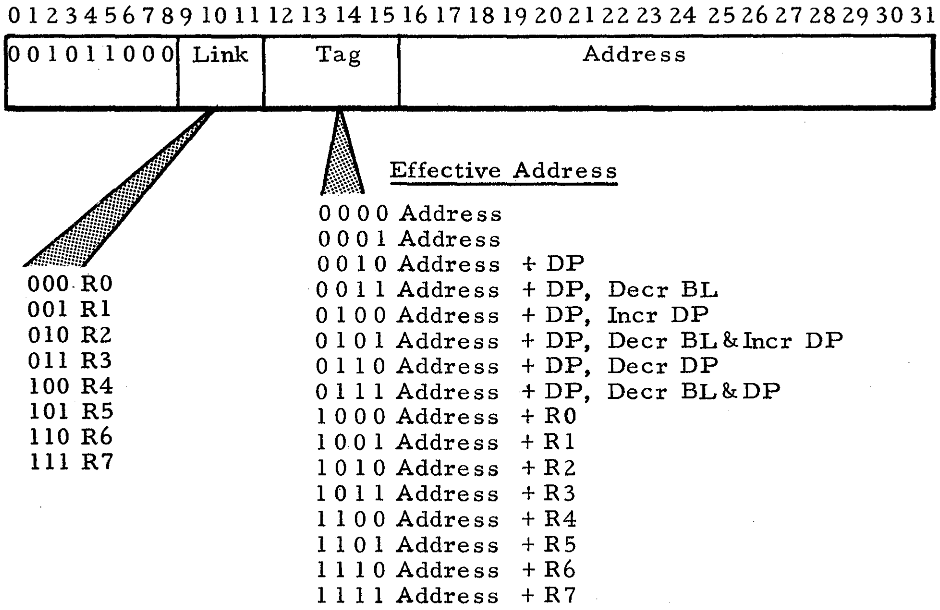
[2] AS register selects enabled arrays.

[3] nnnnn = Effective Address

BRANCH AND LINK INSTRUCTION

Branch and Link instructions cause the contents of the current program counter to be stored in the right half of a Branch and Link Register, with zeros stored in the left half. (The current program counter will contain the address of the next sequential instruction after the Branch and Link instruction.) Then the effective address in the Branch and Link instruction is loaded into the program counter.

Instruction Format



0 1 2 3 4 5 6 7 8	9 10 11	12 13 14 15	16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
0 0 1 0 1 1 0 0 0	Link	Tag	Address

Link	
000	R0
001	R1
010	R2
011	R3
100	R4
101	R5
110	R6
111	R7

Tag	Effective Address
0 0 0 0	Address
0 0 0 1	Address
0 0 1 0	Address + DP
0 0 1 1	Address + DP, Decr BL
0 1 0 0	Address + DP, Incr DP
0 1 0 1	Address + DP, Decr BL & Incr DP
0 1 1 0	Address + DP, Decr DP
0 1 1 1	Address + DP, Decr BL & DP
1 0 0 0	Address + R0
1 0 0 1	Address + R1
1 0 1 0	Address + R2
1 0 1 1	Address + R3
1 1 0 0	Address + R4
1 1 0 1	Address + R5
1 1 1 0	Address + R6
1 1 1 1	Address + R7

Sample Coding

Instruction Hex Code (Bits 0-31)	PROGRAM COUNTER STORED IN REGISTERS	
	Register	Address (Hex)
2C0nnnnn[2]	R0	0600
2C1nnnnn	R1	0601
2C2nnnnn	R2	0602
2C3nnnnn	R3	0603
2C4nnnnn	R4	0604
2C5nnnnn	R5	0605
2C6nnnnn	R6	0606
2C7nnnnn	R7	0607

[2] nnnnn = Effective Address

LOOP INSTRUCTION

The Loop instruction permits the repetitive execution of one or more sequential instructions. Before the Loop operation is performed, FL1 should contain the number of repetitions minus 1 (i.e., if 10 repetitions are required FL1 should contain a 9).

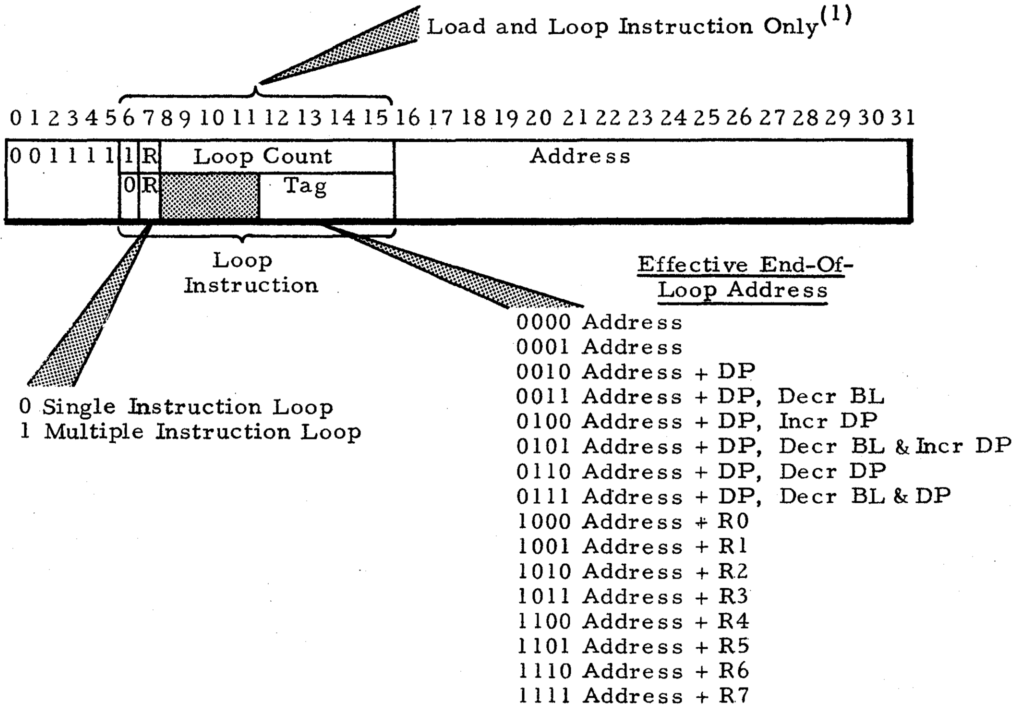
LOAD AND LOOP INSTRUCTION

The Load and Loop operation is identical to the Loop operation,except FL1 need not be preloaded with the repetition count. This operation permits loading FL1 with the count in the same instruction. In both Loop and Load and Loop instructions, the end-of-loop address is the effective address.

Restrictions

Instructions that alter the Program Counter, i.e., branches, skips, external functions, etc., should not be used within a loop. No instruction within the loop should alter FL1.

Instruction Format



	Effective End-Of-Loop Address
0000	Address
0001	Address
0010	Address + DP
0011	Address + DP, Decr BL
0100	Address + DP, Incr DP
0101	Address + DP, Decr BL & Incr DP
0110	Address + DP, Decr DP
0111	Address + DP, Decr BL & DP
1000	Address + R0
1001	Address + R1
1010	Address + R2
1011	Address + R3
1100	Address + R4
1101	Address + R5
1110	Address + R6
1111	Address + R7

[1] With Load and Loop operation, no address-tag modification is permitted.

Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
3C0nnnnn[2]	Single instruction Loop
3D0nnnnn	Multi-instruction Loop
(Bits 0-7)[3]	
3Ekknnnn	Single instruction Load and Loop
3Fkknnnn	Multi-instruction Load and Loop

[2] Effective Address

[3] kk = Loop Count
nnnn = Address

ASSOCIATIVE ARRAY INSTRUCTIONS

Associative array instructions perform operations on the arrays. Each array contains 65,536 bits, organized as a matrix of 256 words by 256 bits. Access may be made in either the bit or word direction. Each array also contains 256 response-store elements.

RESPONSE STORE ELEMENTS (M, X, Y)

Included in the response store elements for each array are the M, X, and Y response store registers. The M, X, and Y registers are each 256 bits in length.

ARRAY SELECT

The associative array instructions operate on all arrays enabled by the Array Select register (32-bit register) or only on the single array specified by the contents of field pointer 1 (link pointer mode). Therefore the Array Select register must be set for the arrays that will be active in the associative instructions. A 1 in bit 0 means array 0 is enabled, a 1 in bit 1 means array 1 is enabled, etc.

ASSOCIATIVE ARRAY OPERATIONS

The associative instruction permits the following operations within the associative arrays:

1) Load Response Store Registers
2) Store Response Store Registers to Array Memory
3) Mask Operations
4) Common Register Operations
5) Resolve Operations

These five operations are discussed in detail in following sections.

WORD/BIT MODE

A mode bit in the instruction format selects either word or bit-slice mode. With the mode bit the programmer can reverse the meaning of bits or words at any time. This allows data to be accessed vertically and horizontally. If this bit is set to one, data will be accessed horizontally (word mode); if it is zero, data will be accessed vertically (bit-slice mode).

FLIP MODE

The flip bit must be set to 1 when loading from array memory to the X and Y registers or storing from the X, Y, and M registers into array memory to permit accessing the data in both directions (i.e., word or bit mode). However, if the flip bit is 0, access is confined to only one direction when loading or storing data. When loading into the M register from memory, the flip bit = 0 if the data was previously stored in flip mode.

ARRAY ADDRESS

The address of the array bit slice or array word slice may be specified directly in the instruction format or indirectly in a field pointer. When specified indirectly, the selected field pointer may be incremented or decremented to permit automatic stepping through the bits of a data field.

ASSOCIATIVE ARRAY INSTRUCTIONS

LOGIC FUNCTION

The logic functions performed on the X and Y registers are specified directly in the Load Response Store instruction format. Two logic functions can be specified in the instruction format. If the selected Common Register bit equals 1, the first logic function is enabled; if the Common Register bit equals 0, the second logic function is enabled. (The low-order five bits of Field Pointer 1 selects a bit of the Common Register.) If the first and second logic functions are identical, the common register bit is not considered.

SHIFTING

Shifting may also be specified in some associative operations directly in the instruction format or indirectly in a field pointer (FPE). A shift amount of n with a modulus of m causes the network to divide the 256 bit quantity into 256/m groups of m bits each. Within each group an end-around-right shift (shift from high to low order, i. e., from bit 0 toward bit 255) of n places occurs. (Refer to Shift Constant Table 2-1.) Both n and m must be powers of 2.

MIRRORING

Mirroring the input is also possible in some associative operations. The mirroring will cause the 256-bit input quantity to flip end-for-end (bit i is put into bit 255-i) before shifting as specified by the shift constant.

LEFT SHIFT

A mirror, a shift of n places,and a mirror again results in a left shift (shift from low order to high order, i. e., from bit 255 toward bit 0) of n places.

INPUT SOURCE

In each associative array instruction an input is selected from the following:

1) A bit slice containing one bit from each of the 256 words in array memory

2) A word slice containing all 256 bits in one array word
3) The contents of the 256-bit M Register (MASK)
4) The contents of the 256-bit X Register
5) The contents of the 256-bit Y Register
6) The contents of the 32-bit Common Register in bits 0 to 31, with zeros in bits 32 to 255

*7) A 256-bit input from the parallel input/output channel of the custom I/O cabinet

*Optional Parallel I/O

DESTINATION

The selected input is transmitted through a network, which may shift or permute it in various ways as controlled by the instruction. The 256-bit output is then communicated to one of the following:

1) A bit slice containing one bit from each of the 256 words in array memory
2) A word slice containing all 256 bits in one array word
3) The M Register (MASK)
4) The Y Register, which may combine it logically with the current contents of Y register
5) The X Register, which may combine it logically with the current contents of the X register or with the contents of the Y register to modify the contents of the X register
6) The Common Register, which receives bits 0 to 31 from the source and ignores bits 32 to 255

*7) A 256-bit parallel output to the parallel I/O channel of the custom I/O cabinet

*Optional Parallel I/O

LOAD RESPONSE STORE REGISTERS

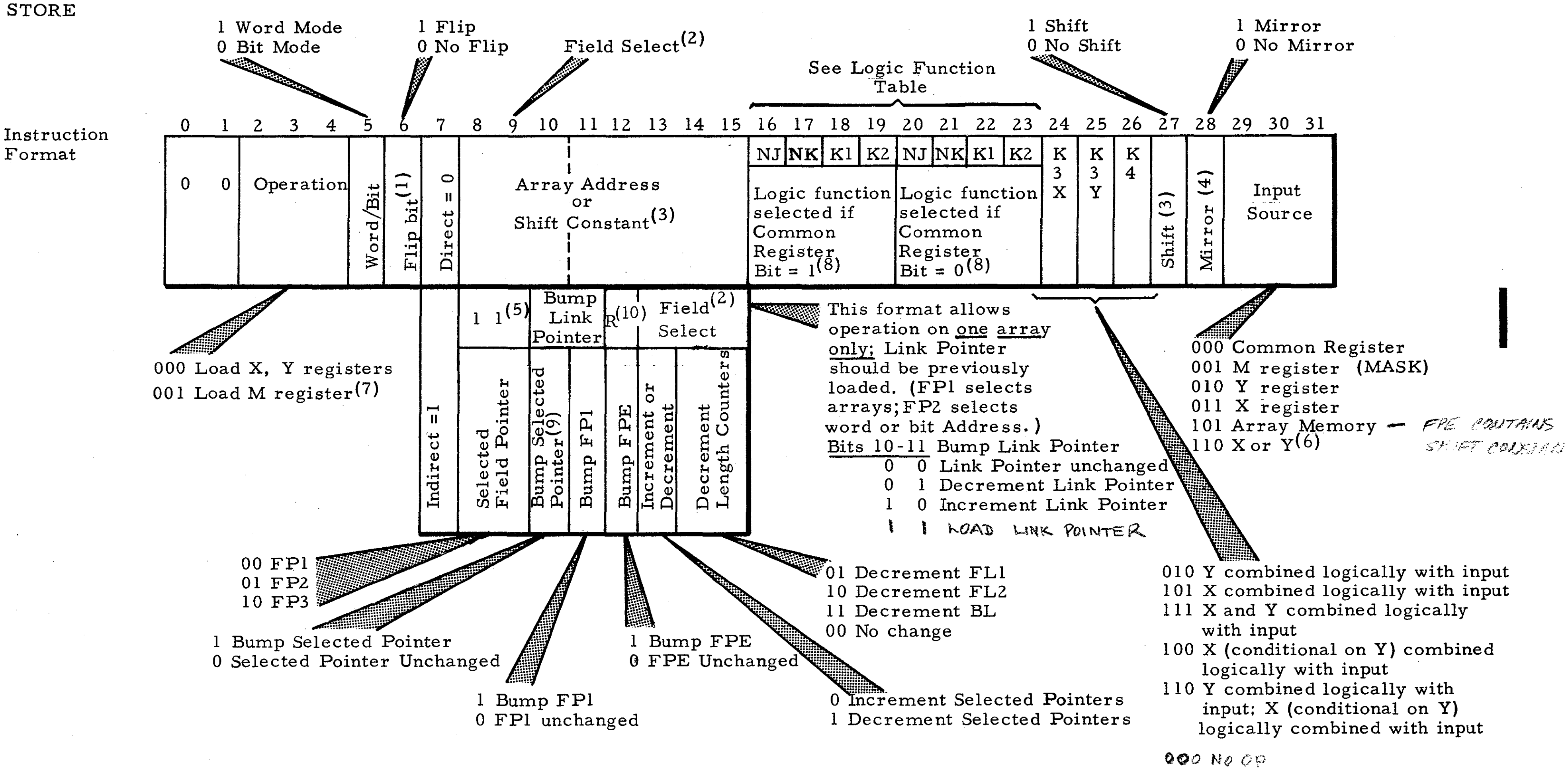
The Load Response Store instruction loads a response store register from an array bit column or word, one of the Response Store registers ($M^{(1)}$, X, or Y), or the 32-bit Common Register (0-31 bits) with zeros padded in bits (32 to 255). The destination of the data may be one of the response store registers ($M^{(1)}$, X or Y). Logic may be performed simultaneously with the loading operation in the X and Y registers only. The source data input is combined logically with the previous contents of the X and Y registers to produce new values in the X and Y registers. This instruction is also used to increment and decrement FP1, FP2, FP3, and FPE, and decrement FL1, FL2, and BL.

For those operations containing a shift constant, the input network may shift the input (see shift constant table 2-1).

When the mirror bit is set to 1, the input network flips the input quantity end-for-end (bit i is put into position 255-i) before shifting it as specified by the shift constant.

(1) Details on operations involving M response store register are discussed in the Mask Operation section.

LOAD RESPONSE STORE



This format allows operation on one array only; Link Pointer should be previously loaded. (FP1 selects arrays; FP2 selects word or bit Address.)

Bits 10-11 Bump Link Pointer

0 0	Link Pointer unchanged
0 1	Decrement Link Pointer
1 0	Increment Link Pointer
1 1	LOAD LINK POINTER

Notes:

1. When loading X or Y, flip bit = 1 when input source is Array memory, if data was stored to memory with the flip bit = 1.
2. When Common Register is input source, this field may be used to select a 32 bit field in the response store. See field select table
3. See shift constant table 2-1.
4. Flipping occurs before mirror or shift; mirror occurs before shift.
5. Link pointer (FP1 and FP2 linked).
6. If all Y register bits in all selected arrays are zero, the X register is input source; if any Y register bits in any selected array is one, the Y register is input source.
7. Detail discussion in Mask Operation Section.
8. Rightmost five bits of FP1 selects a bit of Common Register.
9. When FP1 is selected pointer, bit 10 must equal bit 11.
10. Resolve function. THIS MEANS THAT INPUT SOURCE IS RESOLVER

Bits 8, 9, 10	Field Selected
0 0 0	0 (Bits 0-31)
0 0 1	1 (Bits 32-63)
0 1 0	2 (Bits 64-95)
0 1 1	3 (Bits 96-127)
1 0 0	4 (Bits 128-159)
1 0 1	5 (Bits 160-191)
1 1 0	6 (Bits 192-223)
1 1 1	7 (Bits 224-255)

Instruction Format (cont)

- Logic Function Table

	COL				24	25	26	24	25	26	24	25	26	24	25	26	24	25	26
Common Register Bit = 1	16	17	18	19	K3X	K3Y	K4	K3X	K3Y	K4	K3X	K3Y	K4	K3X	K3Y	K4	K3X	K3Y	K4
Common Register Bit = 0	20	21	22	23	0	1	0	1	0	1	1	1	1	1	0	0	1	1	0
Logic Function	NJ	NK	K1	K2	New State of			New State of			New State of			New State of			New State of		
					X	Y		X	Y		X	Y		X		Y	X		Y
Exclusive OR Complement F	0	0	0	0	X	$Y\oplus\overline{F}$		$X\oplus\overline{F}$	Y		$X\oplus\overline{F}$	$Y\oplus\overline{F}$		$X\oplus Y\overline{F}$		Y	$X\oplus Y\overline{F}$		$Y\oplus\overline{F}$
Inclusive OR Complement F	0	0	0	1	X	$Yv\overline{F}$		$Xv\overline{F}$	Y		$Xv\overline{F}$	$Yv\overline{F}$		$XvY\overline{F}$		Y	$XvY\overline{F}$		$Yv\overline{F}$
Logical AND	0	0	1	0	X	YF		XF	Y		XF	YF		$\overline{Y}XvXF$		Y	$\overline{Y}XvXF$		YF
NO-Op	0	0	1	1	X	Y		X	Y		X	Y		X		Y	X		Y
Load Complement F	0	1	0	0	X	$\overline{F}$		$\overline{F}$	Y		$\overline{F}$	$\overline{F}$		$\overline{Y}XvY\overline{F}$		Y	$\overline{Y}XvY\overline{F}$		$\overline{F}$
NOT-Inclusive OR	0	1	0	1	X	$\overline{Y}\overline{F}$		$\overline{X}\overline{F}$	Y		$\overline{X}\overline{F}$	$\overline{Y}\overline{F}$		$\overline{Y}XvY\overline{X}\overline{F}$		Y	$\overline{Y}XvY\overline{X}\overline{F}$		$\overline{Y}\overline{F}$
AND Complement F	0	1	1	0	X	$Y\overline{F}$		$X\overline{F}$	Y		$X\overline{F}$	$Y\overline{F}$		$\overline{Y}XvX\overline{F}$		Y	$\overline{Y}XvX\overline{F}$		$Y\overline{F}$
Clear to Zero	0	1	1	1	X	0		0	Y		0	0		$\overline{Y}X$		Y	$\overline{Y}X$		0
Input (F)	1	0	0	0	X	F		F	Y		F	F		$\overline{Y}XvYF$		Y	$\overline{Y}XvYF$		F
Inclusive OR	1	0	0	1	X	YvF		XvF	Y		XvF	YvF		XvYF		Y	XvYF		YvF
NOT AND Complement F	1	0	1	0	X	$\overline{Y}vF$		$\overline{X}vF$	Y		$\overline{X}vF$	$\overline{Y}vF$		$\overline{Y}XvYFvY\overline{X}$		Y	$\overline{Y}XvYFvY\overline{X}$		$\overline{Y}vF$
SET to One	1	0	1	1	X	1		1	Y		1	1		XvY		Y	XvY		1
Exclusive OR	1	1	0	0	X	$Y\oplus F$		$X\oplus F$	Y		$X\oplus F$	$Y\oplus F$		$X\oplus YF$		Y	$X\oplus YF$		$Y\oplus F$
NOT Inclusive OR Complement F	1	1	0	1	X	$\overline{Y}F$		$\overline{X}F$	Y		$\overline{X}F$	$\overline{Y}F$		$\overline{Y}XvY\overline{X}F$		Y	$\overline{Y}XvY\overline{X}F$		$\overline{Y}F$
NOT AND	1	1	1	0	X	$\overline{Y}v\overline{F}$		$\overline{X}v\overline{F}$	Y		$\overline{X}v\overline{F}$	$\overline{Y}v\overline{F}$		$\overline{Y}XvY\overline{X}vY\overline{F}$		Y	$\overline{Y}XvY\overline{X}vY\overline{F}$		$\overline{Y}v\overline{F}$
NEGATE	1	1	1	1	X	$\overline{Y}$		$\overline{X}$	Y		$\overline{X}$	$\overline{Y}$		$Y\oplus X$		Y	$Y\oplus X$		$\overline{Y}$

$\oplus$ Exclusive OR

v Inclusive OR

- Complementation

F : Bit from input network (Source determined by bits 29-31 of Associative Instruction Format)

X : Old State of X - Response Store Register

Y : Old State of Y - Response Store Register

Table 2-1. Shift Constants For Associative Instructions (Bit Positions 8 thru 15 of Instruction)

Shift Constants

Hexadecimal Shift Constant	Modulus	Shift Amount	Shift Constant							
			8	9	10	11	12	13	14	15
00	—	0	0	0	0	0	0	0	0	0
80	256	128	1	0	0	0	0	0	0	0
C0		64	1	1	0	0	0	0	0	0
E0		32	1	1	1	0	0	0	0	0
F0		16	1	1	1	1	0	0	0	0
F8		8	1	1	1	1	1	0	0	0
FC		4	1	1	1	1	1	1	0	0
FE		2	1	1	1	1	1	1	1	0
FF		1	1	1	1	1	1	1	1	1
40	128	64	0	1	0	0	0	0	0	0
60		32	0	1	1	0	0	0	0	0
70		16	0	1	1	1	0	0	0	0
78		8	0	1	1	1	1	0	0	0
7C		4	0	1	1	1	1	1	0	0
7E		2	0	1	1	1	1	1	1	0
7F		1	0	1	1	1	1	1	1	1
20	64	32	0	0	1	0	0	0	0	0
30		16	0	0	1	1	0	0	0	0
38		8	0	0	1	1	1	0	0	0
3C		4	0	0	1	1	1	1	0	0
3E		2	0	0	1	1	1	1	1	0
3F		1	0	0	1	1	1	1	1	1
10	32	16	0	0	0	1	0	0	0	0
18		8	0	0	0	1	1	0	0	0
1C		4	0	0	0	1	1	1	0	0
1E		2	0	0	0	1	1	1	1	0
1F		1	0	0	0	1	1	1	1	1
08	16	8	0	0	0	0	1	0	0	0
0C		4	0	0	0	0	1	1	0	0
0E		2	0	0	0	0	1	1	1	0
0F		1	0	0	0	0	1	1	1	1
04	8	4	0	0	0	0	0	1	0	0
06		2	0	0	0	0	0	1	1	0
07		1	0	0	0	0	0	1	1	1
02	4	2	0	0	0	0	0	0	1	0
03		1	0	0	0	0	0	0	1	1
01	2	1	0	0	0	0	0	0	0	1

Note: A shift amount of n with a modulus of m divides the 256-bit quantity into 256/m groups of m bits each. Within each group an end-around-right shift of n places occurs.

LOAD RESPONSE STORE REGISTERS

Sample Coding (cont)

- Combine X&Y Registers Logically (Operation in all Arrays Selected by Array select Register)

The Load Response Store instruction permits logic functions of X and Y, with results stored into X and/or Y. There are 16 functions of X and Y. To put any function into X,and simultaneously put any function into Y, requires 256 combinations. Sixty-four of the combinations are obtainable with one instruction. Of the 64, one combination leaves X and Y alone, 15 modify X only, and 15 modify Y only. These combinations are shown in the first two examples. The other 33 combinations modify both X and Y and are shown in the third example.

- • Leave Y Alone and Load X With Logic Function

Instruction Hex Code (Bits 0-31)	Logic Function
000033A2	X
000088A2	Y
000077A2	0
0000BBA2	1
0000FFA2	$\overline{X}$
000044A2	$\overline{Y}$
000000A2	$\overline{X}\oplus Y$
0000CCA2	$X\oplus Y$
000055A2	$\overline{X}\overline{Y}$
0000DDA2	$\overline{X}Y$
000066A2	$X\overline{Y}$
000022A2	XY
0000EEA2	$\overline{X}v\overline{Y}$
0000AAA2	$\overline{X}vY$
000011A2	$Xv\overline{Y}$
000099A2	XvY

LOAD
RESPONSE
STORE
REGISTERS

Sample
Coding
(cont)

- • • Leave X Alone and Load Y With Logic Function

Instruction Hex Code (Bits 0-31)	Logic Function
00008843	X
00003343	Y
00007743	0
0000BB43	1
00004443	$\overline{X}$
0000FF43	$\overline{Y}$
00000043	$\overline{X}\oplus Y$
0000CC43	$X\oplus Y$
00005543	$\overline{X}\overline{Y}$
00006643	$\overline{X}Y$
0000DD43	$X\overline{Y}$
00002243	XY
0000EE43	$\overline{X}v\overline{Y}$
00001 143	$\overline{X}vY$
0000AA43	$Xv\overline{Y}$
00009943	XvY

- • • Simultaneous X, Y Operations (Instruction Hex Code for bits 0-15 is 0000.)

(1) Instruction Hex Code (Bits 16-31)		NEW STATE OF X												
		0	1	$\overline{X}$	$\overline{Y}$	$\overline{X}\oplus Y$	$X\oplus Y$	$\overline{X}\overline{Y}$	$\overline{X}Y$	$X\overline{Y}$	$\overline{X}v\overline{Y}$	$\overline{X}vY$	$Xv\overline{Y}$	XvY
NEW STATE OF Y	0	77E3	▨	▨	▨	▨	CCE2	▨	DDE2	77C3	▨	▨	▨	▨
	1	▨	BBE3	▨	▨	00E2	▨	▨	▨	▨	▨	AAE2	11E2	BBC3
	$\overline{X}$	▨	▨	44E3	▨	▨	44C3	▨	▨	▨	▨	▨	▨	▨
	$\overline{Y}$	▨	▨	FFE3	44E2	▨	FFC3	55E2	▨	44C2	EEE2	▨	▨	▨
	$\overline{X}\oplus Y$	▨	00E3	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	00C3
	$X\oplus Y$	CCE3	▨	▨	▨	▨	▨	▨	▨	CCC3	▨	▨	▨	▨
	$\overline{X}\overline{Y}$	▨	▨	55E3	▨	▨	55C3	▨	▨	▨	▨	▨	▨	▨
	$\overline{X}Y$	66E3	▨	▨	▨	▨	▨	▨	▨	66C3	▨	▨	▨	▨
	$X\overline{Y}$	DDE3	▨	▨	▨	▨	▨	▨	▨	DDC3	▨	▨	▨	▨
	$\overline{X}v\overline{Y}$	▨	▨	EEE3	▨	▨	EEC3	▨	▨	▨	▨	▨	▨	▨
	$\overline{X}vY$	▨	11E3	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	11C3
	$Xv\overline{Y}$	▨	AAE3	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	AAC3

Shaded entries indicate manipulations not possible with one instruction.

(1) Hex Code for bits 16 - 31 in table blocks above.

