INTRODUCTION

System 2400 is a 24-bit general purpose systems oriented digital computer which provides extensive computing capability in a low cost processor of advanced design. It is ideally suited to a wide range of digital applications where performance, reliability, programming efficiency, and cost effectiveness are of prime consideration.

STANDARD FEATURES - The basic sytem 2400 central processor incorporates a number of standard features normally available only as separately priced options. Included in the list of standard features are:

- Memory Protection
- Memory Parity
- Priority Interrupt System
- Power Fail Detection
- Real Time Clock
- Cabinet Enclosure

MEMORY - The main memory is a 750 nanosecond 24-bit ferrite core memory with standard byte parity checking. The basic memory size of 8192 24-bit words is modularly expandable to 32768 words in 8192 word increments. The 2400 CPU is pre-wired to accept the maximum memory configuration without additional mainframe hardware.

ADDRESSING - All of memory is directly addressable by each single word memory reference instruction. Programming problems inherent with paged