LOAD RESPONSE STORE REGISTERS

Sample Coding (cont)

- Load X & Y Response Store Registers In All Selected Arrays (Arrays selected by AS register)

- Load X & Y In One Array (Rightmost 5 bits of FP1 selects array; FP2 contains bit or word address)

Instruction Hex Code (Bits 0-31)			Operation
Load X in all selected arrays (ff=logic)[3]	Load Y in all selected arrays (ff=logic)[3]	Load X&Y in all selected arrays (ff=logic)[3]	Load selected Response Store Register from:
02nnffA5	02nnff45	02nnffE5	Array bit column nn
06nnffA5	06nnff45	06nnffE5	Array word nn
03iiffA5	03iiff45	03iiffE5	Array bit column indirectly through field pointer (ii = indirect coding)[2]
07iiffA5	07iiff45	07iiffE5	Array word indirectly through field pointer (ii = indirect coding)[2]
0000ffA1	0000ff41	0000ffE1	M register
0000ffA2	0000ff42	0000ffE2	Y register
0000ffA3	0000ff43	0000ffE3	X register
02s0ffA0	02s0ff40	02s0ffE0	Common register into one 32 bit field(s)[1]· other fields cleared.
Clear X	Clear Y	Clear X&Y	
000077A3	00007742	000077E2	
Set X	Set Y	Set X&Y	
0000BBA3	0000BB42	0000BBE2	
Load X in selected Array (ff=logic)[3]	Load Y in selected Array (ff=logic)[3]	Load X & Y in selected Array (ff=logic)[3]	Load selected Response Store Register From:
03CyffA0	03Cyff40	03CyffE0	Common Register into one 32-bit field (y); other fields cleared
03CyffA5	03Cyff45	03CyffE5	Array bit column nn (Field y)
07CyffA5	07Cyff45	07CyffE5	Array word nn (Field y)

(1) Field Select Table

s	Field (y)	Bits
0	0	0-31
2	1	32-63
4	2	64-95
6	3	96-127
8	4	128-159
A	5	160-191
C	6	192-223
E	7	224-255

(3) Logic Code
(no Common Register bit selected)

ff	Logic Function
00	Exclusive OR Complement Input
11	Inclusive OR Complement Input
22	Logical AND
33	No-op
44	Complement Input
55	NOT Inclusive OR
66	AND Complement Input
77	Clear to zero
88	Input
99	Inclusive OR
AA	NOT AND Complement Input
BB	Set to one
CC	Exclusive OR
DD	NOT Inclusive OR Complement Input
EE	NOT AND
FF	NEGATE

(2) Indirect Code ii

i	Selected Field Pointer	Bump	i	Incr or Decr		Decr Field Length Counter		
0	FP1	–	0	Incr		-		
1	FP1	FP1	1	Incr		FL1		
2	FP1	FP1	2	Incr		FL2		
3	FP1	FP1	3	Incr		BL		
4	FP2	-	4	Decr		-		
5	FP2	FP1	5	Decr		FL1		
6	FP2	FP2	6	Decr		FL2		
7	FP2	FP1&FP2	7	Decr		BL		
8	FP3	–	8	Incr	BUMP FPE	-		* RESOLVER
9	FP3	FP1	9	Incr		FL1		
A	FP3	FP3	A	Incr		FL2	STEP	
B	FP3	FP1&FP3	B	Incr		BL		
C	LP(a)	-	C	Decr		-		
D	LP	Decr LP	D	Decr		FL1		
E	LP	Incr LP	E	Decr		FL2		
F*	LP	Load Link	F	Decr		BL		

(a) LP = Link Pointer

- Miscellaneous Operations (For this Instruction Format)

Instruction Hex Code (Bits 0-31)	Operation
00E088B2	Load X with Y shifted right 32 bits, modulus 256
0000884B	Load Y with X mirrored (bit 0 of X moves to bit 255 of Y, bit 255 of X moves to bit 0 of Y, etc.)
01340000	Decrement FP1
01A00000	Increment FP3
01010000	Decrement FL1
01030000	Decrement BL
007F88B3	Shift X right 1 bit, modulus 128
001E8852	Shift Y right 2 bits, modulus 32
02nn9CA5	Load X with current X inclusive ORed with bit column nn if selected common register bit equals 1 or load X with current X exclusive ORed with bit column nn if selected common register bit equals 0.
000088A6	Load X with input (If all Y register bits in all arrays enabled by the array select register are zero then the X register is input; if any Y register bits in any enabled array is one, the Y register is input)
000088E6	Load X with Y if any bit of Y in any selected array is 1. Load Y with X if all bits of Y in all selected arrays are 0.

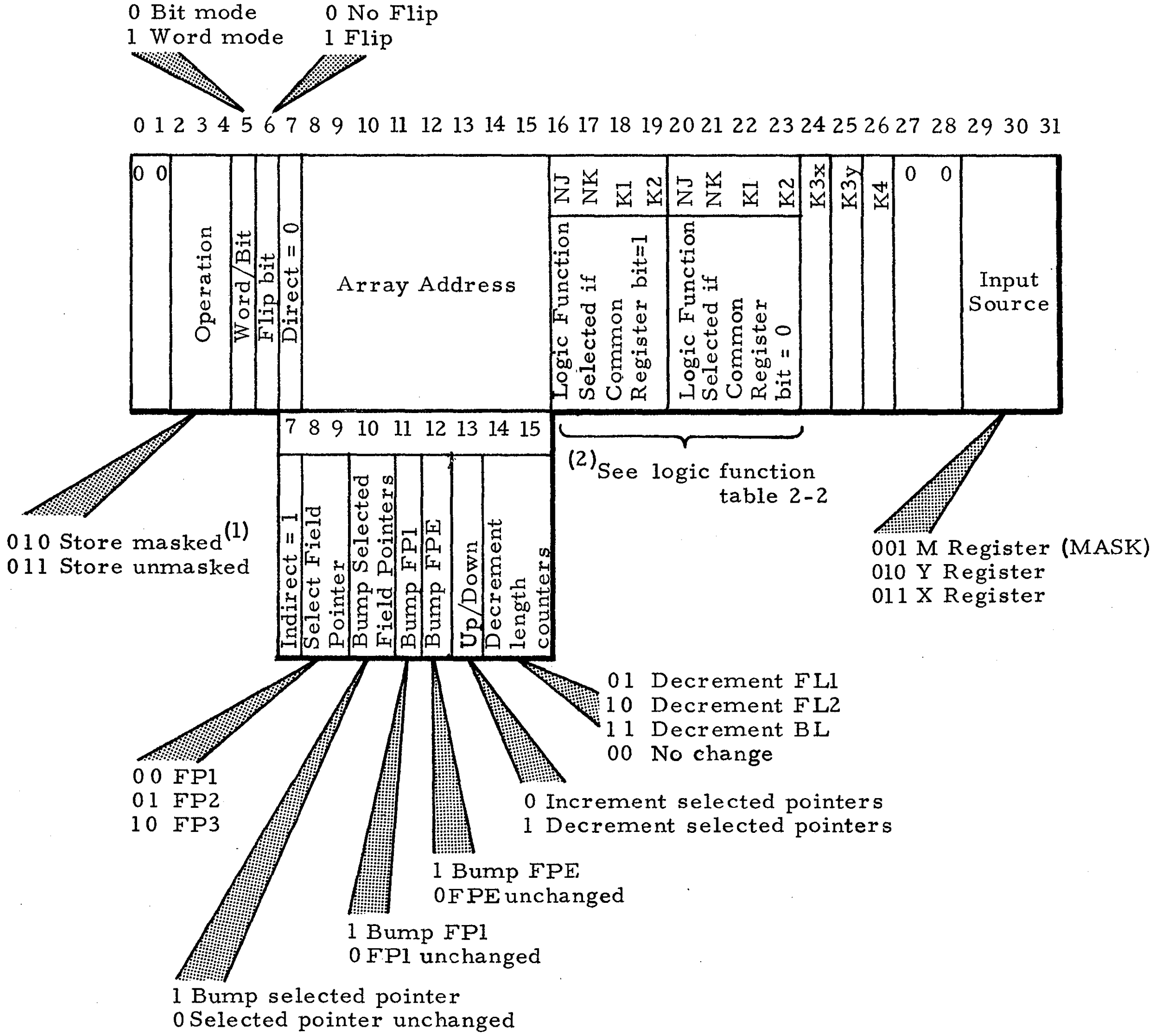
STORE RESPONSE STORE REGISTERS TO ARRAY MEMORY

This instruction allows the 256-bit response store quantities to be stored in array words or bit columns in all selected arrays. When storing into the array, the flip bit should be equal to 1 if it is necessary to access data in both word and bit mode.

Store Masked

Masked store operations are permitted from the X and Y response store registers. A Setup Mask operation is required before a masked store operation. When storing masked, the MASK response store register bits select the words or bits to receive data. Only words (bit mode) or bits (word mode) whose corresponding MASK register bit contains a 1 will receive data; all others will remain unchanged.

Instruction Format



(1) A masked Store must be preceded by a Setup MASK instruction.

(2) Logic may be performed on X and Y registers simultaneously with a store operation.

STORE RESPONSE
STORE REGISTERS TO ARRAY MEMORY

Sample Coding

- Store to Array Memory (Arrays selected by Array Select Register)
- Store Masked to Array Memory (Arrays selected by Array Select Register)

Instruction Hex Code (Bits 0-31)	Operation
1Ann0001	Store M (MASK) to array bit column nn
1Ann0002	Store Y to array bit column nn
1Enn0003	Store X to array word nn
1Bii0003	Store X to array bit column indirectly through field pointer (ii=indirect code)(3)
12nn0002	Store Y (masked) to array bit column nn (Must be preceded by a setup mask to same bit column to precondition the MASK; i, e., 0Ann0001$_{16}$)
17ii 0003	Store X (masked) to array word indirectly through field pointer (ii=indirect mode)[3] (The mask must be preconditioned to the same word address; i, e., 0Fii0001$_{16}$)

(3) ii= Indirect Code

i	Selected Field Pointer	Bump	i	Incr or Decr		Decr Field Length Counter		
0	FP1	–	0	Incr		–		
1	FP1	FP1	1	Incr		FL1		
2	FP1	FP1	2	Incr		FL2		
3	FP1	FP1	3	Incr		BL		
4	FP2	–	4	Decr		–		
5	FP2	FP1	5	Decr		FL1		
6	FP2	FP2	6	Decr		FL2		
7	FP2	FP1 & FP2	7	Decr		BL		
8	FP3	–	8	Incr	B	–	*	R
9	FP3	FP1	9	Incr	U	FL1		E
A	FP3	FP3	A	Incr	M	FL2	S	S
B	FP3	FP1 & FP3	B	Incr	P	BL	T	O
C	LP(a)	–	C	Decr		–	E	L
D	LP	Decr LP	D	Decr	F	FL1	P	V
E	LP	Incr LP	E	Decr	P	FL2		E
F*	LP	Load Link	F	Decr	E	BL		R

(a) LP = Link Pointer

MASK OPERATIONS (M-Response Store Register): There are three MASK operations: Load Mask, Setup Mask, and Generate Mask. The M response store register will be altered by these operations in all selected arrays. In arrays that are not selected, the M registers are affected unpredictably.

Load Mask Load Mask permits loading the M response store register from array memory, X and Y registers, and Common register. When the M register is loaded from array memory, the flip bit = 0 if the data being loaded is already in the flipped state (i.e., data was flipped when stored).

Instruction Format The Load Mask instruction format is the same as Load Response Store Instruction format (Page 2-21).

Sample Coding

	Instruction Hex code (Bits 0-31)	Operation
		Load M in all selected Arrays from:
• Load M Response Store Register in All Selected Arrays (Arrays selected by AS register)	08nn0005	Array bit column nn
	0Cnn0005	Array word nn
	09ii0005	Array bit column indirectly through field pointer (ii = indirect coding)(2)
	0Dii0005	Array word indirectly through field pointer (ii = indirect coding)(2)
	08000002	Y register
	08000003	X register
	08s00000	Common register into one 32 bit field (s)(1) other fields cleared
• Shift M In All Selected Arrays(3) (Arrays selected by AS register)	0A000019	Mirror and Shift M (FPE contains shift constant)
	08000009	Mirror M (No shift)
	0A000011	Shift M (FPE contains shift constant)

Reference notes (1) and (2) on following page.

(3) When shifting the M response store register, the shift constant must be specified in FPE, not in bits 8-15 of the instruction format. An alternate method of shifting M is to move M to one of the other response store registers and perform the shift in that register.

- Mask Operations In a Single Array. (Rightmost 5 bits of FP1 selects array)

Instruction Hex Code (Bits 0-31)	Operation
	Load M in one array from:
0BCy0000	Common register into one 32-bit field (y)[1]. Other fields are cleared.

(1) Field Select Table

s	Field (y)	Bits
0	0	0-31
2	1	32-63
4	2	64-95
6	3	96-127
8	4	128-159
A	5	160-191
C	6	192-223
E	7	224-255

(2) i i = Indirect Code

i	Selected Field Pointer	Bump	i	Incr or Decr		Decr Field Length Counter	
0	FP1	—	0	Incr		—	
1	FP1	FP1	1	Incr		FL1	
2	FP1	FP1	2	Incr		FL2	
3	FP1	FP1	3	Incr		BL	
4	FP2	—	4	Decr		—	
5	FP2	FP1	5	Decr		FL1	
6	FP2	FP2	6	Decr		FL2	
7	FP2	FP1 & FP2	7	Decr		BL	
8	FP3	—	8	Incr	BUMP FPPE	—	* STEP RESOLVER
9	FP3	FP1	9	Incr		FL1	
A	FP3	FP3	A	Incr		FL2	
B	FP3	FP1 & FP3	B	Incr		BL	
C	LP[a]	—	C	Decr		—	
D	LP	Decr LP	D	Decr		FL1	
E	LP	Incr LP	E	Decr		FL2	
F	LP	Load Link	F	Decr		BL	

[a] LP = Link Pointer

SETUP MASK

The Setup Mask instruction allows the MASK to be Setup to an array memory address before a Masked Store operation.

Instruction Format

The Setup Mask instruction format is the same as the Load Response Store instruction format (page 2-21).

Sample Coding

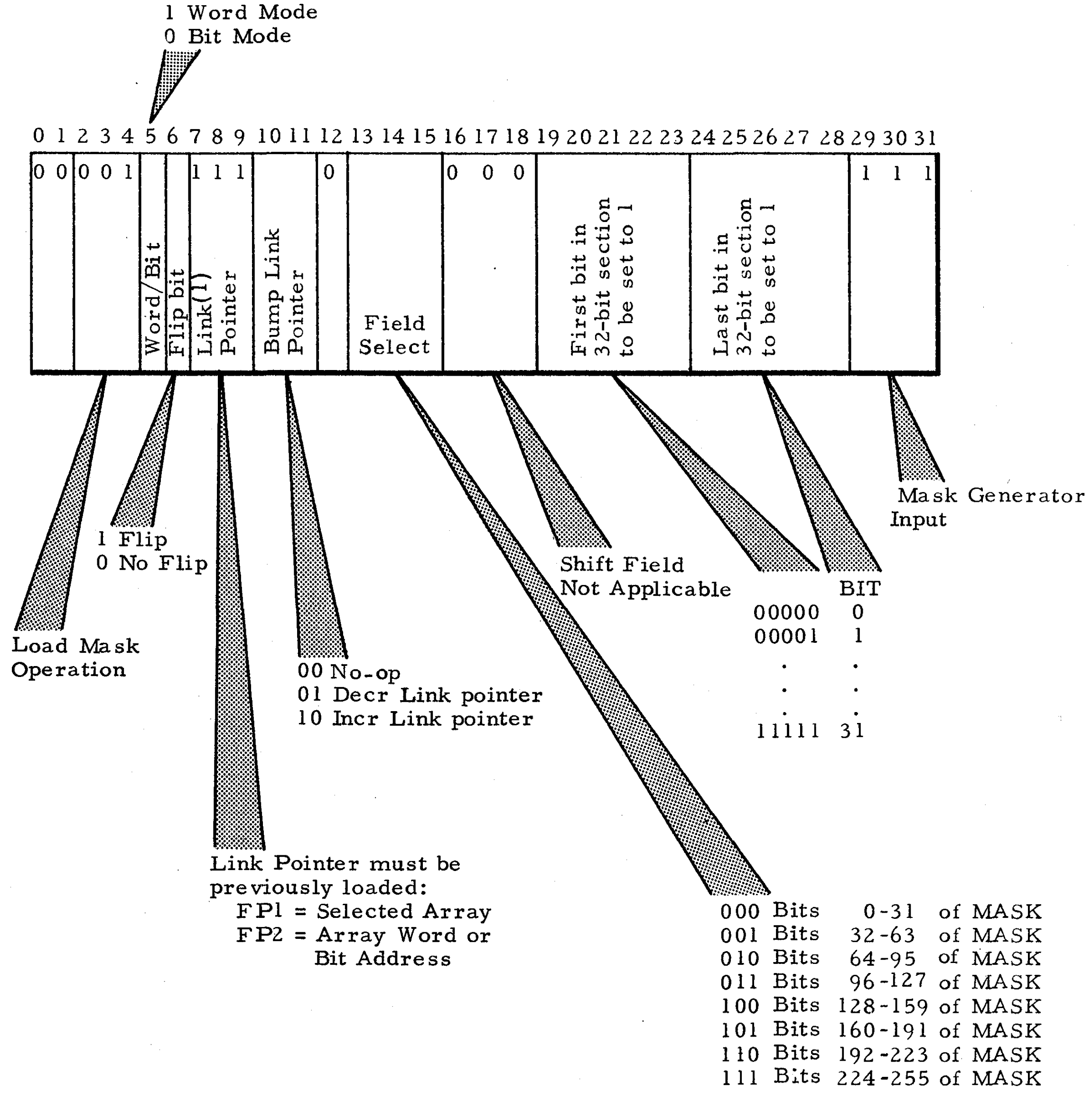
- Setup MASK In All Selected Arrays (Arrays selected by AS register)

Instruction Hex Code (Bits 0-31)	Operation
0Ann0001	Setup M to bit address nn.
0Enn0002	Load M from Y and setup to word address nn.
0Ann0003	Load M from X and setup to bit address nn.

GENERATE MASK

The Generate Mask instruction loads a new MASK into the M response store register of the selected array. This instruction usually precedes a Store Common Register instruction.

Instruction Format



	BIT
00000	0
00001	1
.	.
.	.
.	.
11111	31

000	Bits	0-31	of MASK
001	Bits	32-63	of MASK
010	Bits	64-95	of MASK
011	Bits	96-127	of MASK
100	Bits	128-159	of MASK
101	Bits	160-191	of MASK
110	Bits	192-223	of MASK
111	Bits	224-255	of MASK

(1) Link Pointer is FPl & FP2 linked together

GENERATE MASK

Sample Coding

	Instruction Hex Code (Bits 0-31)	Operation
● Generate Mask In all Selected Arrays (Arrays selected by Array Select Register)	08s0bbee[2]	Load Mask from Mask Generator into one 32-bit field(s)[1] other fields are cleared. (Load Mask format)
● Generate Mask In One Array (Rightmost 5 bits of FP1 selects array)	0BCybbee[2]	Load Mask from Mask Generator into one 32-bit field (y)[1] other fields are cleared. (Generate Mask format, i. e., Link Pointer mode)

(1) Field Select Table

s	Field (y)	Bits
0	0	0-31
2	1	32-63
4	2	64-95
6	3	96-127
8	4	128-159
A	5	160-191
C	6	192-223
E	7	224-255

(2) Field Start and End Bit Positions for Mask Generate

Starting Bit Position (Leftmost)	bb (Hex Code)	Ending Bit Position (Rightmost)	ee (Hex Code)
0	00	0	07
1	01	1	0F
2	02	2	17
3	03	3	1F
4	04	4	27
5	05	5	2F
6	06	6	37
7	07	7	3F
8	08	8	47
9	09	9	4F
10	0A	10	57
11	0B	11	5F
12	0C	12	67
13	0D	13	6F
14	0E	14	77
15	0F	15	7F
16	10	16	87
17	11	17	8F
18	12	18	97
19	13	19	9F
20	14	20	A7
21	15	21	AF
22	16	22	B7
23	17	23	BF
24	18	24	C7
25	19	25	CF
26	1A	26	D7
27	1B	27	DF
28	1C	28	E7
29	1D	29	EF
30	1E	30	F7
31	1F	31	FF

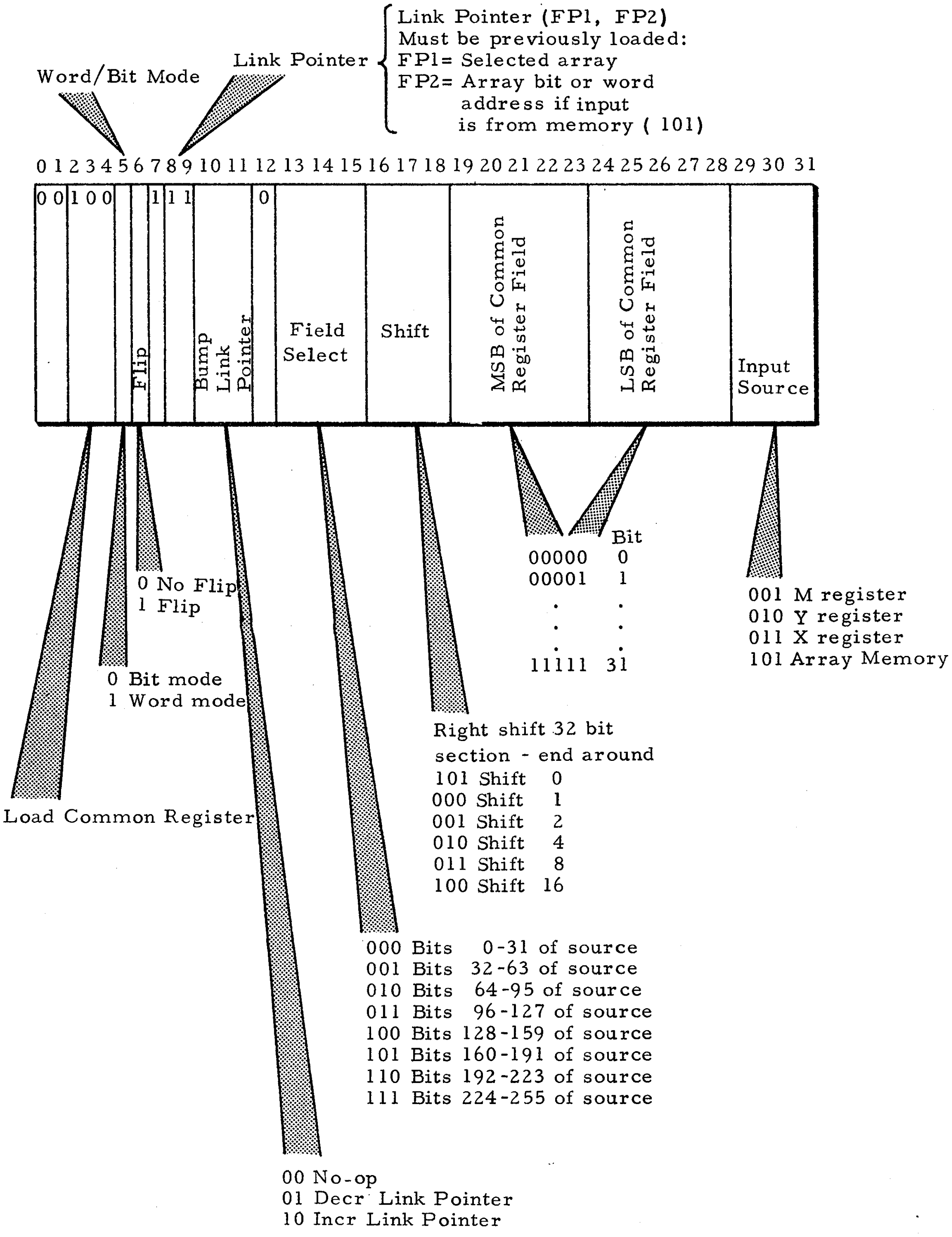
COMMON REGISTER INSTRUCTIONS

The Common Register group of instructions permits loading all or part of the Common Register from array memory or a response store register and storing all or part of the Common Register into an array bit column or word.

LOAD COMMON REGISTER

The Load Common Register instruction will load a selected part of the Common Register (other Common Register bits unchanged) from one array. Field Pointer 1 contains the address of the selected array.

Instruction Format



Link Pointer (FP1, FP2)
Must be previously loaded:
FP1= Selected array
FP2= Array bit or word address if input is from memory (101)

0 No Flip
1 Flip

0 Bit mode
1 Word mode

	Bit
00000	0
00001	1
.	.
.	.
.	.
11111	31

001	M register
010	Y register
011	X register
101	Array Memory

Right shift 32 bit section - end around

101	Shift 0
000	Shift 1
001	Shift 2
010	Shift 4
011	Shift 8
100	Shift 16

000	Bits 0-31 of source
001	Bits 32-63 of source
010	Bits 64-95 of source
011	Bits 96-127 of source
100	Bits 128-159 of source
101	Bits 160-191 of source
110	Bits 192-223 of source
111	Bits 224-255 of source

00	No-op
01	Decr Link Pointer
10	Incr Link Pointer

LOAD
COMMON
REGISTER

Sample
Coding

Instruction Hex Code (Bits 0-31)	Operation Load Common Register from:
21Cybbee(1), (2)	A 32-bit field of X register.
21Cybbee(1), (2)	A 32-bit field of Y register.
21Cybbee(1), (2)	A 32-bit field of M register.
21Cybbee(1), (2)	A 32-bit field of array bit column nn. FP1 and FP2 contain array address and bit column address.
25Cybbee(1), (2)	A 32-bit field of array word nn. FP1 and FP2 contain array address and word address.

(1) Field Select Table

y	Bits
0	0- 31
1	32- 63
2	64- 95
3	96-127
4	128-159
5	160-191
6	192-223
7	224-255

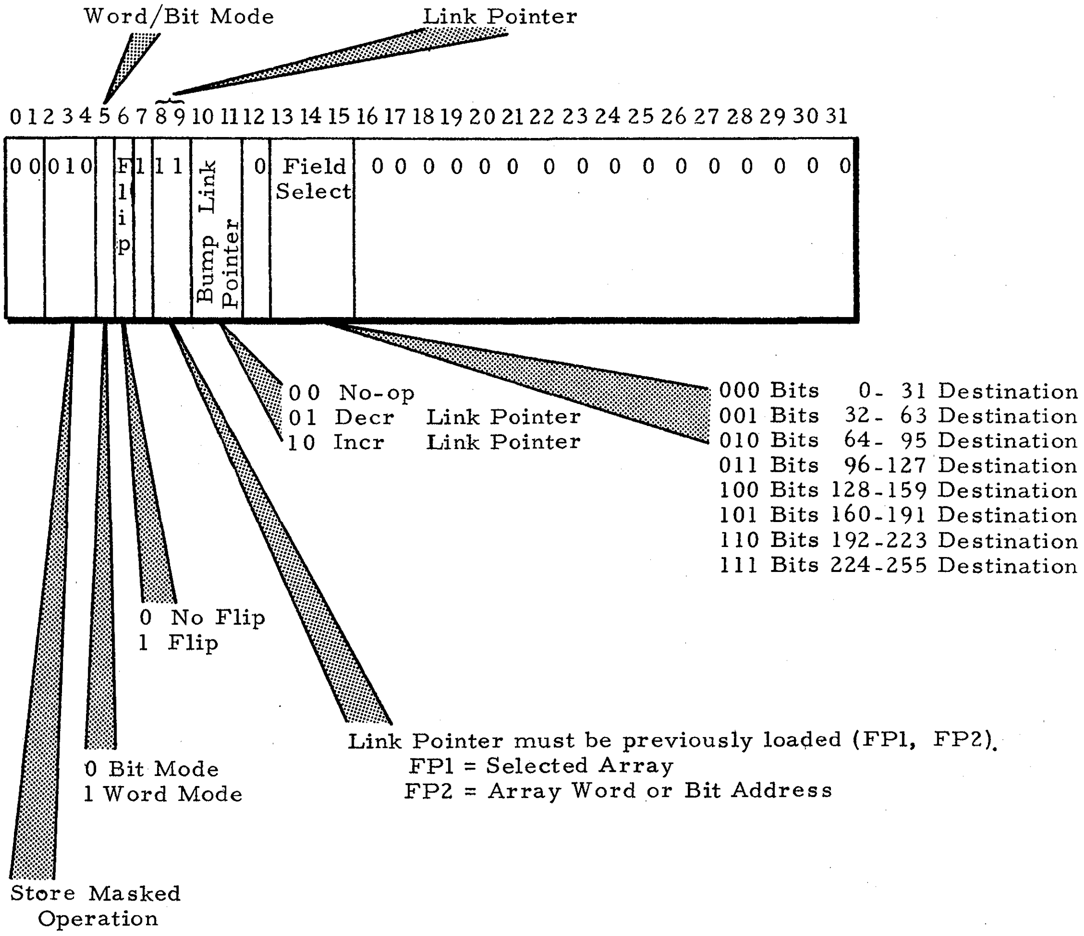
(2) Field start-and-end bit positions for load Common Register (assumes no shift)

Starting Bit Position (Leftmost)	bb (Hex Code)	Ending Bit Position (Rightmost)	ee (Hex Code) Input			
			Y	X	M	Array Memory
0	A0	0	02	03	01	05
1	A1	1	0A	0B	09	0D
2	A2	2	12	13	11	15
3	A3	3	1A	1B	19	1D
4	A4	4	22	23	21	25
5	A5	5	2A	2B	29	2D
6	A6	6	32	33	31	35
7	A7	7	3A	3B	39	3D
8	A8	8	42	43	41	45
9	A9	9	4A	4B	49	4D
10	AA	10	52	53	51	55
11	AB	11	5A	5B	59	5D
12	AC	12	62	63	61	65
13	AD	13	6A	6B	69	6D
14	AE	14	72	73	71	75
15	AF	15	7A	7B	79	7D
16	B0	16	82	83	81	85
17	B1	17	8A	8B	89	8D
18	B2	18	92	93	91	95
19	B3	19	9A	9B	99	9D
20	B4	20	A2	A3	A1	A5
21	B5	21	AA	AB	A9	AD
22	B6	22	B2	B3	B1	B5
23	B7	23	BA	BB	B9	BD
24	B8	24	C2	C3	C1	C5
25	B9	25	CA	CB	C9	CD
26	BA	26	D2	D3	D1	D5
27	BB	27	DA	DB	D9	DD
28	BC	28	E2	E3	E1	E5
29	BD	29	EA	EB	E9	ED
30	BE	30	F2	F3	F1	F5
31	BF	31	FA	FB	F9	FD

STORE COMMON REGISTER TO ARRAY MEMORY

The Store Common Register to Array Memory instruction permits storing all or part of the Common register to an array bit column or word. This * instruction must be preceded by a Generate Mask instruction to the same array and same bit column or word address. The Generate Mask instruction determines what portion of the Common register is to be stored. The field select must be the same in this instruction as in the Generate Mask instruction.

Instruction Format



Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
13Cy0000(1)	Store Common Register to 32-bit field of array bit column. (FP1 and FP2 contain array address and bit column address.)
17Cy0000(1)	Store Common Register to 32-bit field of array word (FP1 and FP2 contain array address and word address.)

(1) Field Select Table

Y	Bits
0	0-31
1	32-63
2	64-95
3	96-127
4	128-159
5	160-191
6	192-223
7	224-225

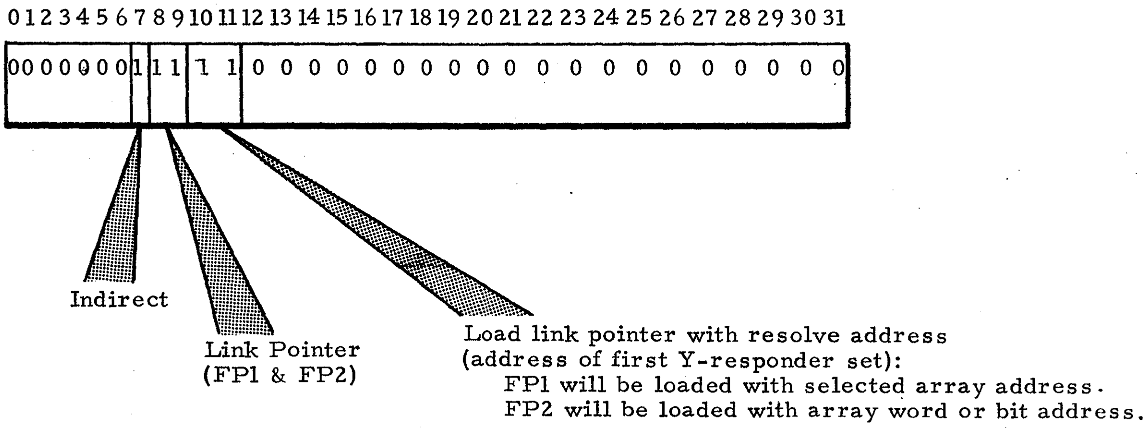
RESOLVE INSTRUCTIONS

The Resolve instructions allow the first responder (first Y register bit set to 1 in any selected array) to be located and manipulated. This is normally used in a search operation. The first data field that satisfies the search criteria will have its corresponding Y register bit set to 1.

FIND FIRST RESPONDER

The Find First Responder instruction finds the first selected array that contains at least one Y register bit that is set to a one and puts the array address in FP1. It also finds the first Y register bit in this array that is set to one and puts the address of that array bit column or word in FP2.

Instruction Format



Load link pointer with resolve address (address of first Y-responder set):
FP1 will be loaded with selected array address.
FP2 will be loaded with array word or bit address.

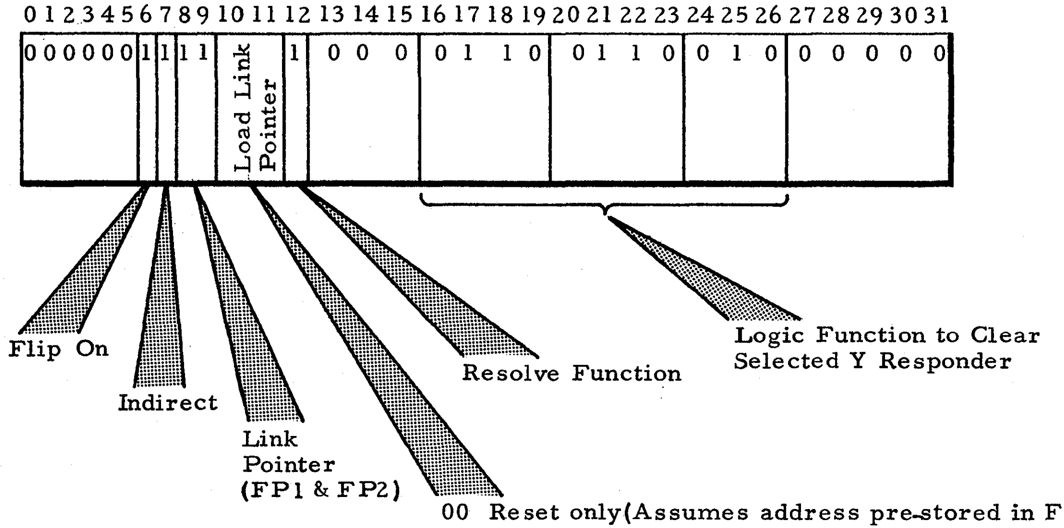
Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
01F00000	Find first responder.

RESET FIRST RESPONDER

The Reset First Responder instruction can find and reset the first responder to a search (first Y register bit set to 1 in selected arrays and words or bit columns) or reset only. This instruction may be used to reset any Y register bit if the address has been previously stored in FP1 and FP2.

Instruction Format



00 Reset only(Assumes address pre-stored in FP1 &FP2)
11 Load link pointer with resolve address and reset:
FP1 will be loaded with selected array address
FP2 will be loaded with array word or bit address

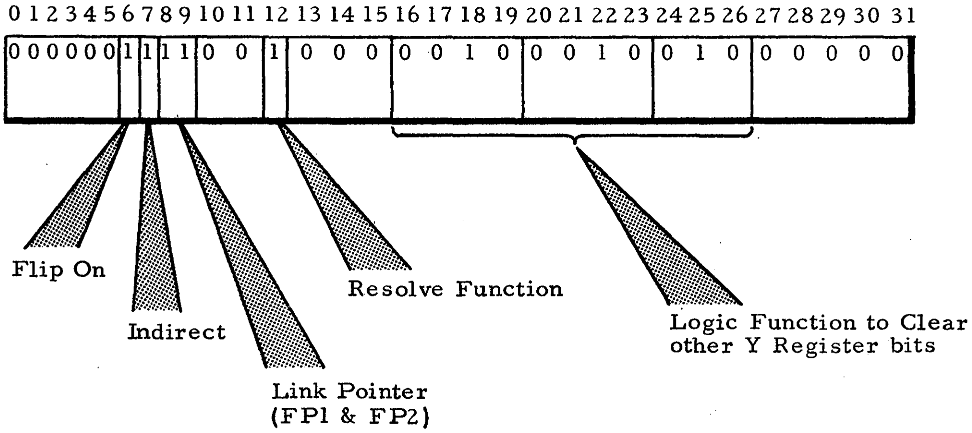
Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
03C86640	Reset first responder.
03F86640	Find and reset first responder.

RESET OTHER RESPONDERS

The Reset Other Responders instruction leaves the first responder set and resets all other Y register bits in the same array. It must be * preceded by a Find First Responder instruction (or another instruction which pre-loads the Y address in FP1 and FP2), since this instruction requires the address to be preloaded in the link pointer.

Instruction Format



0 1 2 3 4 5	6	7	8 9	10 11	12	13 14 15	16 17 18 19	20 21 22 23	24 25 26	27 28 29 30 31
0 0 0 0 0 0	1	1	1 1	0 0	1	0 0 0	0 0 1 0	0 0 1 0	0 1 0	0 0 0 0 0

Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
03C82240	Reset other than first responder (in same array only).

EXTERNAL FUNCTION INSTRUCTION FROM AP CONTROL

The External Function Instruction from AP Control issues an external function from AP control to the EXF logic. The EXF logic returns a sense bit. If the sense bit is one, the next sequential instruction will be skipped; if the sense bit is zero, the next sequential instruction will be executed. Section III of this Chapter discusses External Function operations in detail.

Instruction Format

0 1 2 3 4 5 6 7	8 9 10 11 12	13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
0 0 1 1 1 0 0 0	▨	EXTERNAL FUNCTION CODE

GENERAL

PROGRAM SEQUENCE

GET Address Register

Pager operation is initiated by an external function code (Load Get). When the pager is first put in the busy state, the GET address register points to the location of the first pager instruction. Normally, the pager fetches instructions from sequential memory locations, and the GET address register is incremented by 1 for each instruction (operation is similar to that of the Program Counter in AP control). The execution sequence is modified if a Move Data instruction is fetched. In this case, the memory locations immediately following the Move Data command contain the source block to be moved, and the next pager instruction immediately follows this source data block.

The execution sequence is also modified when a Load Get external function is issued by the pager or another STARAN S element. In this case, the GET address register is loaded with a new address from which the next instruction is fetched.

The execution sequence is halted when an external function to stop the pager is issued by the pager or another STARAN S element.

INSTRUCTION LENGTH

Each pager instruction is a 32-bit word. Bits 0 and 1 determine the instruction type.

INSTRUCTION TYPES

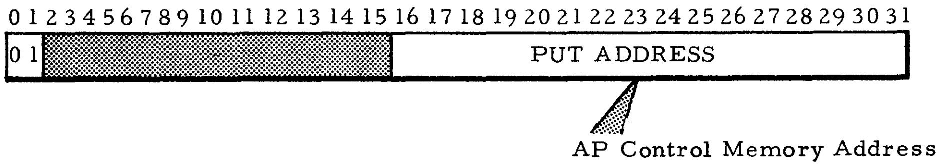
The pager instructions are: Load Put, Move Data, Load Put and Move Data, Issue EXF, Branch, Halt, Skip, Halt and Skip.

PROGRAM PAGER INSTRUCTIONS

LOAD PUT

The Load Put instruction loads the PUT address register with the address field (bits 16-31) of the instruction. The first word transferred by the next Move Data instruction will be put into this address.

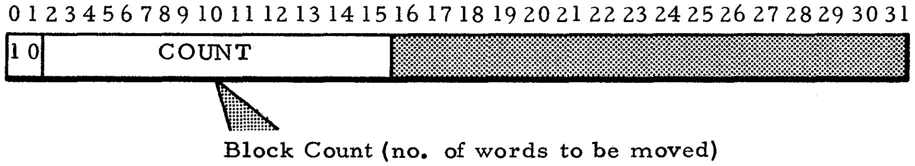
Instruction Format



MOVE DATA

The Move Data instruction causes memory words immediately following the instruction to be loaded into the page memory. The block length is contained in the count field of the command. The first instruction of the block is loaded into the location pointed to by the PUT Address register. Succeeding instructions are loaded in sequential page memory locations. At the end of the Move Data execution, the PUT Address register points to the location following the last word of the destination block and the GET Address register points to the location following the last word of the source block where the next instruction is fetched.

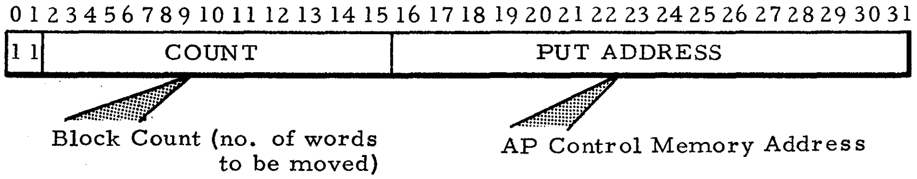
Instruction Format



LOAD PUT AND MOVE DATA

The Load PUT and Move Data instruction loads the PUT Address register and then causes a block of data to be moved. It performs the same function as a Load PUT instruction followed by a Move Data instruction.

Instruction Format



Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
4000nnnn (nnnn = page address)	Load PUT Address register.
8kkk0000(1) (kkk = no. of words to be moved)	Move block from bulk core to page memory.
Ckkknnnn(1) (kkk = no. of words to be moved)	Load PUT Address register and move block from bulk core to page memory.

(1) The count may extend into the leftmost hex digit, because the count field is actually 14 bits in length.

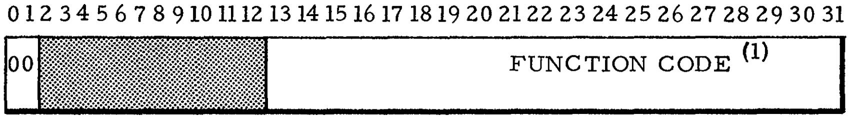
PAGER EXTERNAL FUNCTION INSTRUCTIONS

When the pager issues an external function, it competes with other STARAN S elements that are issuing external functions. The EXF logic treats one EXF at a time. If the EXF logic accepts one EXF from another element (which affects the pager) before the pager EXF is accepted, the pager EXF will be ignored.

Pager Issue EXF Instruction

This instruction issues a function code to the EXF logic. When the EXF logic accepts the code, it returns a sense bit dependent on the particular code and status of the STARAN S element. If the sense bit is 0, the pager fetches its next instruction from the next sequential memory location. If the sense bit is 1, a skip of one instruction is performed and the GET address register is incremented by 2 instead of 1 at the end of instruction execution.

Instruction Format



(1) See Appendix A, page A-7, for hex codes of function codes.

Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
000 (19-bit function code)(1)	Issue external function from pager.

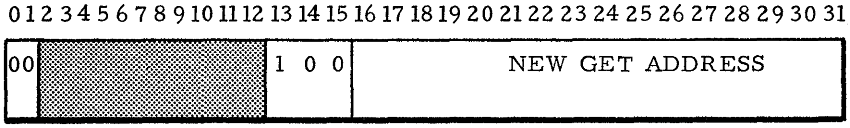
Note

Certain External functions affect the pager itself. The instructions which issue these functions are described in the following items.

PAGER BRANCH INSTRUCTION

The Pager Branch instruction causes the next pager instructions to be fetched from the address specified in bits 16 to 31 of the instruction format. The PUT address register is unchanged.

Instruction Format



Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
0004nnnn (where nnnn is hexadecimal code for new address)	Branch to new GET address.

PAGER HALT & SKIP INSTRUCTION

The Pager Halt and Skip instruction halts the pager operation with the GET address register pointing to the second location after the Halt and Skip instruction.

Instruction Format

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

00	▨	0 0 0 1	▨	0 0 0 1

Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
00008001	Halt Pager and Skip Next Sequential instruction.

PAGER SKIP INSTRUCTION

The Pager Skip instruction causes the next sequential instruction to be skipped.

Instruction Format

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

00	▨	0 0 0 1	▨	0 1 0 1

Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
00008005	Skip next sequential instruction.

PAGER HALT INSTRUCTION

The Pager Halt instruction halts the pager operation with the GET address register pointing to the location following the Halt instruction.

Instruction Format

0 1	2 3 4 5 6 7 8 9 10 11 12	13 14 15 16	17 18 19 20 21 22 23 24 25 26 27	28 29 30 31
00	▨	0 0 0 1	▨	0 0 0 0

Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
00008000	Halt pager.

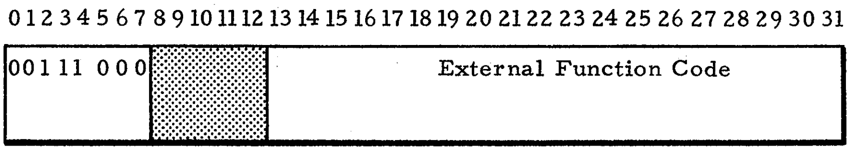
GENERAL

The External Function instruction issues an external function (EXF) code to EXF logic. If the EXF logic returns a sense bit of 1, the next sequential instruction is skipped; if the sense bit is 0, the next sequential instruction is executed. Each function code is 19 bits long and indicates what element of Staran S is to be interrogated and/or controlled. Function codes may be transmitted to EXF logic by AP control, the program pager, sequential control or the host computer (if the EXF channel is implemented). One function code at a time is accepted by EXF logic. A function code can both interrogate and control an element in one operation.

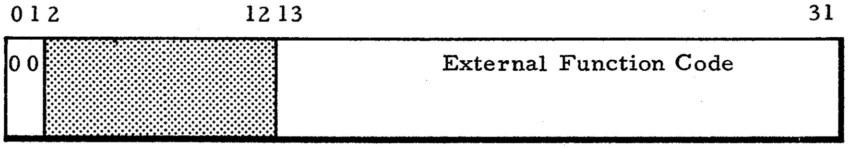
FUNCTION CODE CLASSES

The classes of function codes are Page Port Switches, Interlocks, Program Pager, Error Control, AP control interrupts, Sequential Control interrupts, and AP activity.

Instruction Format From Associative Processor Control



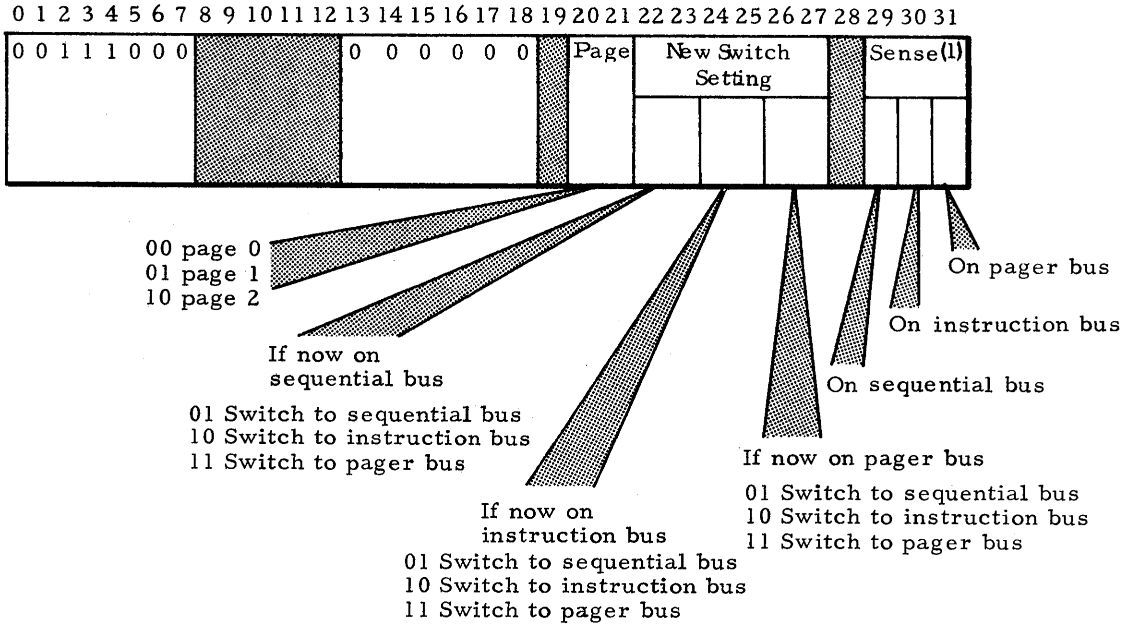
Instruction Format From Program Pager



PAGER PORT SWITCH INSTRUCTION

Each page memory has a port switch which can connect the memory to the AP control instruction bus, the pager bus, or the sequential control bus. This function code permits interrogation and control of these switches.

Instruction Format



(1) Only one of bits 29-31 is returned as sense bit.

Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
380002A0 380006A0 38000AA0	Switch Page 0 to Instruction Bus Switch Page 1 to Instruction Bus Switch Page 2 to Instruction Bus
380003F0 380007F0 38000BF0	Switch Page 0 to Pager Bus Switch Page 1 to Pager Bus Switch Page 2 to Pager Bus
38000150 38000550 38000950	Switch Page 0 to Sequential Bus Switch Page 1 to Sequential Bus Switch Page 2 to Sequential Bus
380001B2 380005B2 380009B2	Sense if Page 0 on Instruction Bus Sense if Page 1 on Instruction Bus Sense if Page 2 on Instruction Bus
380001B1 380005B1 380009B1	Sense if Page 0 on Pager Bus Sense if Page 1 on Pager Bus Sense if Page 2 on Pager Bus
380001B4 380005B4 380009B4	Sense if Page 0 on Sequential Bus Sense if Page 1 on Sequential Bus Sense if Page 2 on Sequential Bus

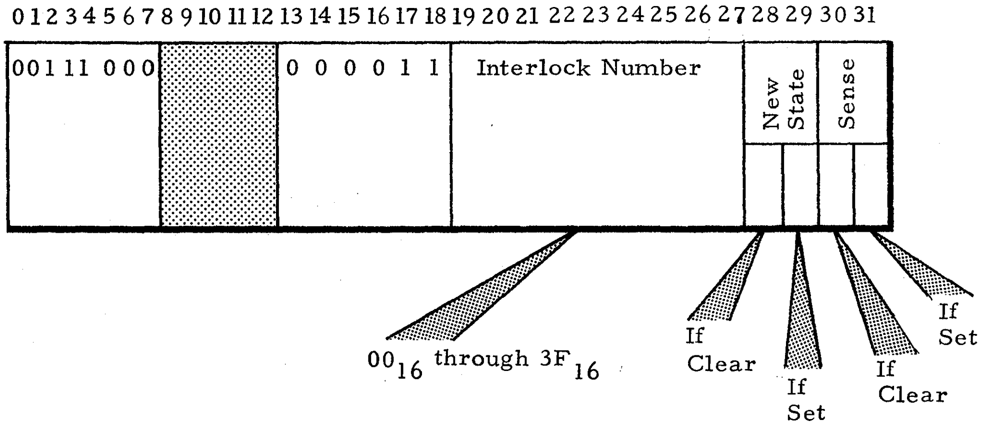
INTERLOCKS

The EXF logic contains 64 stored bits called interlocks. These bits have no predetermined meaning. Software can assign a meaning to an interlock and use it for any purpose. Function codes allow the current state of an interlock to be sensed and a new state entered in one operation. The current state of the selected interlock is sensed, and depending on the state, either bit 30 or bit 31 is returned as a sense bit. The current state also selects bit 28 or bit 29 as the new state of the interlock. If the selected bit for the new state is 0, the interlock is cleared; if the selected bit for the new state is 1, the interlock is set.

Interlock Switches and Lights

Sixteen interlocks (00 through $0F_{16}$) can be controlled and sensed manually by panel switches and lights. The other 48 interlocks (10_{16} through $3F_{16}$) can be sensed and controlled only by function codes.

Instruction Format



Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
38006(n)C	Set Interlock n
38006(n)0	Clear Interlock n
38006(n)5	Sense if Interlock n set
38006(n)6	Sense if Interlock n clear
38006(n)1	Sense if Interlock n set, then clear it
38006(n)D	Sense if Interlock n set, then set it
38006(n)2	Sense if Interlock n clear, then clear it
38006(n)E	Sense if Interlock n clear, then set it

00≤n≤3F

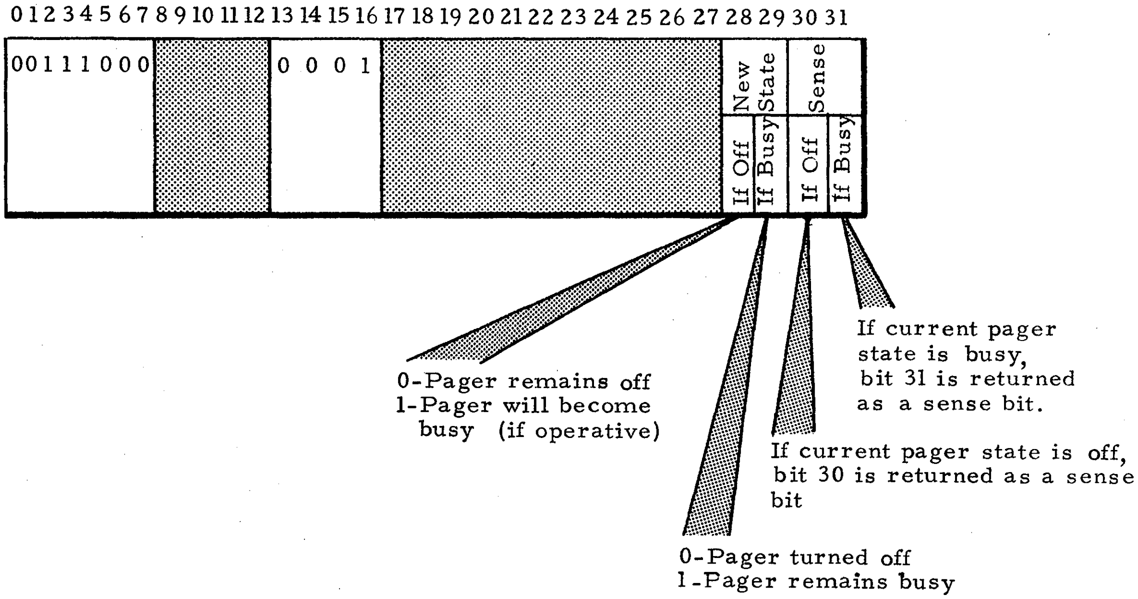
PROGRAM PAGER

The program pager has two states, off and busy. A function code allows the state to be sensed and changed. Another function code affecting the pager loads its GET address register with a new address.

PAGER STATE INSTRUCTION

The Pager State instruction allows the state of the pager to be sensed and changed. If the current pager state is off, bit 30 is returned as a sense bit and bit 28 governs the new pager state; if the pager is busy, bit 31 is returned and bit 29 governs the new pager state. If the selected bit for the new state is 1, the pager will become busy; if the selected bit is 0, the pager will be turned off.

Instruction Format



0-Pager remains off
1-Pager will become busy (if operative)

0-Pager turned off
1-Pager remains busy

If current pager state is off, bit 30 is returned as a sense bit

If current pager state is busy, bit 31 is returned as a sense bit.

Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
38008000	Stop pager.
3800800C	Start pager where previously stopped (if operative).
38008005	Sense if pager is busy.
38008006	Sense if pager off.

PAGER LOAD GET INSTRUCTION

The Pager Load GET instruction loads the pager GET address register with a new address. The sense bit returned for this function code will always be zero. If the pager is in the midst of moving data or loading the PUT address and moving data, the moving will be interrupted between word transfers. If the pager was issuing an EXF, the EXF will not be processed if EXF logic has accepted the Load GET EXF first.

Instruction Format

0 1 2 3 4 5 6 7	8 9 10 11 12	13 14 15	16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
0 0 1 1 1 0 0 0	▨	1 0 0	GET Address

If pager is inoperative, its GET address register is not disturbed and pager remains inoperative.

If pager is operative, its GET address register is loaded with bits 16-31 of the instruction and the count register is cleared; the pager becomes busy and fetches its first instruction from the new get address.

Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
3804nnnn (nnnn-GET address)	Load pager GET address register.

ASSOCIATIVE PROCESSOR CONTROL INTERRUPTS

The AP Control Interrupts instruction is used to sense, set, and reset the state of the 15 interrupts to AP control. AP control interrupts are given hex numbers 01 (lowest priority) to 0F (highest priority).

Interrupt Mask

Bits 28 through 31 of the program status word (PSW) in AP control contain an interrupt mask (IMASK).

Interrupt Conditions

AP control accepts interrupt n if the following conditions exist:

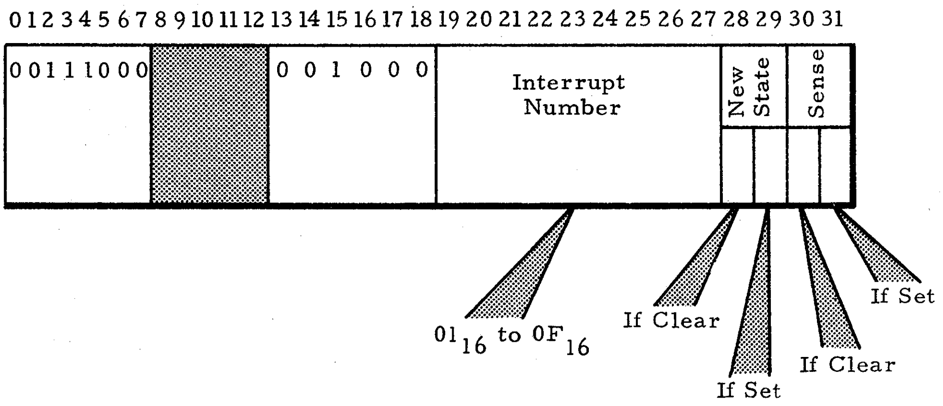
1) AP control is active on at an interruptable point.
2) The current IMASK is less than n.
3) No interrupt of higher priority is set.
4) Interrupt n is set.

Interrupt Handling

When interrupt n is accepted, AP control fetches the next instruction from hex address 8000 + n without altering the program counter. Normally the instruction at the above address is a swap PSW, which saves the old PSW and loads a new PSW, causing control to be transferred to an interrupt handling routine. The IMASK of the new PSW should be n or greater to prevent AP control from accepting the n interrupt again until the interrupt handling routine is complete.

The current state of the interrupt is used to select bit 30 or 31 to be returned as a sense bit and to select either bit 28 or 29 for the new state of the interrupt. If the selected bit for the new state is 1, the interrupt is left in the set state; if it is 0, the interrupt is left in the clear state.

Instruction Format



Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
38010(n)C	Set AP Interrupt n.
38010(n)0	Clear AP Interrupt n.
38010(n)5	Sense if AP Interrupt n is Set.
38010(n)6	Sense if AP Interrupt n is Clear.

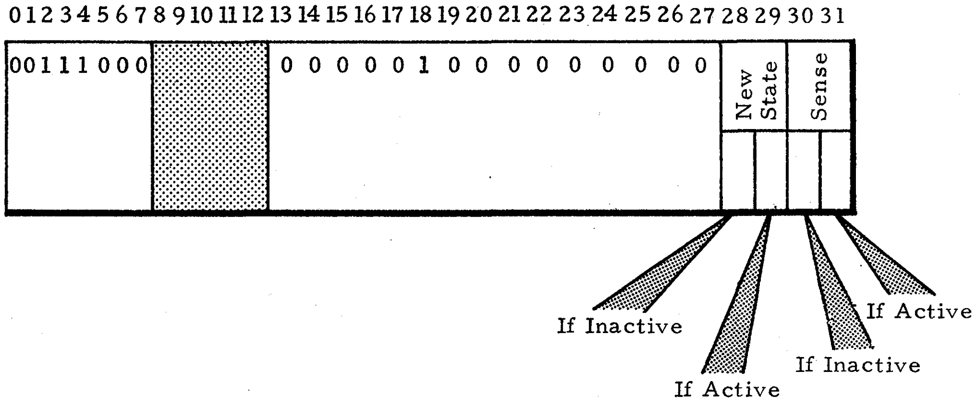
01 ≤ n ≤ 0F

ASSOCIATIVE PROCESSOR CONTROL ACTIVITY

The AP Control Activity instruction permits the sensing and controlling of AP activity. The AP control has two states: active and inactive. In the active state AP control fetches instructions from AP control memory and exercises the associative arrays. When switched from the active to the inactive state, all AP control registers remain as they were after executing the last instruction. When going from the inactive to the active state, AP control fetches its first instruction from location 8000_{16} without altering the program counter. Thus, if 8000_{16} contains a No-op instruction, AP control would continue with its previous program. If 8000_{16} contains a branch, a new program sequence would be entered. If 8000_{16} contains a swap PSW, a new program sequence would be entered while saving the old Program Counter status to allow future re-entry into the old program sequence.

The current state of AP activity is sensed and bit 30 or 31 is returned as a sense bit; bit 28 or 29 is selected as the new state. If the selected bit for the new state is 0, the AP will become inactive; if the selected bit is 1, the AP will become active.

Instruction Format



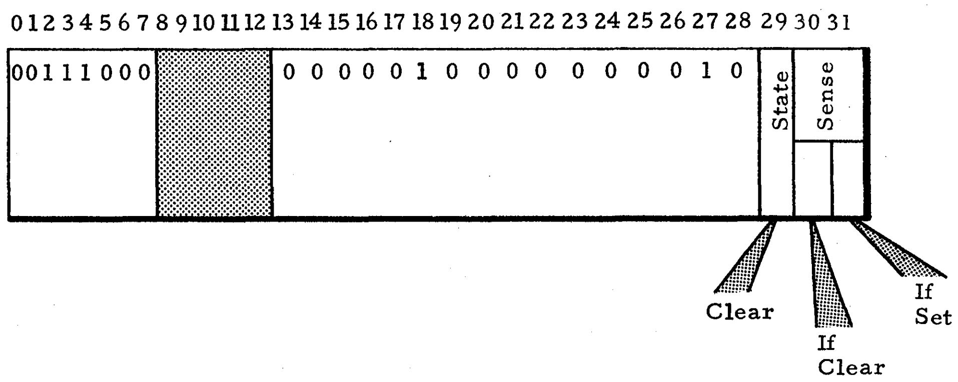
Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
38002000	Halt AP (AP becomes inactive).
3800200C	Start AP at location 8000_{16} (AP becomes active).
38002005	Sense if AP is active.
38002006	Sense if AP is inactive.

ASSOCIATIVE PROCESSOR CONTROL LOOP INDICATOR

The AP control loop indicator is set and remains set until all repetitions of a loop are completed. The indicator is neither disturbed by changes in activity nor by interrupts. This function code allows the loop indicator to be sensed and/or cleared. Clearing the set loop indicator causes AP control to terminate the loop,even if all repetitions have not been completed. The current state of the loop indicator is sensed and bit 30 or 31 is returned as a sense bit; if bit 29 is set to 1, the indicator is cleared; otherwise, it remains unchanged.

Instruction Format



Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
38002011	Sense if AP is in loop mode.
38002012	Sense if AP is not in loop mode.
38002014	Take AP out of loop mode.

ERROR CONTROL

The Error Control instruction will sense, set, and/or reset error indicators. Error detectors are included in various elements of STARAN to sense hardware faults and program errors. Each error detector sets an error indicator when an error is detected. Each error indicator is given a number by which it may be sensed, set, and/or reset. If any error indicator is set, an interrupt to sequential control is generated. Certain errors will make the pager inoperative and the AP control inactive.

Error Table

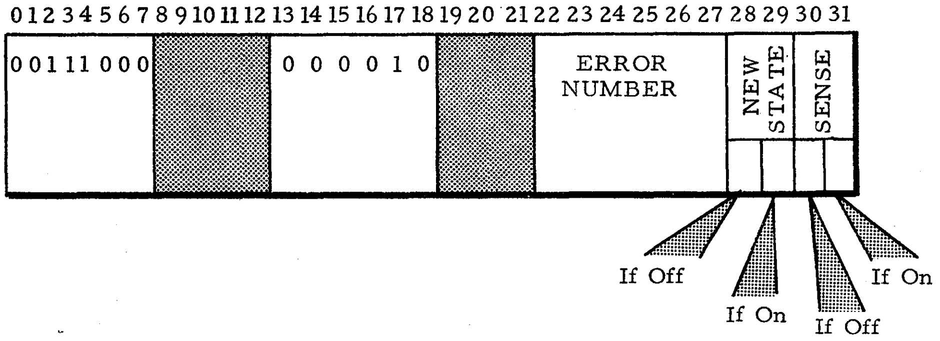
NUMBER (Hexadecimal)	ERROR DESCRIPTION	ACTION WHEN SET[1]
00	Illegal instruction in AP control	AP control inactive
01	AP control hung up in a loop	AP control inactive
02	Buffered I/O bus hung up	-
03	AP control data bus hung up	AP control inactive
04	AP control instruction bus hung up	AP control inactive
05	Pager GET bus hung up	Pager inoperative
06	Pager PUT bus hung up	Pager inoperative
08	Parity error on AP control data bus	AP control inactive
09	Parity error on AP control instruction in bus	AP control inactive
0B	Parity error on Sequential Control bus	-

[1] All errors generate an interrupt to sequential control

The current state of the selected error is sensed, and depending on the state, either bit 30 or bit 31 is returned as a sense bit.

The current state also selects either bit 28 or 29 and loads the selected bit into the error indicator as a new state. If the selected new state bit is 1, the error indicator will be set; if the selected bit is 0, the error indicator will be cleared.

Instruction Format



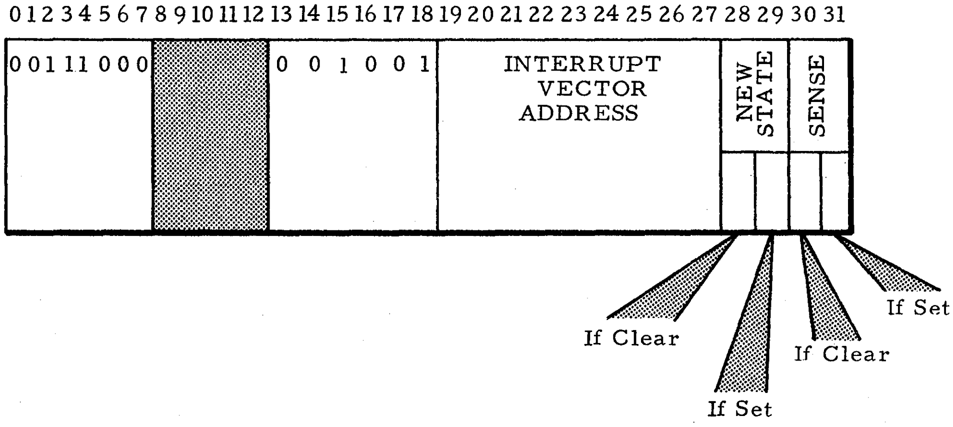
Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
38004011	Test to see if AP control is hung up in a loop; if yes,reset error indicator 01.

SEQUENTIAL CONTROL INTERRUPT

The Sequential Control Interrupt instruction can sense, set, and reset the state of eight interrupts to sequential control. The current state of the selected interrupt is sensed, and depending on the state, either bit 30 or bit 31 is returned as a sense bit. The current state also selects bit 28 or bit 29 and uses the selected bit as the new state of the interrupt. If the selected bit for the new state is 1, the interrupt is left in the set state; if the selected bit is 0, the interrupt is left in the clear state.

Instruction Format



Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
38012(n)C	Set Sequential Interrupt n (C0≤n≤DC).
38012(n)0	Clear Sequential Interrupt n
38012(n)5	Sense if Sequential Interrupt n is set.
38012(n)6	Sense if Sequential Interrupt n is clear.

n	Octal Vector Address	Priority	Remarks
C0	300	4	
C4	304	4	
C8	310	5	
CC	314	5	
D0	320	6	
D4	324	6	
D8	330	7	Panel Interrupt
DC	334	7	Errors

RESETS AND CLEARS

The Reset and Clear function codes are used for resetting and clearing various STARAN S logic and status bits. They are used for clearing hangup conditions and initial clearing when STARAN S power is turned on. Because certain clears will clear out the logic issuing the EXF, these function codes are not usually issued by AP control or by the program pager.

Instruction Format

0 1 2 3 4 5 6 7	8 9 10 11 12	13 14 15 16 17 18	19 20 21	22 23 24 25 26 27	28 29 30 31
0 0 1 1 1 0 0 0	▨	0 0 0 0 0 1	▨	Clear & Reset Number	▨

Sample Coding

Instruction Hex Code (Bits 0-31)	Operation
38002(n)0	Reset n (02≤n≤2E)
38002020	Clear EXF Resolve
38002030	Clear Errors, Seq. Interrupts
38002040	Clear AP Interrupts
38002050	Clear DMA Logic
38002060	Clear Core Logic
38002070	Clear Next Instruction Logic
38002080	Clear Branch Logic
38002090	Clear Register Logic
380020A0	Clear Pager Logic
380020B0	Clear HSDB Logic
380020D0	Clear Associative Instruction Logic
380020E0	Clear Page 0 Logic
380020F0	Clear Page 1 Logic
28002100	Clear Page 2 Logic
38002300	Master Clear

SPARE EXTERNAL FUNCTIONS

Certain function codes are left as spare functions to be used by the custom I/O cabinet. They may be used for generating interrupts to a host computer or setting up I/O channels.

CHAPTER III

INPUT/OUTPUT VARIATIONS

GENERAL

A wide variety of Input/Output options are available with STARAN S. Input/output (I/O) control in STARAN S is located in an I/O cabinet, which can be customized to fit the needs of a particular installation. Since the custom I/O cabinet will differ from installation to installation, this chapter discusses only what I/O options are available. The I/O controls for particular installations are discussed in a separate document.

One I/O option available with STARAN S is integration with another computer system (called the host computer), either with direct memory access (DMA), buffered I/O, external functions or, a combination of these. Another option available is interfacing with a wide variety of data gathering, data transmitting, or data storing devices with either the standard STARAN S buffered I/O channel or the very high bandwidth parallel I/O channel. Error checking can also be included in the I/O cabinet to check for errors in data transmission. The different types of I/O options available are discussed in the following subsections.

DIRECT MEMORY ACCESS CHANNEL

Direct access to a host computer memory allows that memory to act as part of of the AP control memory. Items in the memory are equally accessible by a host computer and by STARAN S, thus reducing the need for buffered I/O transfers between the host computer and STARAN S.

Implementation of Direct Memory Access (DMA) from STARAN S to the host computer depends on the characteristics of the host computer. A wide variety of computers can be accommodated. There are, however, some that do not support DMA without extensive modification. Cycle times with DMA will be somewhat longer than the normal cycle times of the host-computer memory because of extra cable lengths and logic gating. If the host-computer word length is less than 32 bits, the host-computer word is padded with zeros to provide 32 bits when it is read by STARAN S; the padded bits are truncated when the word is written by STARAN S. If the host-computer word length is more than 32 bits, the host-computer word is truncated to 32 bits when it is read by STARAN S; the truncated bits are padded with zeros when the word is written by STARAN S.

Address translation may be required on the DMA interface to match the AP control addresses with the host-computer addresses. This translation can be accommodated in the custom I/O cabinet.

The DMA block of addresses may be put to other uses besides access to a host-computer memory. Possible uses include access to an external memory, which may or may not be accessible by other devices, and access to special I/O devices.

BUFFERED INPUT/ OUTPUT CHANNEL

Buffered Input/Output (BI/O) is available for tying several different types of peripherals into the AP control memory of STARAN S. In addition, BI/Ocan be used to transfer blocks of data and/or programs between the AP control memory and a host-computer memory. In general, a DMA interface to a host-computer memory is preferable to the BI/Ointerface. BI/O cycle times are usually longer than DMA cycle times in the host computer. Also, BI/O forces the programmer to put the data in blocks, while DMA can operate with scattered data.

Of the several data buses accessing the high-speed data buffer (HSDB), the bulk core memory (BCM), and the external memory accessed via DMA channel, priority resolver logic gives the BI/O port highest priority.

The basic width of the BI/O interface is 32 bits plus a parity bit. The custom I/O cabinet can include buffers, which will allow a wide variety of repacking to take place so that I/O channels of any width can be assembled.

Normally, the External Function channel is used to set up and initiate buffered I/O transfers.

EXTERNAL FUNCTION CHANNEL

GENERAL

The external function (EXF) channel is used extensively for direct communication with the control logic of peripherals connected to the custom I/O cabinet. Up to 19 bits of function code can be received simultaneously at the custom I/O from a large number of peripheral units. The custom I/O handles each of the function codes one by one on a priority basis until all have been processed.

HOST-COMPUTER EXTERNAL FUNCTION INTERFACE

When STARAN S is integrated with a host computer, the external function interface is used to pass control information back and forth.

If the host computer can accept external interrupts, STARAN S can generate such interrupts by using spare external function codes. The custom I/O cabinet generates an interrupt to the host whenever these external function codes are decoded. A large number of separate interrupts can be accommodated if necessary.

STARAN S can accept interrupts from a host computer if the host can generate signals on discrete output lines. The custom I/O cabinet will accept such signals and generate certain function codes to the external function logic in the main frame, which can cause interrupts or control other elements of STARAN S.

If the host computer has an external function output (direct output, etc.) capability which can output a data item at least 19 bits wide, the custom I/O cabinet can accept the item and send it to the EXF logic in the main frame as a function code. This allows the host computer to generate any external function code and exercise complete control of STARAN. Not only can it generate interrupts, but it can also activate and deactivate elements of STARAN S, control interlock bits, initiate transfers, etc. If the host computer can accept one bit of data (for instance, into a condition code) while it is outputting the 19 bits, then the sense bit output of the EXF logic can be sent back to the host, allowing it to interrogate various elements of STARAN S.

BUFFERED INPUT/ OUTPUT EXTERNAL FUNCTIONS

Buffered I/O can be easily initiated by means of EXF codes. Word counts, starting addresses, and other information can be packed into the specific format necessary to communicate with a peripheral device. This data is sent along with the command that initiates a sequence of events.

PARALLEL INPUT/ OUTPUT EXTERNAL FUNCTIONS

Parallel I/O is initiated with EXF codes in much the same manner as BI/O. However, PI/O may need more data than can be packed in one or two EXF codes. Because of this, it may be desirable to use the EXF code to initiate an operation that transfers several data words over buffered I/O. These data words then allow a PI/O transfer to be executed.

PARALLEL INPUT/ OUTPUT CHANNEL

GENERAL

Each associative array can have up to 256 inputs and 256 outputs into the custom I/O cabinet. (A 32-array system could have 8,192 input and 8,192 output lines). These can be used for various purposes. For example, it could greatly speed up inter-array data communication through the shifting of array data, allow a very high bandwidth I/O device to communicate with STARAN S, or allow any device to talk directly with the associative arrays. Three of these applications are discussed in the following subsections.

INTER-ARRAY COMMUNICATION

The PI/O lines for each array may be interconnected in one or more ways to effect the fast transfer of data between arrays. The control of these lines will be set up by various external function codes sent to the I/O cabinet from STARAN S prior to the transfer. Without inter-array communications, data can still be transferred from one array to another by sending it through the common register.

HIGH BANDWIDTH I/O

The basic width of the PI/O is 256 n where n is equal to the number of associative arrays in the system. (n can have a maximum value of 3?) The custom I/O cabinet is capable of buffering and reformatting the data received from any peripheral device to match the width necessary to communicate with the STARAN S associative array. In order to synchronize read and write operations with an external device operating through PI/O, certain AP control instructions wait for sync pulses from the external device. As in inter-aray communication, an external function code sent to the custom I/O will set up and initiate the PI/O operation.

SEQUENTIAL CONTROL PERIPHERALS

STARAN S has as standard peripherals a keyboard/printer and a perforated tape reader/punch. Other optional peripheral s are noted below.

PAPER TAPE READER/PUNCH

The perforated tape reader and punch units handle 8-level tape with a reader speed of 300 characters per second and a punch speed of 50 characters per second.

The keyboard/printer and tape reader/punch peripherals interface directly with sequential control. Communication between the remainder of STARAN S and the peripherals is then handled by sequential control.

KEYBOARD/PRINTER

The keyboard/printer provides a communication link between the STARAN S system and an operator. During operation STARAN S fault and error messages are output on the printer as detected by on-line diagnostic software in the sequential control. The keyboard provides a means for the operator to input command statements to change operation modes of STARAN S.

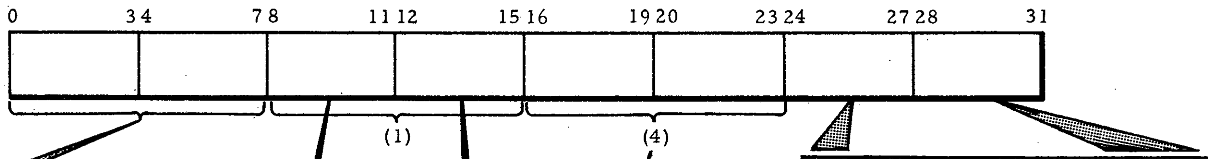
The keyboard/printer and reader/punch also provide a stand-alone capability to:

1) Load STARAN S system programs.
2) Run and control STARAN S maintenance and diagnostic programs.
3) Assemble, edit, and debug STARAN S assembly language programs.

PERIPHERAL OPTIONS

Other sequential control peripherials such as magnetic tape, disc, card readers, line printers, and communications to remote devices are also available as options.

ASSOCIATIVE ARRAY INSTRUCTIONS



Hex Code	Function	Word (W) or Bit (B) Mode	Flip Mode (ON=◆)	Indirect Mode (ON=◆)
00	Load X or Y Registers	B		
01		B		◆
02		B	◆	
03		B	◆	◆
04		W		
05		W		◆
06		W	◆	
07		W	◆	◆
08	Load (2) M (MASK) Register	B		
09		B		◆
0A		B	◆	
0B		B	◆	◆
0C		W		
0D		W		◆
0E		W	◆	
0F		W	◆	◆
10	Masked Store to Array Memory	B		
11		B		◆
12		B	◆	
13		B	◆	◆
14		W		
15		W		◆
16		W	◆	
17		W	◆	◆
18	Store to Array Memory	B		
19		B		◆
1A		B	◆	
1B		B	◆	◆
1C		W		
1D		W		◆
1E		W	◆	
1F		W	◆	◆
20	Load (2) Common Register from Array Memory	B		
21		B		◆
22		B	◆	
23		B	◆	◆
24		W		
25		W		◆
26		W	◆	
27		W	◆	◆

ii Indirect Mode

Hex Code	Selected Field Pointer	Bump	Hex Code	Incr or Decr		Decr Field Length Counter	
0	FP1	—	0	Incr		—	
1	FP1	FP1	1	Incr		FL1	
2	FP1	FP1	2	Incr		FL2	
3	FP1	FP1	3	Incr		BL	
4	FP2	—	4	Decr		—	
5	FP2	FP1	5	Decr		FL1	
6	FP2	FP2	6	Decr		FL2	
7	FP2	FP1&FP2	7	Decr		BL	
8	FP3	—	8	Incr	BUMP FPE	—	STEP * RESOLVER
9	FP3	FP1	9	Incr		FL1	
A	FP3	FP3	A	Incr		FL2	
B	FP3	FP1&FP3	B	Incr		BL	
C	LP (a)	—	C	Decr		—	
D	LP	Decr LP	D	Decr		FL1	
E	LP	Incr LP	E	Decr		FL2	
F*	LP	Load Link	F	Decr		BL	

(a) LP = Link Pointer

Hex Code	Operate on X	Operate on Y	Shift	Hex Code	Input Source	Mirror
0				0	Common Rgtr	
1			◆	1	M	
2				2	Y	
3			◆	3	X	
4		◆		4	PI/O	
5		◆	◆	5	Array Mem	
6		◆		6	X or Y	
7		◆	◆	7	Mask Gen	
8	◆ Cond(3)			8	Common Rgtr	◆
9	◆ Cond		◆	9	M	◆
A	◆			A	Y	◆
B	◆		◆	B	X	◆
C	◆Cond	◆		C	PI/O	◆
D	◆Cond	◆	◆	D	Array Mem	◆
E	◆	◆		E	X or Y	◆
F	◆	◆	◆	F	Mask Gen	◆

Hex Code	Logic Function
00	Exclusive OR Complement Input
11	Inclusive OR Complement Input
22	Logical AND
33	No-op
44	Complement Input
55	NOT Inclusive OR
66	AND Complement Input
77	Clear to zero
88	Input
99	Inclusive OR
AA	NOT AND Complement Input
BB	Set to one
CC	Exclusive OR
DD	NOT Inclusive OR Complement Input
EE	NOT AND
FF	NEGATE

(1) In direct mode bits 8-15 contain the Array Address or a shift constant for Response Store to Response Store operations.

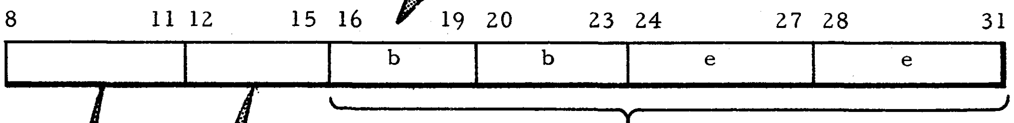
(2) Mask Generate and Load Common Register instructions have a different instruction Format for bits 8-31. (see page A2)

(3) X conditional on Y being set.

(4) If selected Common Register bit = 1, the hex code for bits 16-19 determines the logic function; if the selected Common Register bit = 0, the hex code for bits 20-23 determines the logic function; if the Common Register is not considered, the two hex codes will be equal as shown in the table.

GENERATE MASK AND LOAD COMMON REGISTER FORMATS

Table below assumes no shift. If shift is required, this hex digit will change. (See details on shift in Load Common Register Instruction section of manual.)



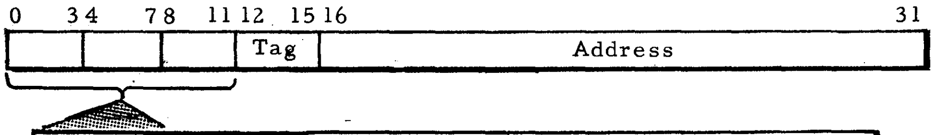
Field Select Hex Code	Bits
0	0-31
1	32-63
2	64-95
3	96-127
4	128-159
5	160-191
6	192-223
7	224-255

Hex Code	Selected Field Pointer	Bump
C	LP[(a)]	-
D	LP	Decr LP
E	LP	Incr LP

(a) LP = Link Pointer

Field Start and End Bit Positions (Assumes no shift.)

Starting Bit Position (Leftmost)	bb Hex Code (Bits 16-23)	Ending Bit Position (Rightmost)	ee Hex Code (Bits 24-31) INPUT				
			Y	X	M	Array Memory	Mask Generator
0	A0	0	02	03	01	05	07
1	A1	1	0A	0B	09	0D	0F
2	A2	2	12	13	11	15	17
3	A3	3	1A	1B	19	1D	1F
4	A4	4	22	23	21	25	27
5	A5	5	2A	2B	29	2D	2F
6	A6	6	32	33	31	35	37
7	A7	7	3A	3B	39	3D	3F
8	A8	8	42	43	41	45	47
9	A9	9	4A	4B	49	4D	4F
10	AA	10	52	53	51	55	57
11	AB	11	5A	5B	59	5D	5F
12	AC	12	62	63	61	65	67
13	AD	13	6A	6B	69	6D	6F
14	AE	14	72	73	71	75	77
15	AF	15	7A	7B	79	7D	7F
16	B0	16	82	83	81	85	87
17	B1	17	8A	8B	89	8D	8F
18	B2	18	92	93	91	95	97
19	B3	19	9A	9B	99	9D	9F
20	B4	20	A2	A3	A1	A5	A7
21	B5	21	AA	AB	A9	AD	AF
22	B6	22	B2	B3	B1	B5	B7
23	B7	23	BA	BB	B9	BD	BF
24	B8	24	C2	C3	C1	C5	C7
25	B9	25	CA	CB	C9	CD	CF
26	BA	26	D2	D3	D1	D5	D7
27	BB	27	DA	DB	D9	DD	DF
28	BC	28	E2	E3	E1	E5	E7
29	BD	29	EA	EB	E9	ED	EF
30	BE	30	F2	F3	F1	F5	F7
31	BF	31	FA	FB	F9	FD	FF

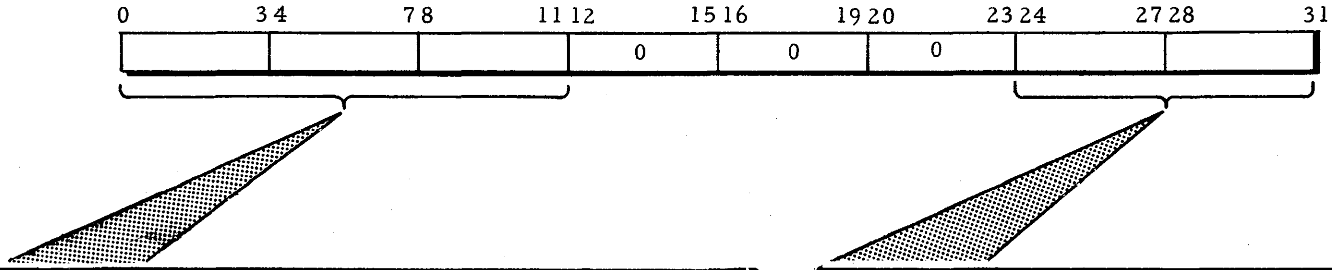


	Hex Code (Bits 0-11)	Operation
BRANCH INSTRUCTIONS Unconditional Branch	280	Branch Unconditionally
	281	No Branch
Conditional Branches	282	Branch if FP1=0
	283	Branch if FP1≠0
	284	Branch if FP2=0
	285	Branch if FP2≠0
	286	Branch if FP3=0
	287	Branch if FP3≠0
	288	Branch if FL1=0
	289	Branch if FL1≠0
	28A	Branch if FL2=0
	28B	Branch if FL2≠0
	28C	Common Register = 0 (1)
	28D	Common Register = 1 (1)
	28E	If all Y register bits in all selected arrays = 0(2)
	28F	If any Y register bit in any selected arrays= 1(2)
	290	Branch if BL=0
	291	Branch if BL≠0
	292	Branch if FPE=0
	293	Branch if FPE≠0
	294	Branch if DP=0
	295	Branch if DP≠0
Branch and Link Instruction	2C0	Store PC in R0; Branch to effective address
	2C1	Store PC in R1; Branch to effective address
	2C2	Store PC in R2; Branch to effective address
	2C3	Store PC in R3; Branch to effective address
	2C4	Store PC in R4; Branch to effective address
	2C5	Store PC in R5; Branch to effective address
	2C6	Store PC in R6; Branch to effective address
	2C7	Store PC in R7; Branch to effective address

(1) Rightmost 5 bits of FP1 selects tested Common Register bit.

(2) Selected Arrays specified in AS register; MASK bit must be set to 1 to select participating words.

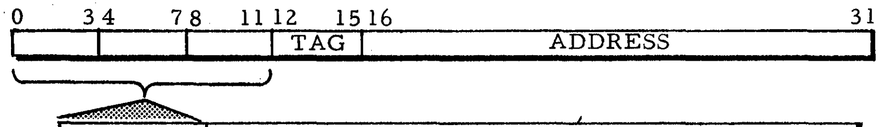
STORE
REGISTER
TO
REGISTER
INSTRUCTIONS



Hex Code (Bits 0-11)	SOURCE REGISTERS OPTIONS			
	Bits 0-7	Bits 8-15	Bits 16-23	Bits 24-31
300	C			
301	C1	CL		C0
302	CL		CH	
303	C3	CH		C2
304	AS			
305	AS1	ASL		AS0
306	ASL		ASH	
307	AS3	ASH		AS2
308	BL		DP	
309	BL1	DP		BL0
30A	DP		BL	
30B	DP1	BL		DP0
30C	PC		0	IMASK
30D	PC1	0	IMASK	PC0
30E	0	IMASK	PC	
30F	IMASK	PC		0
310	FL2	FPE	0	0
311	FPE	0	0	FL2
312	0	0	FL2	FPE
313	0	FL2	FPE	0
31C	FL1	FP3	FP1	FP2
31D	FP3	FP1	FP2	FL1
31E	FP1	FP2	FL1	FP3
31F	FP2	FL1	FP3	FP1

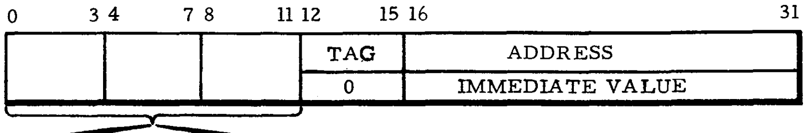
Hex Code (Bits 24-31)	DESTINATION REGISTERS OPTIONS			
	Bits 0-7	Bits 8-15	Bits 16-23	Bits 24-31
08	▨	▨	CL	
09	▨	▨	ASL	
0A	▨	▨	DP	
0B	▨	▨	▨	IMASK
0C	FL2	▨	▨	▨
0D	▨	▨	▨	FP2
0E	▨	▨	FP1	▨
0F	▨	▨	FP1	FP2
10	CH		▨	▨
11	ASH		▨	▨
12	BL		▨	▨
13	PC		▨	▨
14	▨	FPE	▨	▨
15	FL1	▨	▨	▨
16	▨	FP3	▨	▨
17	FL1	FP3	▨	▨
18	C			
19	AS			
1A	BL		DP	
1B	PC		▨	IMASK
1C	FL2	FPE	▨	▨
1D	FL1	▨	▨	FP2
1E	▨	FP3	FP1	▨
1F	FL1	FP3	FP1	FP2

STORE
REGISTER
INSTRUCTIONS



	Hex Code (Bits 0-11)	REGISTER(S) STORED TO 32-BIT MEMORY WORD			
		Bits 0-7	Bits 8-15	Bits 16-23	Bits 24-31
	300	C			
	301	C1	CL		C0
	302	CL		CH	
	303	C3	CH		C2
	304	AS			
	305	AS1	AL		AS0
	306	ASL		ASH	
	307	AS3	ASH		AS2
	308	BL		DP	
	309	BL1	DP		BL0
	30A	DP		BL	
	30B	DP1	BL		DP0
Swap PSW	30C	PC		0	▨ IMASK
	30D	PC1	0	▨ IMASK	PC0
	30E	0	▨ IMASK	PC	
	30F	▨ IMASK	PC		0
	310	FL2	FPE	0	0
	311	FPE	0	0	FL2
	312	0	0	FL2	FPE
	313	0	FL2	FPE	0
	31C	FL1	FP3	FP1	FP2
	31D	FP3	FP1	FP2	FL1
	31E	FP1	FP2	FL1	FP3
	31F	FP2	FL1	FP3	FP1

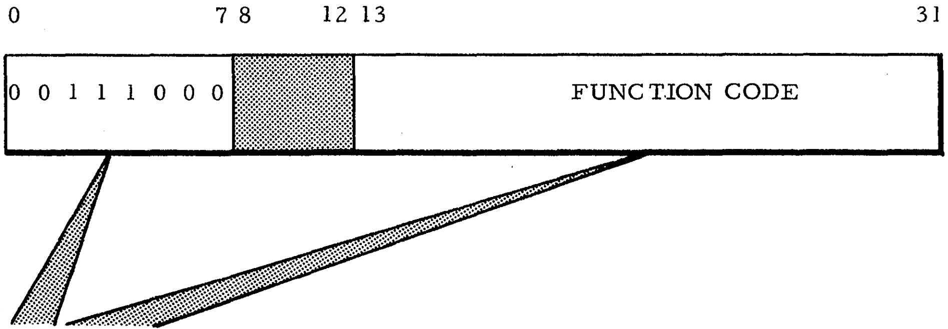
LOAD REGISTER AND LOAD IMMEDIATE[2] INSTRUCTIONS



<table>
<tr><th></th><th colspan="4">SOURCE[1]</th></tr>
<tr><th rowspan="3">HEX CODE
(Bits 0-11)</th><th colspan="2">0</th><th colspan="2">IMMEDIATE VALUE</th></tr>
<tr><th colspan="4">MEMORY</th></tr>
<tr><th>Bits 0-7</th><th>Bits 8-15</th><th>Bits 16-23</th><th>Bits 24-31</th></tr>
<tr><td>320</td><td></td><td></td><td colspan="2">CL</td></tr>
<tr><td>321</td><td>C3</td><td></td><td></td><td>C2</td></tr>
<tr><td>322</td><td colspan="2">CL</td><td></td><td></td></tr>
<tr><td>323</td><td colspan="4">CL</td></tr>
<tr><td>324</td><td></td><td></td><td colspan="2">ASL</td></tr>
<tr><td>325</td><td>AS3</td><td></td><td></td><td>AS2</td></tr>
<tr><td>326</td><td colspan="2">ASL</td><td></td><td></td></tr>
<tr><td>327</td><td colspan="4">ASL</td></tr>
<tr><td>328</td><td></td><td></td><td colspan="2">DP</td></tr>
<tr><td>329</td><td>DP1</td><td></td><td></td><td>DP0</td></tr>
<tr><td>32A</td><td colspan="2">DP</td><td></td><td></td></tr>
<tr><td>32B</td><td colspan="4">DP</td></tr>
<tr><td>32C</td><td></td><td></td><td></td><td>IMASK</td></tr>
<tr><td>32D</td><td>IMASK</td><td></td><td></td><td></td></tr>
<tr><td>32E</td><td></td><td>IMASK</td><td></td><td></td></tr>
<tr><td>32F</td><td></td><td></td><td>IMASK</td><td></td></tr>
<tr><td>330</td><td>FL2</td><td></td><td></td><td></td></tr>
<tr><td>331</td><td></td><td>FL2</td><td></td><td></td></tr>
<tr><td>332</td><td></td><td></td><td>FL2</td><td></td></tr>
<tr><td>333</td><td></td><td></td><td></td><td>FL2</td></tr>
<tr><td>334</td><td></td><td></td><td></td><td>FP2</td></tr>
<tr><td>335</td><td>FP2</td><td></td><td></td><td></td></tr>
<tr><td>336</td><td></td><td>FP2</td><td></td><td></td></tr>
<tr><td>337</td><td></td><td></td><td>FP2</td><td></td></tr>
<tr><td>338</td><td></td><td></td><td>FP1</td><td></td></tr>
<tr><td>339</td><td></td><td></td><td></td><td>FP1</td></tr>
<tr><td>33A</td><td>FP1</td><td></td><td></td><td></td></tr>
<tr><td>33B</td><td></td><td>FP1</td><td></td><td></td></tr>
<tr><td>33C</td><td></td><td></td><td>FP1</td><td>FP2</td></tr>
<tr><td>33D</td><td>FP2</td><td></td><td></td><td>FP1</td></tr>
<tr><td>33E</td><td>FP1</td><td>FP2</td><td></td><td></td></tr>
<tr><td>33F</td><td></td><td>FP1</td><td>FP2</td><td></td></tr>
<tr><td>340</td><td colspan="2">CH</td><td></td><td></td></tr>
<tr><td>341</td><td colspan="4">CH</td></tr>
<tr><td>342</td><td></td><td></td><td colspan="2">CH</td></tr>
<tr><td>343</td><td>C1</td><td></td><td></td><td>C0</td></tr>
<tr><td>344</td><td colspan="2">ASH</td><td></td><td></td></tr>
<tr><td>345</td><td colspan="4">ASH</td></tr>
<tr><td>346</td><td></td><td></td><td colspan="2">ASH</td></tr>
<tr><td>347</td><td>AS1</td><td></td><td></td><td>AS0</td></tr>
<tr><td>348</td><td colspan="2">BL</td><td></td><td></td></tr>
<tr><td>349</td><td colspan="4">BL</td></tr>
<tr><td>34A</td><td></td><td></td><td colspan="2">BL</td></tr>
<tr><td>34B</td><td>BL1</td><td></td><td></td><td>BL0</td></tr>
<tr><td>34C</td><td colspan="2">PC</td><td></td><td></td></tr>
<tr><td>34D</td><td colspan="4">PC</td></tr>
<tr><td>34E</td><td></td><td></td><td colspan="2">PC</td></tr>
<tr><td>34F</td><td>PC1</td><td></td><td></td><td>PC0</td></tr>
<tr><td>350</td><td></td><td>FPE</td><td></td><td></td></tr>
<tr><td>351</td><td></td><td></td><td>FPE</td><td></td></tr>
<tr><td>352</td><td></td><td></td><td></td><td>FPE</td></tr>
<tr><td>353</td><td>FPE</td><td></td><td></td><td></td></tr>
<tr><td>354</td><td>FL1</td><td></td><td></td><td></td></tr>
<tr><td>355</td><td></td><td>FL1</td><td></td><td></td></tr>
<tr><td>356</td><td></td><td></td><td>FL1</td><td></td></tr>
<tr><td>357</td><td></td><td></td><td></td><td>FL1</td></tr>
<tr><td>358</td><td></td><td>FP3</td><td></td><td></td></tr>
<tr><td>359</td><td></td><td></td><td>FP3</td><td></td></tr>
<tr><td>35A</td><td></td><td></td><td></td><td>FP3</td></tr>
<tr><td>35B</td><td>FP3</td><td></td><td></td><td></td></tr>
<tr><td>35C</td><td>FL1</td><td>FP3</td><td></td><td></td></tr>
<tr><td>35D</td><td></td><td>FL1</td><td>FP3</td><td></td></tr>
<tr><td>35E</td><td></td><td></td><td>FL1</td><td>FP3</td></tr>
<tr><td>35F</td><td>FP3</td><td></td><td></td><td>FL1</td></tr>
<tr><td>360</td><td colspan="4">C</td></tr>
<tr><td>361</td><td>C3</td><td colspan="2">CH</td><td>C2</td></tr>
<tr><td>362</td><td colspan="2">CL</td><td colspan="2">CH</td></tr>
<tr><td>363</td><td>C1</td><td colspan="2">CL</td><td>C0</td></tr>
<tr><td>364</td><td colspan="4">AS</td></tr>
<tr><td>365</td><td>AS3</td><td colspan="2">ASH</td><td>AS2</td></tr>
<tr><td>366</td><td colspan="2">ASL</td><td colspan="2">ASH</td></tr>
<tr><td>367</td><td>AS1</td><td colspan="2">ASL</td><td>AS0</td></tr>
<tr><td>368</td><td colspan="2">BL</td><td colspan="2">DP</td></tr>
<tr><td>369</td><td>DP1</td><td colspan="2">BL</td><td>DP0</td></tr>
<tr><td>36A</td><td colspan="2">DP</td><td colspan="2">BL</td></tr>
<tr><td>36B</td><td>BL1</td><td colspan="2">DP</td><td>BL0</td></tr>
<tr><td>36C</td><td colspan="2">PC</td><td></td><td>IMASK</td></tr>
<tr><td>36D</td><td>IMASK</td><td colspan="2">PC</td><td></td></tr>
<tr><td>36E</td><td></td><td>IMASK</td><td colspan="2">PC</td></tr>
<tr><td>36F</td><td>PC1</td><td></td><td>IMASK</td><td>PC0</td></tr>
<tr><td>370</td><td>FL2</td><td>FPE</td><td></td><td></td></tr>
<tr><td>371</td><td></td><td>FL2</td><td>FPE</td><td></td></tr>
<tr><td>372</td><td></td><td></td><td>FL2</td><td>FPE</td></tr>
<tr><td>373</td><td>FPE</td><td></td><td></td><td>FL2</td></tr>
<tr><td>374</td><td>FL1</td><td></td><td></td><td>FP2</td></tr>
<tr><td>375</td><td>FP2</td><td>FL1</td><td></td><td></td></tr>
<tr><td>376</td><td></td><td>FP2</td><td>FL1</td><td></td></tr>
<tr><td>377</td><td></td><td></td><td>FP2</td><td>FL1</td></tr>
<tr><td>378</td><td></td><td>FP3</td><td>FP1</td><td></td></tr>
<tr><td>379</td><td></td><td></td><td>FP3</td><td>FP1</td></tr>
<tr><td>37A</td><td>FP1</td><td></td><td></td><td>FP3</td></tr>
<tr><td>37B</td><td>FP3</td><td>FP1</td><td></td><td></td></tr>
<tr><td>37C</td><td>FL1</td><td>FP3</td><td>FP1</td><td>FP2</td></tr>
<tr><td>37D</td><td>FP2</td><td>FL1</td><td>FP3</td><td>FP1</td></tr>
<tr><td>37E</td><td>FP1</td><td>FP2</td><td>FL1</td><td>FP3</td></tr>
<tr><td>37F</td><td>FP3</td><td>FP1</td><td>FP2</td><td>FL1</td></tr>
</table>

Note: [1] The register that receives the memory byte, half-word, or word is specified in the table blocks. The bits indicated at the top of each column indicate the position of the source data in the memory word.

[2] Shaded columns indicate registers that are cleared during Load Immediate instructions.



EXTERNAL FUNCTION INSTRUCTIONS

	Hex Code (Bits 0-31)	Operation	Sequential Processor Octal Code	
			EFS	EFB
Page Port Switches	380002A0	Switch Page 0 to Instruction Bus	0	1240
	380006A0	Switch Page 1 to Instruction Bus	0	3240
	38000AA0	Switch Page 2 to Instruction Bus	0	5240
	380003F0	Switch Page 0 to Pager Bus	0	1760
	380007F0	Switch Page 1 to Pager Bus	0	3760
	38000BF0	Switch Page 2 to Pager Bus	0	5760
	38000150	Switch Page 0 to Sequential Bus	0	0520
	38000550	Switch Page 1 to Sequential Bus	0	2520
	38000950	Switch Page 2 to Sequential Bus	0	4520
	380001B2	Sense if Page 0 on Instruction Bus	0	0662
	380005B2	Sense if Page 1 on Instruction Bus	0	2662
	380009B2	Sense if Page 2 on Instruction Bus	0	4662
	380001B1	Sense if Page 0 on Pager Bus	0	0661
	380005B1	Sense if Page 1 on Pager Bus	0	2661
	380009B1	Sense if Page 2 on Pager Bus	0	4661
	380001B4	Sense if Page 0 on Sequential Bus	0	0664
	380005B4	Sense if Page 1 on Sequential Bus	0	2664
	380009B4	Sense if Page 2 on Sequential Bus	0	4664
Interlocks	38006(n)C	Set Interlock n (00 ≤ n ≤ 3F)	0	
	38006(n)0	Clear Interlock n	0	
	38006(n)5	Sense if Interlock n Set	0	
	38006(n)6	Sense if Interlock n Clear	0	
	38006(n)1	Sense if Interlock n Set, Then Clear it	0	
	38006(n)D	Sense if Interlock n Set, Then Set it	0	
	38006(n)2	Sense if Interlock n Clear, Then Clear it	0	
	38006(n)E	Sense if Interlock n Clear, Then Set it	0	
Pager	3804(ADDR)	Start Pager at an Address	4	(Addr)
	38008000	Stop Pager	0	100000
	3800800C	Start Pager Where Previously Stopped	0	100014
	38008005	Sense if Pager Busy	0	100005
	38008006	Sense if Pager Off	0	100006
Error Detectors	38004(n)6	Sense if Error Detector n is off	0	
	38004(n)5	Sense if Error Detector n is On	0	
	38004(n)0	Turn Off Error Detector n	0	
	38004(n)C	Turn On Effor Detector n	0	
Error Interrupt Enable/Disable	380040DC	Enable Error Interrupts	0	40334
	380040D0	Disable Error Interrupts	0	40320
Single Step	380040EC	Put AP in Single Step Mode	0	40354
	380040E0	Put AP in Continuous Mode	0	40340
	380022F0	Do One AP Instruction (Single Step Mode Only)	0	21360
AP Trap	380040FC	Set AP Trap	0	40374
	380040F0	Clear AP Trap	0	40360
	380040F5	Sense if AP Trap Set	0	40365
	380040F6	Sense if AP Trap Clear	0	40366
AP Interrupts	38010(n)C	Set AP Interrupt n (01 ≤ n ≤ 0F)	1	
	38010(n)0	Clear AP Interrupt n	1	
	38010(n)5	Sense if AP Interrupt n Set	1	
	38010(n)6	Sense if AP Interrupt n Clear	1	
Sequential Control Interrupts	38012(n)C	Set Sequential Interrupt n (C0 ≤ n ≤ DC)	1	
	38012(n)0	Clear Sequential Interrupt n	1	
	38012(n)5	Sense if Sequential Interrupt n Set	1	
	38012(n)6	Sense if Sequential Interrupt n Clear	1	

EXTERNAL FUNCTION INSTRUCTIONS (cont)

	Hex Code (Bits 0-31)	Operation	Sequential Processor Octal Code	
			EFS	EFB
AP Activity	3800200C	Start AP at Location 8000_{16}	0	20014
	38002000	Stop AP	0	20000
	38002005	Sense if AP Active	0	20005
	38002006	Sense if AP Inactive	0	20006
Loop Indicator	38002011	Sense if AP in Loop Mode	0	20021
	38002012	Sense if AP Not in Loop Mode	0	20022
	38002014	Take AP Out of Loop Mode	0	20024
Clears	38002020	Clear EXF Resolve	0	20040
	38002030	Clear Errors, Seq. Interrupts	0	20060
	38002040	Clear AP Interrupts	0	20100
	38002050	Clear DMA Logic	0	20120
	38002060	Clear Core Logic	0	20140
	38002070	Clear Next Instruction Logic	0	20160
	38002080	Clear Branch Logic	0	20200
	38002090	Clear Register Logic	0	20220
	380020A0	Clear Pager	0	20240
	380020B0	Clear HSDB Logic	0	20260
	380020D0	Clear Associative Instruction Logic	0	20320
	380020E0	Clear Page 0 Logic	0	20340
	380020F0	Clear Page 1 Logic	0	20360
	38002100	Clear Page 2 Logic	0	20400
Master Clear	38002300	Master Clear	0	21400

LOOP, and LOAD and LOOP INSTRUCTIONS

LOOP INSTRUCTIONS

0 11	12 15	16 31
	TAG	ADDRESS

Instruction Hex Code (Bits 0-11)	Operation
3C0	Single Instruction Loop
3D0	Multi-Instruction Loop

LOAD & LOOP INSTRUCTION

0 7	8 15	16 31
	LOOP Count	ADDRESS

Instruction Hex Code (Bits 0-7)	Operation
3E	Single Instruction Loop
3F	Multi-Instruction Loop

PAGER INSTRUCTIONS

Hex code (Bits 0-31)	Operation
000nnnnn (nnnnn = external function code; same as EXTERNAL FUNCTION INSTRUCTIONS)	Issue EXF from PAGER
00008000	HALT Pager
00008001	HALT Pager and Skip next Instruction
00008005	Skip next Instruction
0004nnnn (nnnn- new GET address)	Pager Branch to new GET address
4000 nnnn (nnnn -PUT address)	Load PUT address register

Binary Code			Operation
Bits 0-1	Bits 2-15	Bits 16-31	
01	Count (no. of words to be moved)	All zeros	Move Data
11	Count (no. of words to be moved)	PUT address	Load PUT address and Move Data

DEFINITION OF TERMS

AP CONTROL INTERRUPTS

AP control interrupts are given hex addresses from 01_{16} (lowest priority) to $0F_{16}$ (highest priority). A class of external codes is used to sense, set, and reset the state of the 15 interrupts to AP control. Bits 28 through 31 of the program status word (PSW) in AP control contain an interrupt mask. AP control accepts interrupt n if AP control is active and at an interruptable point, if the interrupt mask is less than n, if no interrupt of higher priority is set, and if interrupt n is set. When it accepts interrupt n, it fetches its next instruction from hex address 8000+n (without disturbing the contents of its program counter).

AP CONTROL MEMORY

The main function of AP control memory is to store assembled AP application programs. It can also store items of data and act as a buffer between AP control and other elements of STARAN S. Each AP control memory word contains 32 bits of either data or instructions. Bit 0 is the left (most-significant) bit and bit 31 is the right (least-significant) bit of each word. Each word is given a 16-bit address.

ARRAY

The associative array consists of two basic components: array storage and response store. Each array contains 65,536 bits, organized as a square 256 words by 256 bits of solid state storage. Access may be made in either the bit or word direction. An entire word of 256 bits or a bit slice, bit n of all 256 words, may be accessed. Array input and output may be either 32 bits or 256 bits in parallel. Input data may be written into the array through a mask contained in the response store.

ARRAY ADDRESS SELECT

The array address select logic in array control of AP control selects either the array select register or field pointer 1 to generate the array enable signals. When field pointer 1 is selected, the rightmost five bits of the pointer are decoded to indicate the one and only array to be enabled. Selection is made without modifying the contents of the array select register. Such operations as reading one item of data from an array or writing one item of data into an array enable only one of the associative arrays. The Array Select register selects arrays when more than one array participates in an associative operation.

ARRAY CONTROL

Control lines from AP control to the associative arrays that are not generated by the response store control are generated by the array control. The array control logic selects the arrays that are to be used and controls such things as bit/word mode, store masked, and shifting. The array control logic includes:

1) array select register
2) array address select
3) array mode
4) shift control

ARRAY MODE

The array mode logic controls the addressing mode. Either a bit slice or a word slice in the arrays is selected for loading or storing.

ARRAY SELECT REGISTER

The array select register in array control establishes those array modules that are to be active for an associative operation. The array select register is 32 bits wide. Each bit position controls one array. Bit 0 corresponds to array 0, and a one in a bit position enables the corresponding array. The array select register is loaded and read through the bus logic. The array select register contents are also used by the resolver logic.

BIT COLUMN

A bit column is a selected bit in every word in an array.

BITS

An array word consists of 256 bits, all of which may be accessible in parallel or divisible into eight fields of 32 bits each. Addressing a bit position within an array is accomplished by an address between 000, the most-significant (left) bit, and 255, the least-significant (right) bit position. Bit (n) of all words is accessible from address (n).

BLOCK LENGTH COUNTER

The block length counter is a 16-bit decrementing counter. The block length counter in the bus logic of AP control controls the length of a data block transfer.

BRANCH AND LINK REGISTERS

A group of 8 registers used to facilitate subroutine linkage. The Branch and Link Registers occupy a dedicated area of the High Speed Data Buffer (600_{16} to 607_{16}).

BUFFERED INPUT/OUTPUT

Buffered I/O (BI/O) is available for tying several different types of peripherals into the AP control memory of STARAN S. In addition, BI/O can be used to transfer blocks of data and/or programs between the AP control memory and the host-computer memory. The basic width of the BI/O interface is 32 bits plus a parity bit. The custom I/O cabinet can include buffers that allow a wide variety of repacking to take place so that I/O channels of any width can be assembled. Normally, the EXF channel is used to set up and initiate buffered I/O transfers. Of the several data buses accessible to the high-speed data buffer (HSDB), the bulk core memory, and the external memory accessed from a DMA channel, the priority resolver logic gives the BI/O port highest priority.

BULK CORE MEMORY

The bulk core memory is a section of AP control memory using nonvolatile core storage. In the standard configuration it contains 16,384 words and is expandable to 32,768 words. Like the high-speed data buffer, bulk core memory is accessible to all buses that can gain access to AP control memory (a priority port switch giving each memory cycle to the highest priority bus requesting a bulk core memory address). The priority of the buses is the same as for the high-speed data buffer. Since it is large and nonvolatile, the bulk core memory is useful for storing the AP control programs. Because it is slower than the page memories, it is recommended that program segments be paged into the page memories for execution. Since it is accessible by all buses accessible to AP control memory, it is also useful as a buffer for data items that do not require the higher speed of the HSDB.

BUS LOGIC

The bus logic provides a common data path for all pertinent registers of AP control and the data bus from AP control memory. The bus is 32 bits wide. Registers of less than 32 bits are grouped to form a 32-bit word.

BYTES

A byte is a portion of a word consisting of 8 consecutive bits.

COMMON REGISTER

The common register is an AP control register that contains 32 bits numbered 0 to 31. Bit 0 is the left (most-significant) bit. Bit 31 is the right (least-significant bit. The common register may contain the argument for a search operation performed upon the associative arrays, the input data stored into an array, or the input data received from an array in a load operation. Data from an array is loaded into into the common register through a mask generated by the mask generator.

COMPARATOR

The comparator is a portion of the program control logic in AP control that compares the address contained in the end loop marker with the program counter. The comparator transmits an indication to control when the end of a loop sequence has been reached. Control then loads the start loop register contents into the program counter if there are more repetitions to be completed in the loop count.

CONTROL LINE BUFFER

The control line buffer in the response store control controls the timing of the control lines transmitted to the associative arrays.

CONTROL LINE CONDITIONER

The control line conditioner in the response store control generates the control lines required to manipulate the response store. Control line signals are generated as a function of the instruction register, a selected bit of the Common Register, and the inclusive OR output from the resolver.

DATA POINTER

The data pointer in the bus logic of AP control contains the control memory address for the data bus for block transfers. It is a 16-bit register. The data pointer can be stepped with each transfer within a data block.

DIRECT MEMORY ACCESS

A block of AP control memory addresses is reserved for the direct memory access (DMA) channel to external memory. In the standard configuration this block can contain up to 30,720 addresses. Direct access to a host-computer memory allows that memory to be shared with the AP control memory and a host computer. Items in the memory are equally accessible by a host computer and by STARAN S, thus reducing the need for buffered I/O transfers between the host computer and STARAN S.

END LOOP MARKER

The end loop marker is a 16-bit register in the program control logic of AP control. It is used to store the last address of an instruction loop. The end loop marker register is loaded from the rightmost 16 bits of the loop instruction format.

EXTERNAL FUNCTION LOGIC

The external function (EXF) logic facilitates coordination between the different elements of STARAN S. By issuing external function codes to the EXF logic, an element of STARAN S can control and interrogate the status of other elements. Each function code is 19 bits long and indicates what element of STARAN S is to be interrogated and/or controlled. For each function code transmitted to EXF logic, the logic returns a single sense bit indicating the result of the interrogation. Function codes may be transmitted to EXF logic by AP control, the program pager, sequential control, and a host computer (if the EXF channel to the host computer is implemented).

FIELD LENGTH COUNTERS

Field length counters are 8-bit AP control registers used to contain the length of data fields. They may be decremented to allow stepping through the bits of a data field. When the counter's contents equal zero, an indication is sent to the AP control for test purposes. There are two field length counters: FL1 and FL2.

FIELD POINTER

A field pointer is an 8-bit AP control register that generally contains an array bit column or word address. Field pointers may be incremented or decremented to facilitate stepping through data fields. There are four field pointers: FP1, FP2, FP3, and FPE.

FIELD
POINTER 1

Field pointer 1 is an 8-bit AP control register that may contain an array bit column or word address for the indirect addressing mode. Field pointer 1 is used by the resolver for the address of the array module containing the first responder.

FIELD
POINTER 2

Field pointer 2 is an 8-bit AP control register that may contain an array bit column or word address. Field pointer 2 is also used by the resolver for the array bit column or word address of the first responder in the array specified in FP1.

FIELD
POINTER 3

Field pointer 3 is an 8-bit AP control register that may contain an array bit column or word address.

FIELD
POINTER E

Field pointer E is an 8-bit AP control register that may contain an array bit column or word address or a shift constant.

FIELDS
(SECTIONS)

Each 256-bit word in an array contains eight fields (or sections). Each field contains 32 contiguous bits within the word being addressed. Addressing of a particular field of an array word is accomplished by an address between 0, the most-significant (left) field, and 7, the least-significant (right) field. The most-significant field starts at the most-significant bit position.

FLIP-
FLOP

A flip-flop is a bistable device capable of storing a bit of information.

GET
ADDRESS
REGISTER

The GET address register in the program pager holds a 16-bit AP control memory address. If the pager is in the midst of moving data, the GET address points to the memory location holding the next source word to be moved. If the pager is executing instructions, the GET address acts like a program counter that points to the location of the next pager memory instruction.

HIGH-SPEED DATA BUFFER

The high-speed data buffer (HSDB) is a section of AP control memory using fast bipolar solid-state elements. In the standard configuration of STARAN S it contains 512 words. All buses accessible to AP control memory can gain access to the HSDB, making it a convenient place to store data and instruction items that need to be accessed quickly by different elements of STARAN S.

HOST COMPUTER

A conventional sequential-type computer integrated with STARAN S to provide efficient processing for sequential operations. (STARAN S performs tasks requiring parallel operations.)

INSTRUCTION REGISTER

The instruction register in AP control contains the current instruction being executed. The instruction loaded into the instruction register is received from AP control memory through the instruction bus. Parity is checked at the instruction register. The instruction register contains 32 bits, which are numbered from 0 to 31, with bit 0 at the left. Portions of the instruction register are used as a direct source of immediate data or addresses as a function of the instruction being executed.

INTERLOCKS

The EXF logic contains 64 stored bits called interlocks. These bits have no predetermined meaning. Software may assign a meaning to an interlock and use it for any purpose. Function codes allow the current state of an interlock to be sensed and a new state to be entered in one operation. Sixteen interlocks (hex addresses 00 through 0F) can be controlled and sensed manually by panel switches and lights to facilitate communication with an operator. The other 48 interlocks (hex addresses 10 through 3F) can be sensed and controlled only by function codes.

INTERRUPT MASK

The program status word in the program control logic contains the interrupt mask for the 15 AP control interrupts. All interrupts with numbers greater than the mask are accepted. The interrupt mask is contained in bits 28 through 31 of the program status word.

LEAST-SIGNIFICANT BIT

A significant bit is a bit that contributes to the precision of a numeric value. The number of significant bits is counted beginning with the bit contributing the most value, called the most-significant bit, and ending with the one contributing the least value, called the least-significant bit. In STARAN S memory, the least-significant bit is bit 31 in a 32-bit word or field, or bit 255 in a 256-bit array word.

M RESPONSE STORE REGISTER

The M response store register (MASK) is a 256-bit register contained in the response store element of each array. Its special use is to select array words participating in an associative operation.

MASK GENERATOR

The mask generator in AP control, generates a mask pattern to be used in loading array output data into the common register. The mask enables data to be loaded for a number of contiguous bits. The mask generator requires the bit addresses of the most and least-significant bits of the field to be loaded. All bits between and including these limits are loaded while those outside these limits are unaltered.

MIRRORING

Mirroring will cause the 256-bit input quantity to an associative instruction to be flipped end-for-end (i. e., bit i is put into bit 255-i).

MOST-SIGNIFICANT BIT

A significant bit is a bit that contributes to the precision of a numeric value. The number of significant bits is counted beginning with the bit contributing the most value, called the most-significant bit, and ending with the bit contributing the least value, called the least-significant bit. In STARAN S memory the most-significant bit is bit 0.

PAGE MEMORY

Three page memories are included in the AP control memory: page 0, page 1, and page 2. Each page memory uses fast bipolar solid-state elements and is volatile. Each page contains 512 words in the standard STARAN S configuration.

PAGE PORT SWITCHES

Each page memory has a port switch that connects the memory to the AP control instruction bus, the pager bus, or the sequential control bus. A page port switch function code is used for interrogation and control of these switches.

PARALLEL INPUT/ OUTPUT
* OPTIONAL

Each associative array in STARAN S can have up to 256 inputs and 256 outputs into the custom I/O cabinet. The basic width of the PI/Ois 256n, where n is equal to the number of associative arrays in the system (n can have a maximum value of 32). The custom I/O cabinet is capable of buffering and reformatting the data received from any peripheral device to match the width necessary to communicate with the STARAN S associative array. In order to synchronize read and write operations with an external device operating through PI/O, certain AP control instructions wait for sync pulses from the external device. As in inter-array communication, an external function code sent to the custom I/O will set up and initiate the PI/O operation.

PROGRAM COUNTER

The program counter occupies bits 0-15 of the program status word in AP control. The program counter contains the address of the current instruction being executed. It is normally incremented sequentially through control memory. Its normal sequence may be altered by a branch or loop instruction.

PROGRAM PAGER

The program pager loads the high-speed page memories from the bulk core portion of AP control memory. The pager performs these transfers independent of AP control, so that while AP control is executing a program segment out of one page memory, the pager can be loading another page memory with a future program segment. Pager operation is initiated by external function codes. The program pager contains three registers: a GET address register, a PUT address register, and a word count register. The GET address register holds a 16-bit AP control memory address. If the pager is in the midst of moving data, the GET address points to the memory location holding the next source word to be moved. When executing instructions, the GET address acts like a program counter, pointing to the location of the next page memory memory instruction. The PUT address register contains a 16-bit AP control address. It points to the memory location into which the next destination word is to be put during a move-data operation. The word count register, which is 14 bits in length, contains the number of words still to be transferred during a transfer operation.

PROGRAM STATUS WORD

The Program Status Word (PSW) consists of the program counter (PC) (bits 0-15), which contains the address of the current AP control instruction being executed, and the Interrupt Mask (IMASK) bits 28-31), which contains the current interrupt status. (All interrupts with numbers greater than the IMASK are accepted.)

PUT ADDRESS REGISTER

The PUT address register in the program pager holds a 16-bit page memory address. It points to the memory location into which the next destination word is to be put during a move-data operation.

REGISTER

A register is a memory device capable of containing one or more bits or words.

RESOLVER

The resolver logic in AP control finds the array address and word address of the first (most-significant) responder. The array address is loaded into field pointer 1 and the word address is loaded in field pointer 2. This allows subsequent operations to only affect the first responder of a search.

RESPONDER

A responder is a response store element in an enabled array whose Y register bit is set. Generally, responders indicate those words satisfying some search criteria.

RESPONSE STORE CONTROL

The response store control logic generates the control signals required by the associative arrays and buffers them to insure correct timing at the response store. The response store control consists of two basic portions: 1) the control line conditioner and 2) the control line buffer.

RESPONSE STORE ELEMENT

The response store portion of the array memory consists of 256 response store elements, each containing an X, Y, and M flip-flop. The 256 X flip-flops (bits) are considered the X response store register; the 256 Y flip-flops (bits) are considered the Y response store register; the 256 M flip-flops (bits) are considered the M response store register or the MASK.

SEQUENTIAL CONTROL

The sequential control block of STARAN S contains a sequential processor (SP) with 8K of memory, a keyboard/printer, a perforated tape reader/punch unit, and interface logic to connect the SP to other STARAN elements.

SEQUENTIAL CONTROL INTERRUPTS

Sequential control can accept interrupts from other STARAN S elements. The interrupts arise from error detection, the panel-interrupt button, or external functions. Eight different interrupt vector addresses are provided.

SEQUENTIAL PROCESSOR

The sequential processor is a 16-bit, general-purpose, parallel-logic computer using two's complement arithmetic for addressing 8,192 16-bit words (16,384 8-bit bytes of memory). All communication between system components is done on a single high-speed bus. There are eight general-purpose registers, which can be used as accumulators, index registers, or address pointers, and a multi-level automatic priority interrupt system.

SHIFT CONTROL

The shift control logic in array control generates the controls signals required by the array to perform shifting and mirroring operations.

SHIFT
LOGIC

Data transmitted from the bus logic of AP control passes through the bus shift logic. The bus shift logic is used to shift the 32-bit bus word left-end around by either 0-, 8-, 16-, or 24-bit positions. The shift is controlled by the instruction moving the data. Data received from AP control memory is checked for correct parity as it passes the bus shift logic. Data being stored in the control memory has a parity bit generated at the bus shift logic.

START
LOOP
MARKER

The start loop marker is a 16-bit register in the program control logic of AP control. The start loop marker is used to store the first address of the loop whenever a loop instruction is executed. The start loop marker is loaded directly from the program counter at the start of an instruction loop. It is read into the program counter whenever the last instruction of the loop is executed and the loop is to be repeated.

WORD
COUNT
REGISTER

The word count register is a 14-bit register in the program pager, which contains the number of words still to be transferred during a move-data operation.

WORDS

An array word consists of 256 bits,all of which may be accessible in parallel or divisible into fields of 32 bits each. Addressing of a word within an array is accomplished by using an address from 000, the first word, to 255, the last word. An AP control memory word is 32 bits in length. A sequential control word is 16 bits in length.

X RESPONSE
STORE
REGISTER

The X response store register is a 256-bit register contained in the response store element of each array. It may be used as temporary storage of data loaded from the array or stored in the array. It can be combined logically with data from the input network and/or the Y register. It is useful as temporary storage in parallel arithmetic operations or searches.

Y RESPONSE
STORE
REGISTER

The Y response store register is a 256-bit register contained in the response store element of each array. It may be used as temporary storage of data loaded from the array or stored in the array. It can be combined logically with data from the input network. It is useful as temporary storage in parallel arithmetic operations and searches. It is also used as the responder in a resolve operation.

ABBREVIATIONS

AND	Logical AND
AP	Associative Processor
AS	Array Select Register (32 bits)
ASH	Array Select Register, High Half (bits 0-15)
ASL	Array Select Register, Low Half (bits 16-31)
AS0	Array Select Register, Byte 0 (bits 0-7)
AS1	Array Select Register, Byte 1 (bits 8-15)
AS2	Array Select Register, Byte 2 (bits 16-23)
AS3	Array Select Register, Byte 3 (bits 24-31)
BI/O	Buffered Input/Output
BL	Block Length Counter (16 bits)
BL0	Block Length Counter Byte 0 (bits 0-7)
BL1	Block Length Counter Byte 1 (bits 8-15)
C	Common Register (32 bits)
CH	Common Register, High Half (bits 0-15)
CL	Common Register, Low Half (bits 16-31)
C0	Common Register, Byte 0 (bits 0-7)
C1	Common Register, Byte 1 (bits 8-15)
C2	Common Register, Byte 2 (bits 16-23)
C3	Common Register, Byte 3 (bits 24-31)
DMA	Direct Memory Access
DP	Data Pointer (16 bits)
DP0	Data Pointer, Byte 0 (bits 0-7)
DP1	Data Pointer, Byte 1 (bits 8-15)
ELM	End Loop Marker
EXF	External Function
EFB	External Function Buffer
EFS	External Function Status
FL1	Field Length Counter 1 (8 bits)
FL2	Field Length Counter 2 (8 bits)
FPE	Field Pointer E (8 bits)
FP1	Field Pointer 1 (8 bits)
FP2	Field Pointer 2 (8 bits)
FP3	Field Pointer 3 (8 bits)

GET	Pager GET Address Register
GRP	Group Register
HSDB	High Speed Data Buffer
IMASK	Interrupt Mask (bits 28-31 of PSW)
I/O	Input/Output
LP	Link Pointer (FP1 and FP2 linked)
LSB	Least Significant Bit (bit 31 of 32 bit word); Right Most Bit; Low Order Bit
LSF	Least-Significant Field
M	M-Response Store Register; Mask (256 bits)
MASK	M-Response Store Register
MDA	Multi-Dimensional Access Memory; Associative Array
MSB	Most Significant Bit (bit 0 of word); Left Most Bit; High Order Bit
MSF	Most-Significant Field
OR	Logical Inclusive OR
PAGE0	High Speed Solid State Memory; 512 32-Bit Words
PAGE1	High Speed Solid State Memory; 512 32-Bit Words
PAGE2	High Speed Solid State Memory; 512 32-Bit Words
PC	Program Counter (16 bits)
PC0	Program Counter (bits 0-7)
PC1	Program Counter (bits 8-15)
PI/O	Parallel Input/Output
PSW	Program Status Word
PUT	Pager PUT Address Register
RS	Response Store
R0	Branch and Link Register (Memory Location 600_{16})
R1	Branch and Link Register (Memory Location 601_{16})
R2	Branch and Link Register (Memory Location 602_{16})
R3	Branch and Link Register (Memory Location 603_{16})
R4	Branch and Link Register (Memory Location 604_{16})
R5	Branch and Link Register (Memory Location 605_{16})
R6	Branch and Link Register (Memory Location 606_{16})
R7	Branch and Link Register (Memory Location 607_{16})

SC	Sequential Controller
SLM	Start Loop Marker
SP	Sequential Processor
X	X-Response Store Register (256 bits)
XOR	Logical Exclusive OR
Y	Y-Response Store Register (256 bits)

APPENDIX D

HEXADECIMAL/DECIMAL TABLE

HEXADECIMAL-DECIMAL NUMBER CONVERSION TABLE

GENERAL

The table provides for direct conversion of hexadecimal and decimal numbers in these ranges:

Hexadecimal	Decimal
000 to FFF	0000 to 4095

HEXADECIMAL-DECIMAL NUMBER CONVERSION

In the table, the decimal value appears at the intersection of the row representing the most significant hexadecimal digits (16^2 and 16^1) and the column representing the least significant hexadecimal digit (16^0).

Example

$C21_{16} = 3105_{10}$

HEX	0	1	2
C0	3072	3073	3074
C1	3088	3089	3090
C2	3104	3105	3106
C3	3120	3121	3122

For numbers outside the range of the table, add the following values to the table figures:

Hexadecimal	Decimal	Hexadecimal	Decimal
1000	4,096	C000	49,152
2000	8,192	D000	53,248
3000	12,288	E000	57,344
4000	16,384	F000	61,440
5000	20,480	10000	65,536
6000	24,576	20000	131,072
7000	28,672	30000	196,608
8000	32,768	40000	262,144
9000	36,864	50000	327,680
A000	40,960	60000	393,216
B000	45,056	70000	458,752

Example

$1C21_{16} = 7201_{10}$

Hexadecimal	Decimal
C21	3105
+1000	4096
1C21	7201

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	0000	0001	0002	0003	0004	0005	0006	0007	0008	0009	0010	0011	0012	0013	0014	0015
01	0016	0017	0018	0019	0020	0021	0022	0023	0024	0025	0026	0027	0028	0029	0030	0031
02	0032	0033	0034	0035	0036	0037	0038	0039	0040	0041	0042	0043	0044	0045	0046	0047
03	0048	0049	0050	0051	0052	0053	0054	0055	0056	0057	0058	0059	0060	0061	0062	0063
04	0064	0065	0066	0067	0068	0069	0070	0071	0072	0073	0074	0075	0076	0077	0078	0079
05	0080	0081	0082	0083	0084	0085	0086	0087	0088	0089	0090	0091	0092	0093	0094	0095
06	0096	0097	0098	0099	0100	0101	0102	0103	0104	0105	0106	0107	0108	0109	0110	0111
07	0112	0113	0114	0115	0116	0117	0118	0119	0120	0121	0122	0123	0124	0125	0126	0127
08	0128	0129	0130	0131	0132	0133	0134	0135	0136	0137	0138	0139	0140	0141	0142	0143
09	0144	0145	0146	0147	0148	0149	0150	0151	0152	0153	0154	0155	0156	0157	0158	0159
0A	0160	0161	0162	0163	0164	0165	0166	0167	0168	0169	0170	0171	0172	0173	0174	0175
0B	0176	0177	0178	0179	0180	0181	0182	0183	0184	0185	0186	0187	0188	0189	0190	0191
0C	0192	0193	0194	0195	0196	0197	0198	0199	0200	0201	0202	0203	0204	0205	0206	0207
0D	0208	0209	0210	0211	0212	0213	0214	0215	0216	0217	0218	0219	0220	0221	0222	0223
0E	0224	0225	0226	0227	0228	0229	0230	0231	0232	0233	0234	0235	0236	0237	0238	0239
0F	0240	0241	0242	0243	0244	0245	0246	0247	0248	0249	0250	0251	0252	0253	0254	0255

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
10	0256	0257	0258	0259	0260	0261	0262	0263	0264	0265	0266	0267	0268	0269	0270	0271
11	0272	0273	0274	0275	0276	0277	0278	0279	0280	0281	0282	0283	0284	0285	0286	0287
12	0288	0289	0290	0291	0292	0293	0294	0295	0296	0297	0298	0299	0300	0301	0302	0303
13	0304	0305	0306	0307	0308	0309	0310	0311	0312	0313	0314	0315	0316	0317	0318	0319
14	0320	0321	0322	0323	0324	0325	0326	0327	0328	0329	0330	0331	0332	0333	0334	0335
15	0336	0337	0338	0339	0340	0341	0342	0343	0344	0345	0346	0347	0348	0349	0350	0351
16	0352	0353	0354	0355	0356	0357	0358	0359	0360	0361	0362	0363	0364	0365	0366	0367
17	0368	0369	0370	0371	0372	0373	0374	0375	0376	0377	0378	0379	0380	0381	0382	0383
18	0384	0385	0386	0387	0388	0389	0390	0391	0392	0393	0394	0395	0396	0397	0398	0399
19	0400	0401	0402	0403	0404	0405	0406	0407	0408	0409	0410	0411	0412	0413	0414	0415
1A	0416	0417	0418	0419	0420	0421	0422	0423	0424	0425	0426	0427	0428	0429	0430	0431
1B	0432	0433	0434	0435	0436	0437	0438	0439	0440	0441	0442	0443	0444	0445	0446	0447
1C	0448	0449	0450	0451	0452	0453	0454	0455	0456	0457	0458	0459	0460	0461	0462	0463
1D	0464	0465	0466	0467	0468	0469	0470	0471	0472	0473	0474	0475	0476	0477	0478	0479
1E	0480	0481	0482	0483	0484	0485	0486	0487	0488	0489	0490	0491	0492	0493	0494	0495
1F	0496	0497	0498	0499	0500	0501	0502	0503	0504	0505	0506	0507	0508	0509	0510	0511

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20	0512	0513	0514	0515	0516	0517	0518	0519	0520	0521	0522	0523	0524	0525	0526	0527
21	0528	0529	0530	0531	0532	0533	0534	0535	0536	0537	0538	0539	0540	0541	0542	0543
22	0544	0545	0546	0547	0548	0549	0550	0551	0552	0553	0554	0555	0556	0557	0558	0559
23	0560	0561	0562	0563	0564	0565	0566	0567	0568	0569	0570	0571	0572	0573	0574	0575
24	0576	0577	0578	0579	0580	0581	0582	0583	0584	0585	0586	0587	0588	0589	0590	0591
25	0592	0593	0594	0595	0596	0597	0598	0599	0600	0601	0602	0603	0604	0605	0606	0607
26	0608	0609	0610	0611	0612	0613	0614	0615	0616	0617	0618	0619	0620	0621	0622	0623
27	0624	0625	0626	0627	0628	0629	0630	0631	0632	0633	0634	0635	0636	0637	0638	0639
28	0640	0641	0642	0643	0644	0645	0646	0647	0648	0649	0650	0651	0652	0653	0654	0655
29	0656	0657	0658	0659	0660	0661	0662	0663	0664	0665	0666	0667	0668	0669	0670	0671
2A	0672	0673	0674	0675	0676	0677	0678	0679	0680	0681	0682	0683	0684	0685	0686	0687
2B	0688	0689	0690	0691	0692	0693	0694	0695	0696	0697	0698	0699	0700	0701	0702	0703
2C	0704	0705	0706	0707	0708	0709	0710	0711	0712	0713	0714	0715	0716	0717	0718	0719
2D	0720	0721	0722	0723	0724	0725	0726	0727	0728	0729	0730	0731	0732	0733	0734	0735
2E	0736	0737	0738	0739	0740	0741	0742	0743	0744	0745	0746	0747	0748	0749	0750	0751
2F	0752	0753	0754	0755	0756	0757	0758	0759	0760	0761	0762	0763	0764	0765	0766	0767

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
30	0768	0769	0770	0771	0772	0773	0774	0775	0776	0777	0778	0779	0780	0781	0782	0783
31	0784	0785	0786	0787	0788	0789	0790	0791	0792	0793	0794	0795	0796	0797	0798	0799
32	0800	0801	0802	0803	0804	0805	0806	0807	0808	0809	0810	0811	0812	0813	0814	0815
33	0816	0817	0818	0819	0820	0821	0822	0823	0824	0825	0826	0827	0828	0829	0830	0831
34	0832	0833	0834	0835	0836	0837	0838	0839	0840	0841	0842	0843	0844	0845	0846	0847
35	0848	0849	0850	0851	0852	0853	0854	0855	0856	0857	0858	0859	0860	0861	0862	0863
36	0864	0865	0866	0867	0868	0869	0870	0871	0872	0873	0874	0875	0876	0877	0878	0879
37	0880	0881	0882	0883	0884	0885	0886	0887	0888	0889	0890	0891	0892	0893	0894	0895
38	0896	0897	0898	0899	0900	0901	0902	0903	0904	0905	0906	0907	0908	0909	0910	0911
39	0912	0913	0914	0915	0916	0917	0918	0919	0920	0921	0922	0923	0924	0925	0926	0927
3A	0928	0929	0930	0931	0932	0933	0934	0935	0936	0937	0938	0939	0940	0941	0942	0943
3B	0944	0945	0946	0947	0948	0949	0950	0951	0952	0953	0954	0955	0956	0957	0958	0959
3C	0960	0961	0962	0963	0964	0965	0966	0967	0968	0969	0970	0971	0972	0973	0974	0975
3D	0976	0977	0978	0979	0980	0981	0982	0983	0984	0985	0986	0987	0988	0989	0990	0991
3E	0992	0993	0994	0995	0996	0997	0998	0999	1000	1001	1002	1003	1004	1005	1006	1007
3F	1008	1009	1010	1011	1012	1013	1014	1015	1016	1017	1018	1019	1020	1021	1022	1023

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
40	1024	1025	1026	1027	1028	1029	1030	1031	1032	1033	1034	1035	1036	1037	1038	1039
41	1040	1041	1042	1043	1044	1045	1046	1047	1048	1049	1050	1051	1052	1053	1054	1055
42	1056	1057	1058	1059	1060	1061	1062	1063	1064	1065	1066	1067	1068	1069	1070	1071
43	1072	1073	1074	1075	1076	1077	1078	1079	1080	1081	1082	1083	1084	1085	1086	1087
44	1088	1089	1090	1091	1092	1093	1094	1095	1096	1097	1098	1099	1100	1101	1102	1103
45	1104	1105	1106	1107	1108	1109	1110	1111	1112	1113	1114	1115	1116	1117	1118	1119
46	1120	1121	1122	1123	1124	1125	1126	1127	1128	1129	1130	1131	1132	1133	1134	1135
47	1136	1137	1138	1139	1140	1141	1142	1143	1144	1145	1146	1147	1148	1149	1150	1151
48	1152	1153	1154	1155	1156	1157	1158	1159	1160	1161	1162	1163	1164	1165	1166	1167
49	1168	1169	1170	1171	1172	1173	1174	1175	1176	1177	1178	1179	1180	1181	1182	1183
4A	1184	1185	1186	1187	1188	1189	1190	1191	1192	1193	1194	1195	1196	1197	1198	1199
4B	1200	1201	1202	1203	1204	1205	1206	1207	1208	1209	1210	1211	1212	1213	1214	1215
4C	1216	1217	1218	1219	1220	1221	1222	1223	1224	1225	1226	1227	1228	1229	1230	1231
4D	1232	1233	1234	1235	1236	1237	1238	1239	1240	1241	1242	1243	1244	1245	1246	1247
4E	1248	1249	1250	1251	1252	1253	1254	1255	1256	1257	1258	1259	1260	1261	1262	1263
4F	1264	1265	1266	1267	1268	1269	1270	1271	1272	1273	1274	1275	1276	1277	1278	1279

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
50	1280	1281	1282	1283	1284	1285	1286	1287	1288	1289	1290	1291	1292	1293	1294	1295
51	1296	1297	1298	1299	1300	1301	1302	1303	1304	1305	1306	1307	1308	1309	1310	1311
52	1312	1313	1314	1315	1316	1317	1318	1319	1320	1321	1322	1323	1324	1325	1326	1327
53	1328	1329	1330	1331	1332	1333	1334	1335	1336	1337	1338	1339	1340	1341	1342	1343
54	1344	1345	1346	1347	1348	1349	1350	1351	1352	1353	1354	1355	1356	1357	1358	1359
55	1360	1361	1362	1363	1364	1365	1366	1367	1368	1369	1370	1371	1372	1373	1374	1375
56	1376	1377	1378	1379	1380	1381	1382	1383	1384	1385	1386	1387	1388	1389	1390	1391
57	1392	1393	1394	1395	1396	1397	1398	1399	1400	1401	1402	1403	1404	1405	1406	1407
58	1408	1409	1410	1411	1412	1413	1414	1415	1416	1417	1418	1419	1420	1421	1422	1423
59	1424	1425	1426	1427	1428	1429	1430	1431	1432	1433	1434	1435	1436	1437	1438	1439
5A	1440	1441	1442	1443	1444	1445	1446	1447	1448	1449	1450	1451	1452	1453	1454	1455
5B	1456	1457	1458	1459	1460	1461	1462	1463	1464	1465	1466	1467	1468	1469	1470	1471
5C	1472	1473	1474	1475	1476	1477	1478	1479	1480	1481	1482	1483	1484	1485	1486	1487
5D	1488	1489	1490	1491	1492	1493	1494	1495	1496	1497	1498	1499	1500	1501	1502	1503
5E	1504	1505	1506	1507	1508	1509	1510	1511	1512	1513	1514	1515	1516	1517	1518	1519
5F	1520	1521	1522	1523	1524	1525	1526	1527	1528	1529	1530	1531	1532	1533	1534	1535

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
60	1536	1537	1538	1539	1540	1541	1542	1543	1544	1545	1546	1547	1548	1549	1550	1551
61	1552	1553	1554	1555	1556	1557	1558	1559	1560	1561	1562	1563	1564	1565	1566	1567
62	1568	1569	1570	1571	1572	1573	1574	1575	1576	1577	1578	1579	1580	1581	1582	1583
63	1584	1585	1586	1587	1588	1589	1590	1591	1592	1593	1594	1595	1596	1597	1598	1599
64	1600	1601	1602	1603	1604	1605	1606	1607	1608	1609	1610	1611	1612	1613	1614	1615
65	1616	1617	1618	1619	1620	1621	1622	1623	1624	1625	1626	1627	1628	1629	1630	1631
66	1632	1633	1634	1635	1636	1637	1638	1639	1640	1641	1642	1643	1644	1645	1646	1647
67	1648	1649	1650	1651	1652	1653	1654	1655	1656	1657	1658	1659	1660	1661	1662	1663
68	1664	1665	1666	1667	1668	1669	1670	1671	1672	1673	1674	1675	1676	1677	1678	1679
69	1680	1681	1682	1683	1684	1685	1686	1687	1688	1689	1690	1691	1692	1693	1694	1695
6A	1696	1697	1698	1699	1700	1701	1702	1703	1704	1705	1706	1707	1708	1709	1710	1711
6B	1712	1713	1714	1715	1716	1717	1718	1719	1720	1721	1722	1723	1724	1725	1726	1727
6C	1728	1729	1730	1731	1732	1733	1734	1735	1736	1737	1738	1739	1740	1741	1742	1743
6D	1744	1745	1746	1747	1748	1749	1750	1751	1752	1753	1754	1755	1756	1757	1758	1759
6E	1760	1761	1762	1763	1764	1765	1766	1767	1768	1769	1770	1771	1772	1773	1774	1775
6F	1776	1777	1778	1779	1780	1781	1782	1783	1784	1785	1786	1787	1788	1789	1790	1791

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
70	1792	1793	1794	1795	1796	1797	1798	1799	1800	1801	1802	1803	1804	1805	1806	1807
71	1808	1809	1810	1811	1812	1813	1814	1815	1816	1817	1818	1819	1820	1821	1822	1823
72	1824	1825	1826	1827	1828	1829	1830	1831	1832	1833	1834	1835	1836	1837	1838	1839
73	1840	1841	1842	1843	1844	1845	1846	1847	1848	1849	1850	1851	1852	1853	1854	1855
74	1856	1857	1858	1859	1860	1861	1862	1863	1864	1865	1866	1867	1868	1869	1870	1871
75	1872	1873	1874	1875	1876	1877	1878	1879	1880	1881	1882	1883	1884	1885	1886	1887
76	1888	1889	1890	1891	1892	1893	1894	1895	1896	1897	1898	1899	1900	1901	1902	1903
77	1904	1905	1906	1907	1908	1909	1910	1911	1912	1913	1914	1915	1916	1917	1918	1919
78	1920	1921	1922	1923	1924	1925	1926	1927	1928	1929	1930	1931	1932	1933	1934	1935
79	1936	1937	1938	1939	1940	1941	1942	1943	1944	1945	1946	1947	1948	1949	1950	1951
7A	1952	1953	1954	1955	1956	1957	1958	1959	1960	1961	1962	1963	1964	1965	1966	1967
7B	1968	1969	1970	1971	1972	1973	1974	1975	1976	1977	1978	1979	1980	1981	1982	1983
7C	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999
7D	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015
7E	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031
7F	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044	2045	2046	2047

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
80	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063
81	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079
82	2080	2081	2082	2083	2084	2085	2086	2087	2088	2089	2090	2091	2092	2093	2094	2095
83	2096	2097	2098	2099	2100	2101	2102	2103	2104	2105	2106	2107	2108	2109	2110	2111
84	2112	2113	2114	2115	2116	2117	2118	2119	2120	2121	2122	2123	2124	2125	2126	2127
85	2128	2129	2130	2131	2132	2133	2134	2135	2136	2137	2138	2139	2140	2141	2142	2143
86	2144	2145	2146	2147	2148	2149	2150	2151	2152	2153	2154	2155	2156	2157	2158	2159
87	2160	2161	2162	2163	2164	2165	2166	2167	2168	2169	2170	2171	2172	2173	2174	2175
88	2176	2177	2178	2179	2180	2181	2182	2183	2184	2185	2186	2187	2188	2189	2190	2191
89	2192	2193	2194	2195	2196	2197	2198	2199	2200	2201	2202	2203	2204	2205	2206	2207
8A	2208	2209	2210	2211	2212	2213	2214	2215	2216	2217	2218	2219	2220	2221	2222	2223
8B	2224	2225	2226	2227	2228	2229	2230	2231	2232	2233	2234	2235	2236	2237	2238	2239
8C	2240	2241	2242	2243	2244	2245	2246	2247	2248	2249	2250	2251	2252	2253	2254	2255
8D	2256	2257	2258	2259	2260	2261	2262	2263	2264	2265	2266	2267	2268	2269	2270	2271
8E	2272	2273	2274	2275	2276	2277	2278	2279	2280	2281	2282	2283	2284	2285	2286	2287
8F	2288	2289	2290	2291	2292	2293	2294	2295	2296	2297	2298	2299	2300	2301	2302	2303

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
90	2304	2305	2306	2307	2308	2309	2310	2311	2312	2313	2314	2315	2316	2317	2318	2319
91	2320	2321	2322	2323	2324	2325	2326	2327	2328	2329	2330	2331	2332	2333	2334	2335
92	2336	2337	2338	2339	2340	2341	2342	2343	2344	2345	2346	2347	2348	2349	2350	2351
93	2352	2353	2354	2355	2356	2357	2358	2359	2360	2361	2362	2363	2364	2365	2366	2367
94	2368	2369	2370	2371	2372	2373	2374	2375	2376	2377	2378	2379	2380	2381	2382	2383
95	2384	2385	2386	2387	2388	2389	2390	2391	2392	2393	2394	2395	2396	2397	2398	2399
96	2400	2401	2402	2403	2404	2405	2406	2407	2408	2409	2410	2411	2412	2413	2414	2415
97	2416	2417	2418	2419	2420	2421	2422	2423	2424	2425	2426	2427	2428	2429	2430	2431
98	2432	2433	2434	2435	2436	2437	2438	2439	2440	2441	2442	2443	2444	2445	2446	2447
99	2448	2449	2450	2451	2452	2453	2454	2455	2456	2457	2458	2459	2460	2461	2462	2463
9A	2464	2465	2466	2467	2468	2469	2470	2471	2472	2473	2474	2475	2476	2477	2478	2479
9B	2480	2481	2482	2483	2484	2485	2486	2487	2488	2489	2490	2491	2492	2493	2494	2495
9C	2496	2497	2498	2499	2500	2501	2502	2503	2504	2505	2506	2507	2508	2509	2510	2511
9D	2512	2513	2514	2515	2516	2517	2518	2519	2520	2521	2522	2523	2524	2525	2526	2527
9E	2528	2529	2530	2531	2532	2533	2534	2535	2536	2537	2538	2539	2540	2541	2542	2543
9F	2544	2545	2546	2547	2548	2549	2550	2551	2552	2553	2554	2555	2556	2557	2558	2559

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A0	2560	2561	2562	2563	2564	2565	2566	2567	2568	2569	2570	2571	2572	2573	2574	2575
A1	2576	2577	2578	2579	2580	2581	2582	2583	2584	2585	2586	2587	2588	2589	2590	2591
A2	2592	2593	2594	2595	2596	2597	2598	2599	2600	2601	2602	2603	2604	2605	2606	2607
A3	2608	2609	2610	2611	2612	2613	2614	2615	2616	2617	2618	2619	2620	2621	2622	2623
A4	2624	2625	2626	2627	2628	2629	2630	2631	2632	2633	2634	2635	2636	2637	2638	2639
A5	2640	2641	2642	2643	2644	2645	2646	2647	2648	2649	2650	2651	2652	2653	2654	2655
A6	2656	2657	2658	2659	2660	2661	2662	2663	2664	2665	2666	2667	2668	2669	2670	2671
A7	2672	2673	2674	2675	2676	2677	2678	2679	2680	2681	2682	2683	2684	2685	2686	2687
A8	2688	2689	2690	2691	2692	2693	2694	2695	2696	2697	2698	2699	2700	2701	2702	2703
A9	2704	2705	2706	2707	2708	2709	2710	2711	2712	2713	2714	2715	2716	2717	2718	2719
AA	2720	2721	2722	2723	2724	2725	2726	2727	2728	2729	2730	2731	2732	2733	2734	2735
AB	2736	2737	2738	2739	2740	2741	2742	2743	2744	2745	2746	2747	2748	2749	2750	2751
AC	2752	2753	2754	2755	2756	2757	2758	2759	2760	2761	2762	2763	2764	2765	2766	2767
AD	2768	2769	2770	2771	2772	2773	2774	2775	2776	2777	2778	2779	2780	2781	2782	2783
AE	2784	2785	2786	2787	2788	2789	2790	2791	2792	2793	2794	2795	2796	2797	2798	2799
AF	2800	2801	2802	2803	2804	2805	2806	2807	2808	2809	2810	2811	2812	2813	2814	2815

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
B0	2816	2817	2818	2819	2820	2821	2822	2823	2824	2825	2826	2827	2828	2829	2830	2831
B1	2832	2833	2834	2835	2836	2837	2838	2839	2840	2841	2842	2843	2844	2845	2846	2847
B2	2848	2849	2850	2851	2852	2853	2854	2855	2856	2857	2858	2859	2860	2861	2862	2863
B3	2864	2865	2866	2867	2868	2869	2870	2871	2872	2873	2874	2875	2876	2877	2878	2879
B4	2880	2881	2882	2883	2884	2885	2886	2887	2888	2889	2890	2891	2892	2893	2894	2895
B5	2896	2897	2898	2899	2900	2901	2902	2903	2904	2905	2906	2907	2908	2909	2910	2911
B6	2912	2913	2914	2915	2916	2917	2918	2919	2920	2921	2922	2923	2924	2925	2926	2927
B7	2928	2929	2930	2931	2932	2933	2934	2935	2936	2937	2938	2939	2940	2941	2942	2943
B8	2944	2945	2946	2947	2948	2949	2950	2951	2952	2953	2954	2955	2956	2957	2958	2959
B9	2960	2961	2962	2963	2964	2965	2966	2967	2968	2969	2970	2971	2972	2973	2974	2975
BA	2976	2977	2978	2979	2980	2981	2982	2983	2984	2985	2986	2987	2988	2989	2990	2991
BB	2992	2993	2994	2995	2996	2997	2998	2999	3000	3001	3002	3003	3004	3005	3006	3007
BC	3008	3009	3010	3011	3012	3013	3014	3015	3016	3017	3018	3019	3020	3021	3022	3023
BD	3024	3025	3026	3027	3028	3029	3030	3031	3032	3033	3034	3035	3036	3037	3038	3039
BE	3040	3041	3042	3043	3044	3045	3046	3047	3048	3049	3050	3051	3052	3053	3054	3055
BF	3056	3057	3058	3059	3060	3061	3062	3063	3064	3065	3066	3067	3068	3069	3070	3071

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
C0	3072	3073	3074	3075	3076	3077	3078	3079	3080	3081	3082	3083	3084	3085	3086	3087
C1	3088	3089	3090	3091	3092	3093	3094	3095	3096	3097	3098	3099	3100	3101	3102	3103
C2	3104	3105	3106	3107	3108	3109	3110	3111	3112	3113	3114	3115	3116	3117	3118	3119
C3	3120	3121	3122	3123	3124	3125	3126	3127	3128	3129	3130	3131	3132	3133	3134	3135
C4	3136	3137	3138	3139	3140	3141	3142	3143	3144	3145	3146	3147	3148	3149	3150	3151
C5	3152	3153	3154	3155	3156	3157	3158	3159	3160	3161	3162	3163	3164	3165	3166	3167
C6	3168	3169	3170	3171	3172	3173	3174	3175	3176	3177	3178	3179	3180	3181	3182	3183
C7	3184	3185	3186	3187	3188	3189	3190	3191	3192	3193	3194	3195	3196	3197	3198	3199
C8	3200	3201	3202	3203	3204	3205	3206	3207	3208	3209	3210	3211	3212	3213	3214	3215
C9	3216	3217	3218	3219	3220	3221	3222	3223	3224	3225	3226	3227	3228	3229	3230	3231
CA	3232	3233	3234	3235	3236	3237	3238	3239	3240	3241	3242	3243	3244	3245	3246	3247
CB	3248	3249	3250	3251	3252	3253	3254	3255	3256	3257	3258	3259	3260	3261	3262	3263
CC	3264	3265	3266	3267	3268	3269	3270	3271	3272	3273	3274	3275	3276	3277	3278	3279
CD	3280	3281	3282	3283	3284	3285	3286	3287	3288	3289	3290	3291	3292	3293	3294	3295
CE	3296	3297	3298	3299	3300	3301	3302	3303	3304	3305	3306	3307	3308	3309	3310	3311
CF	3312	3313	3314	3315	3316	3317	3318	3319	3320	3321	3322	3323	3324	3325	3326	3327

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
D0	3328	3329	3330	3331	3332	3333	3334	3335	3336	3337	3338	3339	3340	3341	3342	3343
D1	3344	3345	3346	3347	3348	3349	3350	3351	3352	3353	3354	3355	3356	3357	3358	3359
D2	3360	3361	3362	3363	3364	3365	3366	3367	3368	3369	3370	3371	3372	3373	3374	3375
D3	3376	3377	3378	3379	3380	3381	3382	3383	3384	3385	3386	3387	3388	3389	3390	3391
D4	3392	3393	3394	3395	3396	3397	3398	3399	3400	3401	3402	3403	3404	3405	3406	3407
D5	3408	3409	3410	3411	3412	3413	3414	3415	3416	3417	3418	3419	3420	3421	3422	3423
D6	3424	3425	3426	3427	3428	3429	3430	3431	3432	3433	3434	3435	3436	3437	3438	3439
D7	3440	3441	3442	3443	3444	3445	3446	3447	3448	3449	3450	3451	3452	3453	3454	3455
D8	3456	3457	3458	3459	3460	3461	3462	3463	3464	3465	3466	3467	3468	3469	3470	3471
D9	3472	3473	3474	3475	3476	3477	3478	3479	3480	3481	3482	3483	3484	3485	3486	3487
DA	3488	3489	3490	3491	3492	3493	3494	3495	3496	3497	3498	3499	3500	3501	3502	3503
DB	3504	3505	3506	3507	3508	3509	3510	3511	3512	3513	3514	3515	3516	3517	3518	3519
DC	3520	3521	3522	3523	3524	3525	3526	3527	3528	3529	3530	3531	3532	3533	3534	3535
DD	3536	3537	3538	3539	3540	3541	3542	3543	3544	3545	3546	3547	3548	3549	3550	3551
DE	3552	3553	3554	3555	3556	3557	3558	3559	3560	3561	3562	3563	3564	3565	3566	3567
DF	3568	3569	3570	3571	3572	3573	3574	3575	3576	3577	3578	3579	3580	3581	3582	3583

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
E0	3584	3585	3586	3587	3588	3589	3590	3591	3592	3593	3594	3595	3596	3597	3598	3599
E1	3600	3601	3602	3603	3604	3605	3606	3607	3608	3609	3610	3611	3612	3613	3614	3615
E2	3616	3617	3618	3619	3620	3621	3622	3623	3624	3625	3626	3627	3628	3629	3630	3631
E3	3632	3633	3634	3635	3636	3637	3638	3639	3640	3641	3642	3643	3644	3645	3646	3647
E4	3648	3649	3650	3651	3652	3653	3654	3655	3656	3657	3658	3659	3660	3661	3662	3663
E5	3664	3665	3666	3667	3668	3669	3670	3671	3672	3673	3674	3675	3676	3677	3678	3679
E6	3680	3681	3682	3683	3684	3685	3686	3687	3688	3689	3690	3691	3692	3693	3694	3695
E7	3696	3697	3698	3699	3700	3701	3702	3703	3704	3705	3706	3707	3708	3709	3710	3711
E8	3712	3713	3714	3715	3716	3717	3718	3719	3720	3721	3722	3723	3724	3725	3726	3727
E9	3728	3729	3730	3731	3732	3733	3734	3735	3736	3737	3738	3739	3740	3741	3742	3743
EA	3744	3745	3746	3747	3748	3749	3750	3751	3752	3753	3754	3755	3756	3757	3758	3759
EB	3760	3761	3762	3763	3764	3765	3766	3767	3768	3769	3770	3771	3772	3773	3774	3775
EC	3776	3777	3778	3779	3780	3781	3782	3783	3784	3785	3786	3787	3788	3789	3790	3791
ED	3792	3793	3794	3795	3796	3797	3798	3799	3800	3801	3802	3803	3804	3805	3806	3807
EE	3808	3809	3810	3811	3812	3813	3814	3815	3816	3817	3818	3819	3820	3821	3822	3823
EF	3824	3825	3826	3827	3828	3829	3830	3831	3832	3833	3834	3835	3836	3837	3838	3839

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
F0	3840	3841	3842	3843	3844	3845	3846	3847	3848	3849	3850	3851	3852	3853	3854	3855
F1	3856	3857	3858	3859	3860	3861	3862	3863	3864	3865	3866	3867	3868	3869	3870	3871
F2	3872	3873	3874	3875	3876	3877	3878	3879	3880	3881	3882	3883	3884	3885	3886	3887
F3	3888	3889	3890	3891	3892	3893	3894	3895	3896	3897	3898	3899	3900	3901	3902	3903
F4	3904	3905	3906	3907	3908	3909	3910	3911	3912	3913	3914	3915	3916	3917	3918	3919
F5	3920	3921	3922	3923	3924	3925	3926	3927	3928	3929	3930	3931	3932	3933	3934	3935
F6	3936	3937	3938	3939	3940	3941	3942	3943	3944	3945	3946	3947	3948	3949	3950	3951
F7	3952	3953	3954	3955	3956	3957	3958	3959	3960	3961	3962	3963	3964	3965	3966	3967
F8	3968	3969	3970	3971	3972	3973	3974	3975	3976	3977	3978	3979	3980	3981	3982	3983
F9	3984	3985	3986	3987	3988	3989	3990	3991	3992	3993	3994	3995	3996	3997	3998	3999
FA	4000	4001	4002	4003	4004	4005	4006	4007	4008	4009	4010	4011	4012	4013	4014	4015
FB	4016	4017	4018	4019	4020	4021	4022	4023	4024	4025	4026	4027	4028	4029	4030	4031
FC	4032	4033	4034	4035	4036	4037	4038	4039	4040	4041	4042	4043	4044	4045	4046	4047
FD	4048	4049	4050	4051	4052	4053	4054	4055	4056	4057	4058	4059	4060	4061	4062	4063
FE	4064	4065	4066	4067	4068	4069	4070	4071	4072	4073	4074	4075	4076	4077	4078	4079
FF	4080	4081	4082	4083	4084	4085	4086	4087	4088	4089	4090	4091	4092	4093	4094	4095

0000 to 0777 (Octal) | 0000 to 0511 (Decimal)

Octal	Decimal
10000	4096
20000	8192
30000	12288
40000	16384
50000	20480
60000	24576
70000	28672

	0	1	2	3	4	5	6	7
0000	0000	0001	0002	0003	0004	0005	0006	0007
0010	0008	0009	0010	0011	0012	0013	0014	0015
0020	0016	0017	0018	0019	0020	0021	0022	0023
0030	0024	0025	0026	0027	0028	0029	0030	0031
0040	0032	0033	0034	0035	0036	0037	0038	0039
0050	0040	0041	0042	0043	0044	0045	0046	0047
0060	0048	0049	0050	0051	0052	0053	0054	0055
0070	0056	0057	0058	0059	0060	0061	0062	0063
0100	0064	0065	0066	0067	0068	0069	0070	0071
0110	0072	0073	0074	0075	0076	0077	0078	0079
0120	0080	0081	0082	0083	0084	0085	0086	0087
0130	0088	0089	0090	0091	0092	0093	0094	0095
0140	0096	0097	0098	0099	0100	0101	0102	0103
0150	0104	0105	0106	0107	0108	0109	0110	0111
0160	0112	0113	0114	0115	0116	0117	0118	0119
0170	0120	0121	0122	0123	0124	0125	0126	0127
0200	0128	0129	0130	0131	0132	0133	0134	0135
0210	0136	0137	0138	0139	0140	0141	0142	0143
0220	0144	0145	0146	0147	0148	0149	0150	0151
0230	0152	0153	0154	0155	0156	0157	0158	0159
0240	0160	0161	0162	0163	0164	0165	0166	0167
0250	0168	0169	0170	0171	0172	0173	0174	0175
0260	0176	0177	0178	0179	0180	0181	0182	0183
0270	0184	0185	0186	0187	0188	0189	0190	0191
0300	0192	0193	0194	0195	0196	0197	0198	0199
0310	0200	0201	0202	0203	0204	0205	0206	0207
0320	0208	0209	0210	0211	0212	0213	0214	0215
0330	0216	0217	0218	0219	0220	0221	0222	0223
0340	0224	0225	0226	0227	0228	0229	0230	0231
0350	0232	0233	0234	0235	0236	0237	0238	0239
0360	0240	0241	0242	0243	0244	0245	0246	0247
0370	0248	0249	0250	0251	0252	0253	0254	0255

	0	1	2	3	4	5	6	7
0400	0256	0257	0258	0259	0260	0261	0262	0263
0410	0264	0265	0266	0267	0268	0269	0270	0271
0420	0272	0273	0274	0275	0276	0277	0278	0279
0430	0280	0281	0282	0283	0284	0285	0286	0287
0440	0288	0289	0290	0291	0292	0293	0294	0295
0450	0296	0297	0298	0299	0300	0301	0302	0303
0460	0304	0305	0306	0307	0308	0309	0310	0311
0470	0312	0313	0314	0315	0316	0317	0318	0319
0500	0320	0321	0322	0323	0324	0325	0326	0327
0510	0328	0329	0330	0331	0332	0333	0334	0335
0520	0336	0337	0338	0339	0340	0341	0342	0343
0530	0344	0345	0346	0347	0348	0349	0350	0351
0540	0352	0353	0354	0355	0356	0357	0358	0359
0550	0360	0361	0362	0363	0364	0365	0366	0367
0560	0368	0369	0370	0371	0372	0373	0374	0375
0570	0376	0377	0378	0379	0380	0381	0382	0383
0600	0384	0385	0386	0387	0388	0389	0390	0391
0610	0392	0393	0394	0395	0396	0397	0398	0399
0620	0400	0401	0402	0403	0404	0405	0406	0407
0630	0408	0409	0410	0411	0412	0413	0414	0415
0640	0416	0417	0418	0419	0420	0421	0422	0423
0650	0424	0425	0426	0427	0428	0429	0430	0431
0660	0432	0433	0434	0435	0436	0437	0438	0439
0670	0440	0441	0442	0443	0444	0445	0446	0447
0700	0448	0449	0450	0451	0452	0453	0454	0455
0710	0456	0457	0458	0459	0460	0461	0462	0463
0720	0464	0465	0466	0467	0468	0469	0470	0471
0730	0472	0473	0474	0475	0476	0477	0478	0479
0740	0480	0481	0482	0483	0484	0485	0486	0487
0750	0488	0489	0490	0491	0492	0493	0494	0495
0760	0496	0497	0498	0499	0500	0501	0502	0503
0770	0504	0505	0506	0507	0508	0509	0510	0511

1000 to 1777 (Octal) | 0512 to 1023 (Decimal)

	0	1	2	3	4	5	6	7
1000	0512	0513	0514	0515	0516	0517	0518	0519
1010	0520	0521	0522	0523	0524	0525	0526	0527
1020	0528	0529	0530	0531	0532	0533	0534	0535
1030	0536	0537	0538	0539	0540	0541	0542	0543
1040	0544	0545	0546	0547	0548	0549	0550	0551
1050	0552	0553	0554	0555	0556	0557	0558	0559
1060	0560	0561	0562	0563	0564	0565	0566	0567
1070	0568	0569	0570	0571	0572	0573	0574	0575
1100	0576	0577	0578	0579	0580	0581	0582	0583
1110	0584	0585	0586	0587	0588	0589	0590	0591
1120	0592	0593	0594	0595	0596	0597	0598	0599
1130	0600	0601	0602	0603	0604	0605	0606	0607
1140	0608	0609	0610	0611	0612	0613	0614	0615
1150	0616	0617	0618	0619	0620	0621	0622	0623
1160	0624	0625	0626	0627	0628	0629	0630	0631
1170	0632	0633	0634	0635	0636	0637	0638	0639
1200	0640	0641	0642	0643	0644	0645	0646	0647
1210	0648	0649	0650	0651	0652	0653	0654	0655
1220	0656	0657	0658	0659	0660	0661	0662	0663
1230	0664	0665	0666	0667	0668	0669	0670	0671
1240	0672	0673	0674	0675	0676	0677	0678	0679
1250	0680	0681	0682	0683	0684	0685	0686	0687
1260	0688	0689	0690	0691	0692	0693	0694	0695
1270	0696	0697	0698	0699	0700	0701	0702	0703
1300	0704	0705	0706	0707	0708	0709	0710	0711
1310	0712	0713	0714	0715	0716	0717	0718	0719
1320	0720	0721	0722	0723	0724	0725	0726	0727
1330	0728	0729	0730	0731	0732	0733	0734	0735
1340	0736	0737	0738	0739	0740	0741	0742	0743
1350	0744	0745	0746	0747	0748	0749	0750	0751
1360	0752	0753	0754	0755	0756	0757	0758	0759
1370	0760	0761	0762	0763	0764	0765	0766	0767

	0	1	2	3	4	5	6	7
1400	0768	0769	0770	0771	0772	0773	0774	0775
1410	0776	0777	0778	0779	0780	0781	0782	0783
1420	0784	0785	0786	0787	0788	0789	0790	0791
1430	0792	0793	0794	0795	0796	0797	0798	0799
1440	0800	0801	0802	0803	0804	0805	0806	0807
1450	0808	0809	0810	0811	0812	0813	0814	0815
1460	0816	0817	0818	0819	0820	0821	0822	0823
1470	0824	0825	0826	0827	0828	0829	0830	0831
1500	0832	0833	0834	0835	0836	0837	0838	0839
1510	0840	0841	0842	0843	0844	0845	0846	0847
1520	0848	0849	0850	0851	0852	0853	0854	0855
1530	0856	0857	0858	0859	0860	0861	0862	0863
1540	0864	0865	0866	0867	0868	0869	0870	0871
1550	0872	0873	0874	0875	0876	0877	0878	0879
1560	0880	0881	0882	0883	0884	0885	0886	0887
1570	0888	0889	0890	0891	0892	0893	0894	0895
1600	0896	0897	0898	0899	0900	0901	0902	0903
1610	0904	0905	0906	0907	0908	0909	0910	0911
1620	0912	0913	0914	0915	0916	0917	0918	0919
1630	0920	0921	0922	0923	0924	0925	0926	0927
1640	0928	0929	0930	0931	0932	0933	0934	0935
1650	0936	0937	0938	0939	0940	0941	0942	0943
1660	0944	0945	0946	0947	0948	0949	0950	0951
1670	0952	0953	0954	0955	0956	0957	0958	0959
1700	0960	0961	0962	0963	0964	0965	0966	0967
1710	0968	0969	0970	0971	0972	0973	0974	0975
1720	0976	0977	0978	0979	0980	0981	0982	0983
1730	0984	0985	0986	0987	0988	0989	0990	0991
1740	0992	0993	0994	0995	0996	0997	0998	0999
1750	1000	1001	1002	1003	1004	1005	1006	1007
1760	1008	1009	1010	1011	1012	1013	1014	1015
1770	1016	1017	1018	1019	1020	1021	1022	1023

Octal	Decimal
2000 to 2777 (Octal)	1024 to 1535 (Decimal)

Octal	Decimal
10000 -	4096
20000 -	8192
30000 -	12288
40000 -	16384
50000 -	20480
60000 -	24576
70000 -	28672

	0	1	2	3	4	5	6	7
2000	1024	1025	1026	1027	1028	1029	1030	1031
2010	1032	1033	1034	1035	1036	1037	1038	1039
2020	1040	1041	1042	1043	1044	1045	1046	1047
2030	1048	1049	1050	1051	1052	1053	1054	1055
2040	1056	1057	1058	1059	1060	1061	1062	1063
2050	1064	1065	1066	1067	1068	1069	1070	1071
2060	1072	1073	1074	1075	1076	1077	1078	1079
2070	1080	1081	1082	1083	1084	1085	1086	1087
2100	1088	1089	1090	1091	1092	1093	1094	1095
2110	1096	1097	1098	1099	1100	1101	1102	1103
2120	1104	1105	1106	1107	1108	1109	1110	1111
2130	1112	1113	1114	1115	1116	1117	1118	1119
2140	1120	1121	1122	1123	1124	1125	1126	1127
2150	1128	1129	1130	1131	1132	1133	1134	1135
2160	1136	1137	1138	1139	1140	1141	1142	1143
2170	1144	1145	1146	1147	1148	1149	1150	1151
2200	1152	1153	1154	1155	1156	1157	1158	1159
2210	1160	1161	1162	1163	1164	1165	1166	1167
2220	1168	1169	1170	1171	1172	1173	1174	1175
2230	1176	1177	1178	1179	1180	1181	1182	1183
2240	1184	1185	1186	1187	1188	1189	1190	1191
2250	1192	1193	1194	1195	1196	1197	1198	1199
2260	1200	1201	1202	1203	1204	1205	1206	1207
2270	1208	1209	1210	1211	1212	1213	1214	1215
2300	1216	1217	1218	1219	1220	1221	1222	1223
2310	1224	1225	1226	1227	1228	1229	1230	1231
2320	1232	1233	1234	1235	1236	1237	1238	1239
2330	1240	1241	1242	1243	1244	1245	1246	1247
2340	1248	1249	1250	1251	1252	1253	1254	1255
2350	1256	1257	1258	1259	1260	1261	1262	1263
2360	1264	1265	1266	1267	1268	1269	1270	1271
2370	1272	1273	1274	1275	1276	1277	1278	1279

	0	1	2	3	4	5	6	7
2400	1280	1281	1282	1283	1284	1285	1286	1287
2410	1288	1289	1290	1291	1292	1293	1294	1295
2420	1296	1297	1298	1299	1300	1301	1302	1303
2430	1304	1305	1306	1307	1308	1309	1310	1311
2440	1312	1313	1314	1315	1316	1317	1318	1319
2450	1320	1321	1322	1323	1324	1325	1326	1327
2460	1328	1329	1330	1331	1332	1333	1334	1335
2470	1336	1337	1338	1339	1340	1341	1342	1343
2500	1344	1345	1346	1347	1348	1349	1350	1351
2510	1352	1353	1354	1355	1356	1357	1358	1359
2520	1360	1361	1362	1363	1364	1365	1366	1367
2530	1368	1369	1370	1371	1372	1373	1374	1375
2540	1376	1377	1378	1379	1380	1381	1382	1383
2550	1384	1385	1386	1387	1388	1389	1390	1391
2560	1392	1393	1394	1395	1396	1397	1398	1399
2570	1400	1401	1402	1403	1404	1405	1406	1407
2600	1408	1409	1410	1411	1412	1413	1414	1415
2610	1416	1417	1418	1419	1420	1421	1422	1423
2620	1424	1425	1426	1427	1428	1429	1430	1431
2630	1432	1433	1434	1435	1436	1437	1438	1439
2640	1440	1441	1442	1443	1444	1445	1446	1447
2650	1448	1449	1450	1451	1452	1453	1454	1455
2660	1456	1457	1458	1459	1460	1461	1462	1463
2670	1464	1465	1466	1467	1468	1469	1470	1471
2700	1472	1473	1474	1475	1476	1477	1478	1479
2710	1480	1481	1482	1483	1484	1485	1486	1487
2720	1488	1489	1490	1491	1492	1493	1494	1495
2730	1496	1497	1498	1499	1500	1501	1502	1503
2740	1504	1505	1506	1507	1508	1509	1510	1511
2750	1512	1513	1514	1515	1516	1517	1518	1519
2760	1520	1521	1522	1523	1524	1525	1526	1527
2770	1528	1529	1530	1531	1532	1533	1534	1535

Octal	Decimal
3000 to 3777 (Octal)	1536 to 2047 (Decimal)

	0	1	2	3	4	5	6	7
3000	1536	1537	1538	1539	1540	1541	1542	1543
3010	1544	1545	1546	1547	1548	1549	1550	1551
3020	1552	1553	1554	1555	1556	1557	1558	1559
3030	1560	1561	1562	1563	1564	1565	1566	1567
3040	1568	1569	1570	1571	1572	1573	1574	1575
3050	1576	1577	1578	1579	1580	1581	1582	1583
3060	1584	1585	1586	1587	1588	1589	1590	1591
3070	1592	1593	1594	1595	1596	1597	1598	1599
3100	1600	1601	1602	1603	1604	1605	1606	1607
3110	1608	1609	1610	1611	1612	1613	1614	1615
3120	1616	1617	1618	1619	1620	1621	1622	1623
3130	1624	1625	1626	1627	1628	1629	1630	1631
3140	1632	1633	1634	1635	1636	1637	1638	1639
3150	1640	1641	1642	1643	1644	1645	1646	1647
3160	1648	1649	1650	1651	1652	1653	1654	1655
3170	1656	1657	1658	1659	1660	1661	1662	1663
3200	1664	1665	1666	1667	1668	1669	1670	1671
3210	1672	1673	1674	1675	1676	1677	1678	1679
3220	1680	1681	1682	1683	1684	1685	1686	1687
3230	1688	1689	1690	1691	1692	1693	1694	1695
3240	1696	1697	1698	1699	1700	1701	1702	1703
3250	1704	1705	1706	1707	1708	1709	1710	1711
3260	1712	1713	1714	1715	1716	1717	1718	1719
3270	1720	1721	1722	1723	1724	1725	1726	1727
3300	1728	1729	1730	1731	1732	1733	1734	1735
3310	1736	1737	1738	1739	1740	1741	1742	1743
3320	1744	1745	1746	1747	1748	1749	1750	1751
3330	1752	1753	1754	1755	1756	1757	1758	1759
3340	1760	1761	1762	1763	1764	1765	1766	1767
3350	1768	1769	1770	1771	1772	1773	1774	1775
3360	1776	1777	1778	1779	1780	1781	1782	1783
3370	1784	1785	1786	1787	1788	1789	1790	1791

	0	1	2	3	4	5	6	7
3400	1792	1793	1794	1795	1796	1797	1798	1799
3410	1800	1801	1802	1803	1804	1805	1806	1807
3420	1808	1809	1810	1811	1812	1813	1814	1815
3430	1816	1817	1818	1819	1820	1821	1822	1823
3440	1824	1825	1826	1827	1828	1829	1830	1831
3450	1832	1833	1834	1835	1836	1837	1838	1839
3460	1840	1841	1842	1843	1844	1845	1846	1847
3470	1848	1849	1850	1851	1852	1853	1854	1855
3500	1856	1857	1858	1859	1860	1861	1862	1863
3510	1864	1865	1866	1867	1868	1869	1870	1871
3520	1872	1873	1874	1875	1876	1877	1878	1879
3530	1880	1881	1882	1883	1884	1885	1886	1887
3540	1888	1889	1890	1891	1892	1893	1894	1895
3550	1896	1897	1898	1899	1900	1901	1902	1903
3560	1904	1905	1906	1907	1908	1909	1910	1911
3570	1912	1913	1914	1915	1916	1917	1918	1919
3600	1920	1921	1922	1923	1924	1925	1926	1927
3610	1928	1929	1930	1931	1932	1933	1934	1935
3620	1936	1937	1938	1939	1940	1941	1942	1943
3630	1944	1945	1946	1947	1948	1949	1950	1951
3640	1952	1953	1954	1955	1956	1957	1958	1959
3650	1960	1961	1962	1963	1964	1965	1966	1967
3660	1968	1969	1970	1971	1972	1973	1974	1975
3670	1976	1977	1978	1979	1980	1981	1982	1983
3700	1984	1985	1986	1987	1988	1989	1990	1991
3710	1992	1993	1994	1995	1996	1997	1998	1999
3720	2000	2001	2002	2003	2004	2005	2006	2007
3730	2008	2009	2010	2011	2012	2013	2014	2015
3740	2016	2017	2018	2019	2020	2021	2022	2023
3750	2024	2025	2026	2027	2028	2029	2030	2031
3760	2032	2033	2034	2035	2036	2037	2038	2039
3770	2040	2041	2042	2043	2044	2045	2046	2047

4000 to 4777 (Octal)	2048 to 2559 (Decimal)

Octal	Decimal
10000	4096
20000	8192
30000	12288
40000	16384
50000	20480
60000	24576
70000	28672

	0	1	2	3	4	5	6	7
4000	2048	2049	2050	2051	2052	2053	2054	2055
4010	2056	2057	2058	2059	2060	2061	2062	2063
4020	2064	2065	2066	2067	2068	2069	2070	2071
4030	2072	2073	2074	2075	2076	2077	2078	2079
4040	2080	2081	2082	2083	2084	2085	2086	2087
4050	2088	2089	2090	2091	2092	2093	2094	2095
4060	2096	2097	2098	2099	2100	2101	2102	2103
4070	2104	2105	2106	2107	2108	2109	2110	2111
4100	2112	2113	2114	2115	2116	2117	2118	2119
4110	2120	2121	2122	2123	2124	2125	2126	2127
4120	2128	2129	2130	2131	2132	2133	2134	2135
4130	2136	2137	2138	2139	2140	2141	2142	2143
4140	2144	2145	2146	2147	2148	2149	2150	2151
4150	2152	2153	2154	2155	2156	2157	2158	2159
4160	2160	2161	2162	2163	2164	2165	2166	2167
4170	2168	2169	2170	2171	2172	2173	2174	2175
4200	2176	2177	2178	2179	2180	2181	2182	2183
4210	2184	2185	2186	2187	2188	2189	2190	2191
4220	2192	2193	2194	2195	2196	2197	2198	2199
4230	2200	2201	2202	2203	2204	2205	2206	2207
4240	2208	2209	2210	2211	2212	2213	2214	2215
4250	2216	2217	2218	2219	2220	2221	2222	2223
4260	2224	2225	2226	2227	2228	2229	2230	2231
4270	2232	2233	2234	2235	2236	2237	2238	2239
4300	2240	2241	2242	2243	2244	2245	2246	2247
4310	2248	2249	2250	2251	2252	2253	2254	2255
4320	2256	2257	2258	2259	2260	2261	2262	2263
4330	2264	2265	2266	2267	2268	2269	2270	2271
4340	2272	2273	2274	2275	2276	2277	2278	2279
4350	2280	2281	2282	2283	2284	2285	2286	2287
4360	2288	2289	2290	2291	2292	2293	2294	2295
4370	2296	2297	2298	2299	2300	2301	2302	2303

	0	1	2	3	4	5	6	7
4400	2304	2305	2306	2307	2308	2309	2310	2311
4410	2312	2313	2314	2315	2316	2317	2318	2319
4420	2320	2321	2322	2323	2324	2325	2326	2327
4430	2328	2329	2330	2331	2332	2333	2334	2335
4440	2336	2337	2338	2339	2340	2341	2342	2343
4450	2344	2345	2346	2347	2348	2349	2350	2351
4460	2352	2353	2354	2355	2356	2357	2358	2359
4470	2360	2361	2362	2363	2364	2365	2366	2367
4500	2368	2369	2370	2371	2372	2373	2374	2375
4510	2376	2377	2378	2379	2380	2381	2382	2383
4520	2384	2385	2386	2387	2388	2389	2390	2391
4530	2392	2393	2394	2395	2396	2397	2398	2399
4540	2400	2401	2402	2403	2404	2405	2406	2407
4550	2408	2409	2410	2411	2412	2413	2414	2415
4560	2416	2417	2418	2419	2420	2421	2422	2423
4570	2424	2425	2426	2427	2428	2429	2430	2431
4600	2432	2433	2434	2435	2436	2437	2438	2439
4610	2440	2441	2442	2443	2444	2445	2446	2447
4620	2448	2449	2450	2451	2452	2453	2454	2455
4630	2456	2457	2458	2459	2460	2461	2462	2463
4640	2464	2465	2466	2467	2468	2469	2470	2471
4650	2472	2473	2474	2475	2476	2477	2478	2479
4660	2480	2481	2482	2483	2484	2485	2486	2487
4670	2488	2489	2490	2491	2492	2493	2494	2495
4700	2496	2497	2498	2499	2500	2501	2502	2503
4710	2504	2505	2506	2507	2508	2509	2510	2511
4720	2512	2513	2514	2515	2516	2517	2518	2519
4730	2520	2521	2522	2523	2524	2525	2526	2527
4740	2528	2529	2530	2531	2532	2533	2534	2535
4750	2536	2537	2538	2539	2540	2541	2542	2543
4760	2544	2545	2546	2547	2548	2549	2550	2551
4770	2552	2553	2554	2555	2556	2557	2558	2559

5000 to 5777 (Octal)	2560 to 3071 (Decimal)

	0	1	2	3	4	5	6	7
5000	2560	2561	2562	2563	2564	2565	2566	2567
5010	2568	2569	2570	2571	2572	2573	2574	2575
5020	2576	2577	2578	2579	2580	2581	2582	2583
5030	2584	2585	2586	2587	2588	2589	2590	2591
5040	2592	2593	2594	2595	2596	2597	2598	2599
5050	2600	2601	2602	2603	2604	2605	2606	2607
5060	2608	2609	2610	2611	2612	2613	2614	2615
5070	2616	2617	2618	2619	2620	2621	2622	2623
5100	2624	2625	2626	2627	2628	2629	2630	2631
5110	2632	2633	2634	2635	2636	2637	2638	2639
5120	2640	2641	2642	2643	2644	2645	2646	2647
5130	2648	2649	2650	2651	2652	2653	2654	2655
5140	2656	2657	2658	2659	2660	2661	2662	2663
5150	2664	2665	2666	2667	2668	2669	2670	2671
5160	2672	2673	2674	2675	2676	2677	2678	2679
5170	2680	2681	2682	2683	2684	2685	2686	2687
5200	2688	2689	2690	2691	2692	2693	2694	2695
5210	2696	2697	2698	2699	2700	2701	2702	2703
5220	2704	2705	2706	2707	2708	2709	2710	2711
5230	2712	2713	2714	2715	2716	2717	2718	2719
5240	2720	2721	2722	2723	2724	2725	2726	2727
5250	2728	2729	2730	2731	2732	2733	2734	2735
5260	2736	2737	2738	2739	2740	2741	2742	2743
5270	2744	2745	2746	2747	2748	2749	2750	2751
5300	2752	2753	2754	2755	2756	2757	2758	2759
5310	2760	2761	2762	2763	2764	2765	2766	2767
5320	2768	2769	2770	2771	2772	2773	2774	2775
5330	2776	2777	2778	2779	2780	2781	2782	2783
5340	2784	2785	2786	2787	2788	2789	2790	2791
5350	2792	2793	2794	2795	2796	2797	2798	2799
5360	2800	2801	2802	2803	2804	2805	2806	2807
5370	2808	2809	2810	2811	2812	2813	2814	2815

	0	1	2	3	4	5	6	7
5400	2816	2817	2818	2819	2820	2821	2822	2823
5410	2824	2825	2826	2827	2828	2829	2830	2831
5420	2832	2833	2834	2835	2836	2837	2838	2839
5430	2840	2841	2842	2843	2844	2845	2846	2847
5440	2848	2849	2850	2851	2852	2853	2854	2855
5450	2856	2857	2858	2859	2860	2861	2862	2863
5460	2864	2865	2866	2867	2868	2869	2870	2871
5470	2872	2873	2874	2875	2876	2877	2878	2879
5500	2880	2881	2882	2883	2884	2885	2886	2887
5510	2888	2889	2890	2891	2892	2893	2894	2895
5520	2896	2897	2898	2899	2900	2901	2902	2903
5530	2904	2905	2906	2907	2908	2909	2910	2911
5540	2912	2913	2914	2915	2916	2917	2918	2919
5550	2920	2921	2922	2923	2924	2925	2926	2927
5560	2928	2929	2930	2931	2932	2933	2934	2935
5570	2936	2937	2938	2939	2940	2941	2942	2943
5600	2944	2945	2946	2947	2948	2949	2950	2951
5610	2952	2953	2954	2955	2956	2957	2958	2959
5620	2960	2961	2962	2963	2964	2965	2966	2967
5630	2968	2969	2970	2971	2972	2973	2974	2975
5640	2976	2977	2978	2979	2980	2981	2982	2983
5650	2984	2985	2986	2987	2988	2989	2990	2991
5660	2992	2993	2994	2995	2996	2997	2998	2999
5670	3000	3001	3002	3003	3004	3005	3006	3007
5700	3008	3009	3010	3011	3012	3013	3014	3015
5710	3016	3017	3018	3019	3020	3021	3022	3023
5720	3024	3025	3026	3027	3028	3029	3030	3031
5730	3032	3033	3034	3035	3036	3037	3038	3039
5740	3040	3041	3042	3043	3044	3045	3046	3047
5750	3048	3049	3050	3051	3052	3053	3054	3055
5760	3056	3057	3058	3059	3060	3061	3062	3063
5770	3064	3065	3066	3067	3068	3069	3070	3071

6000 to 6777 (Octal) | 3072 to 3583 (Decimal)

Octal	Decimal
10000	4096
20000	8192
30000	12288
40000	16384
50000	20480
60000	24576
70000	28672

	0	1	2	3	4	5	6	7
6000	3072	3073	3074	3075	3076	3077	3078	3079
6010	3080	3081	3082	3083	3084	3085	3086	3087
6020	3088	3089	3090	3091	3092	3093	3094	3095
6030	3096	3097	3098	3099	3100	3101	3102	3103
6040	3104	3105	3106	3107	3108	3109	3110	3111
6050	3112	3113	3114	3115	3116	3117	3118	3119
6060	3120	3121	3122	3123	3124	3125	3126	3127
6070	3128	3129	3130	3131	3132	3133	3134	3135
6100	3136	3137	3138	3139	3140	3141	3142	3143
6110	3144	3145	3146	3147	3148	3149	3150	3151
6120	3152	3153	3154	3155	3156	3157	3158	3159
6130	3160	3161	3162	3163	3164	3165	3166	3167
6140	3168	3169	3170	3171	3172	3173	3174	3175
6150	3176	3177	3178	3179	3180	3181	3182	3183
6160	3184	3185	3186	3187	3188	3189	3190	3191
6170	3192	3193	3194	3195	3196	3197	3198	3199
6200	3200	3201	3202	3203	3204	3205	3206	3207
6210	3208	3209	3210	3211	3212	3213	3214	3215
6220	3216	3217	3218	3219	3220	3221	3222	3223
6230	3224	3225	3226	3227	3228	3229	3230	3231
6240	3232	3233	3234	3235	3236	3237	3238	3239
6250	3240	3241	3242	3243	3244	3245	3246	3247
6260	3248	3249	3250	3251	3252	3253	3254	3255
6270	3256	3257	3258	3259	3260	3261	3262	3263
6300	3264	3265	3266	3267	3268	3269	3270	3271
6310	3272	3273	3274	3275	3276	3277	3278	3279
6320	3280	3281	3282	3283	3284	3285	3286	3287
6330	3288	3289	3290	3291	3292	3293	3294	3295
6340	3296	3297	3298	3299	3300	3301	3302	3303
6350	3304	3305	3306	3307	3308	3309	3310	3311
6360	3312	3313	3314	3315	3316	3317	3318	3319
6370	3320	3321	3322	3323	3324	3325	3326	3327

	0	1	2	3	4	5	6	7
6400	3328	3329	3330	3331	3332	3333	3334	3335
6410	3336	3337	3338	3339	3340	3341	3342	3343
6420	3344	3345	3346	3347	3348	3349	3350	3351
6430	3352	3353	3354	3355	3356	3357	3358	3359
6440	3360	3361	3362	3363	3364	3365	3366	3367
6450	3368	3369	3370	3371	3372	3373	3374	3375
6460	3376	3377	3378	3379	3380	3381	3382	3383
6470	3384	3385	3386	3387	3388	3389	3390	3391
6500	3392	3393	3394	3395	3396	3397	3398	3399
6510	3400	3401	3402	3403	3404	3405	3406	3407
6520	3408	3409	3410	3411	3412	3413	3414	3415
6530	3416	3417	3418	3419	3420	3421	3422	3423
6540	3424	3425	3426	3427	3428	3429	3430	3431
6550	3432	3433	3434	3435	3436	3437	3438	3439
6560	3440	3441	3442	3443	3444	3445	3446	3447
6570	3448	3449	3450	3451	3452	3453	3454	3455
6600	3456	3457	3458	3459	3460	3461	3462	3463
6610	3464	3465	3466	3467	3468	3469	3470	3471
6620	3472	3473	3474	3475	3476	3477	3478	3479
6630	3480	3481	3482	3483	3484	3485	3486	3487
6640	3488	3489	3490	3491	3492	3493	3494	3495
6650	3496	3497	3498	3499	3500	3501	3502	3503
6660	3504	3505	3506	3507	3508	3509	3510	3511
6670	3512	3513	3514	3515	3516	3517	3518	3519
6700	3520	3521	3522	3523	3524	3525	3526	3527
6710	3528	3529	3530	3531	3532	3533	3534	3535
6720	3536	3537	3538	3539	3540	3541	3542	3543
6730	3544	3545	3546	3547	3548	3549	3550	3551
6740	3552	3553	3554	3555	3556	3557	3558	3559
6750	3560	3561	3562	3563	3564	3565	3566	3567
6760	3568	3569	3570	3571	3572	3573	3574	3575
6770	3576	3577	3578	3579	3580	3581	3582	3583

7000 to 7777 (Octal) | 3584 to 4095 (Decimal)

	0	1	2	3	4	5	6	7
7000	3584	3585	3586	3587	3588	3589	3590	3591
7010	3592	3593	3594	3595	3596	3597	3598	3599
7020	3600	3601	3602	3603	3604	3605	3606	3607
7030	3608	3609	3610	3611	3612	3613	3614	3615
7040	3616	3617	3618	3619	3620	3621	3622	3623
7050	3624	3625	3626	3627	3628	3629	3630	3631
7060	3632	3633	3634	3635	3636	3637	3638	3639
7070	3640	3641	3642	3643	3644	3645	3646	3647
7100	3648	3649	3650	3651	3652	3653	3654	3655
7110	3656	3657	3658	3659	3660	3661	3662	3663
7120	3664	3665	3666	3667	3668	3669	3670	3671
7130	3672	3673	3674	3675	3676	3677	3678	3679
7140	3680	3681	3682	3683	3684	3685	3686	3687
7150	3688	3689	3690	3691	3692	3693	3694	3695
7160	3696	3697	3698	3699	3700	3701	3702	3703
7170	3704	3705	3706	3707	3708	3709	3710	3711
7200	3712	3713	3714	3715	3716	3717	3718	3719
7210	3720	3721	3722	3723	3724	3725	3726	3727
7220	3728	3729	3730	3731	3732	3733	3734	3735
7230	3736	3737	3738	3739	3740	3741	3742	3743
7240	3744	3745	3746	3747	3748	3749	3750	3751
7250	3752	3753	3754	3755	3756	3757	3758	3759
7260	3760	3761	3762	3763	3764	3765	3766	3767
7270	3768	3769	3770	3771	3772	3773	3774	3775
7300	3776	3777	3778	3779	3780	3781	3782	3783
7310	3784	3785	3786	3787	3788	3789	3790	3791
7320	3792	3793	3794	3795	3796	3797	3798	3799
7330	3800	3801	3802	3803	3804	3805	3806	3807
7340	3808	3809	3810	3811	3812	3813	3814	3815
7350	3816	3817	3818	3819	3820	3821	3822	3823
7360	3824	3825	3826	3827	3828	3829	3830	3831
7370	3832	3833	3834	3835	3836	3837	3838	3839

	0	1	2	3	4	5	6	7
7400	3840	3841	3842	3843	3844	3845	3846	3847
7410	3848	3849	3850	3851	3852	3853	3854	3855
7420	3856	3857	3858	3859	3860	3861	3862	3863
7430	3864	3865	3866	3867	3868	3869	3870	3871
7440	3872	3873	3874	3875	3876	3877	3878	3879
7450	3880	3881	3882	3883	3884	3885	3886	3887
7460	3888	3889	3890	3891	3892	3893	3894	3895
7470	3896	3897	3898	3899	3900	3901	3902	3903
7500	3904	3905	3906	3907	3908	3909	3910	3911
7510	3912	3913	3914	3915	3916	3917	3918	3919
7520	3920	3921	3922	3923	3924	3925	3926	3927
7530	3928	3929	3930	3931	3932	3933	3934	3935
7540	3936	3937	3938	3939	3940	3941	3942	3943
7550	3944	3945	3946	3947	3948	3949	3950	3951
7560	3952	3953	3954	3955	3956	3957	3958	3959
7570	3960	3961	3962	3963	3964	3965	3966	3967
7600	3968	3969	3970	3971	3972	3973	3974	3975
7610	3976	3977	3978	3979	3980	3981	3982	3983
7620	3984	3985	3986	3987	3988	3989	3990	3991
7630	3992	3993	3994	3995	3996	3997	3998	3999
7640	4000	4001	4002	4003	4004	4005	4006	4007
7650	4008	4009	4010	4011	4012	4013	4014	4015
7660	4016	4017	4018	4019	4020	4021	1022	4023
7670	4024	4025	4026	4027	4028	4029	4030	4031
7700	4032	4033	4034	4035	4036	4037	4038	4039
7710	4040	4041	4042	4043	4044	4045	4046	4047
7720	4048	4049	4050	4051	4052	4053	4054	4055
7730	4056	4057	4058	4059	4060	4061	4062	4063
7740	4064	4065	4066	4067	4068	4069	4070	4071
7750	4072	4073	4074	4075	4076	4077	4078	4079
7760	4080	4081	4082	4083	4084	4085	4086	4087
7770	4088	4089	4090	4091	4092	4093	4094	4095

APPENDIX F

POWERS OF TWO TABLE

POWERS OF TWO TABLE

2^n	n	2^{-n}
1	0	1.0
2	1	0.5
4	2	0.25
8	3	0.125
16	4	0.062 5
32	5	0.031 25
64	6	0.015 625
128	7	0.007 812 5
256	8	0.003 906 25
512	9	0.001 953 125
1 024	10	0.000 976 562 5
2 048	11	0.000 488 281 25
4 096	12	0.000 244 140 625
8 192	13	0.000 122 070 312 5
16 384	14	0.000 061 035 156 25
32 768	15	0.000 030 517 578 125
65 536	16	0.000 015 258 789 062 5
131 072	17	0.000 007 629 394 531 25
262 144	18	0.000 003 814 697 265 625
524 288	19	0.000 001 907 348 632 812 5
1 048 576	20	0.000 000 953 674 316 406 25
2 097 152	21	0.000 000 476 837 158 203 125
4 194 304	22	0.000 000 238 418 579 101 562 5
8 388 608	23	0.000 000 119 209 289 550 781 25
16 777 216	24	0.000 000 059 604 644 775 390 625
33 554 432	25	0.000 000 029 802 322 387 695 312 5
67 108 864	26	0.000 000 014 901 161 193 847 656 25
134 217 728	27	0.000 000 007 450 580 596 923 828 125
268 435 456	28	0.000 000 003 725 290 298 461 914 062 5
536 870 912	29	0.000 000 001 862 645 149 230 957 031 45
1 073 741 824	30	0.000 000 000 931 322 574 615 478 515 625
2 147 483 648	31	0.000 000 000 465 661 287 307 739 257 812 5
4 294 967 296	32	0.000 000 000 232 830 643 653 869 628 906 25
8 589 934 592	33	0.000 000 000 116 415 321 826 934 814 453 125
17 179 869 184	34	0.000 000 000 058 207 660 913 467 407 226 562 5
34 359 738 368	35	0.000 000 000 029 103 830 456 733 703 613 281 25
68 719 476 736	36	0.000 000 000 014 551 915 228 366 851 806 640 625
137 438 953 472	37	0.000 000 000 007 275 957 614 183 425 903 320 312 5
274 877 906 944	38	0.000 000 000 003 637 978 807 091 712 951 660 156 25
549 755 813 888	39	0.000 000 000 001 818 989 403 545 856 475 830 078 125
1 099 511 627 776	40	0.000 000 000 000 909 494 701 772 928 237 915 039 062 5

PRELIMINARY INSTRUCTION SUMMARY AND EXECUTION TIMES

	Instruction Hex Code		Instruction	Execution Time (μsec)[4]	
				In Page Memory	In[1] Bulk Core
AP CONTROL REGISTER INSTRUCTIONS	3rr0vvvv	rr=destination register vvvv=immediate value (source)	Load Immediate	0.2 to 0.25	1.2
	3rrnnnnn	rr=destination register nnnnn=effective source address	Load Register from Bulk Core	1.5	2.5
	3rrnnnnn	rr=destination register nnnnn=effective source address	Load Register from HSDB	0.8	1.8
	3rr000dd	rr=source register dd=destination register	Store Register to Register	0.2 to 0.25	1.2
	3rrnnnnn	rr=source register nnnnn=effective destination address	Store Register to Bulk Core	1.5	2.5
	3rrnnnnn	rr=source register nnnnn=effective destination address	Store Register to HSDB	0.8	1.8
	30Cnnnnn	nnnnn=effective address	Swap PSW (Bulk Core)	2.6	3.4
	30Cnnnnn	nnnnn=effective address	Swap PSW (HSDB)	1.2	2.0
BRANCH/LOOP INSTRUCTIONS	280nnnnn	nnnnn=effective branch address	Unconditional Branch	0.5	1.3
	28cnnnnn	c=conditional code nnnnn=effective branch address	Conditional Branch	0.5	1.3
	2Crnnnnn	r=Branch Link Register (0≤r ≤7) nnnnn=effective branch address	Branch and Link	0.85	1.62
	3C0nnnnn 3D0nnnnn	single instr nnnnn=effective end-of- multi instr Loop address	Loop	0.25	1.0
	3Ekknnnn 3Fkknnnn	single instr kk=Loop count -1 multi instr nnnn=end-of-loop addr	Load and Loop	0.35	1.0
ASSOCIATIVE INSTRUCTIONS	See Appendix A for coding variations		Load Response Store Registers X&Y[2]	0.13 to 0.2	1.0
	See Appendix A for coding variations		Store Response Store Registers to Array	0.3	1.0
	See Appendix A for coding variations		Load MASK	0.13 to 0.2	1.0
	08nn0001 09ii0001	nn=array bit or word addr ii=indirect address code	Move MASK	0.13 to 0.2	1.0
	See Appendix A for coding variations		Generate MASK	0.3	1.0
	See Appendix A for coding variations		Shift Response Store Step Registers	0.125 to 0.2	1.0
	See Appendix A for coding variations		Load Common Register from Array	0.3	1.0
	See Appendix A for coding variations		Store Common Register to Array	0.3	1.0
	01F00000		Find First Responder	0.8	1.0
	03C86640		Reset First Responder	0.8	1.0
	03C82240		Reset Other Responders (one array)	0.8	1.0
	3F82240		Find first Responder and reset all other Responders in that Array	0.8	1.0
	3F86640		Find and Reset First Responder	0.8	1.0

Notes:

(1) Time is increased if Bulk Core or HSDB is currently being accessed by another STARAN S element; i.e., Sequential Processor, Pager, or PI/O.

(2) Add 0.025 μsec if the data being loaded is from Array Bit Column or Word. Times shown assume Response Store to Response Store Transfer.

Reference notes 3 and 4 are on the following page.

PRELIMINARY INSTRUCTION SUMMARY AND EXECUTION TIMES (cont)

	Instruction Hex Code		Instruction	Execution Time (μsec)(4)	
				In Page Memory	In (1) Bulk Core
PAGER INSTRUCTIONS	4000nnnn	nnnn= page memory address	Load PUT		1.0
	8kkk0000	kkk=no. of words moved	Move Data (n=no. of words transferred)		1.3(n+1)
	Ckkknnnn	kkk=no. of words moved nnnn=page memory address	Move Data and Load PUT		2.0
	000fffff	fffff=function code	Issue EXF		2.0
	0004nnnn	nnnn=pager Branch address; new GET address	Pager Branch		2.0
	00008000		Halt Pager		2.0
	00008005		Skip Pager Instruction		2.0
	00008001		Halt Pager and Skip Instruction		2.0
EXTERNAL FUNCTION INSTRUCTIONS	38000sss	sss=sense status and set switches	Pager Port Switches	1.3	2.0
	3806nnns	nnn=interlock no. s=sense status and new state bits	Interlocks	1.3	2.0
	3800800s	s=sense status and new state bits	Program Pager State		2.0
	3804nnnn	nnnn=new GET address	Pager Load GET	1.3	2.0
	38010nns	nn=interrupt no. s=sense status and new state bits	AP control Interrupts(3)	1.3	2.0
	3800200s	s=sense status and new state bits	AP activity	1.3	2.0
	3800201s	s=sense status and new state bits	Reset Loop		2.0
	3804nns	nn=error no. s=sense status and new state bits	Error Control	1.3	2.0
	38012nns	nn=interrupt vector address s=sense status and new state bits	Sequential Control Interrupts	1.3	2.0
	38002nn0	nn=Clear and Reset address	Resets and Clears	1.3	2.0

Notes: (cont)

(3) Time is increased if EXF logic is currently being used by another STARAN S element;i.e., Sequential Processor or Pager.

(4) All times shown are absolute for nnnnn. If address tag modification is used add 0.4 when using R0 thru R7, and add 0.1 when using DP.

INDEX

INDEX

INDEX

INDEX

INDEX

INDEX

INDEX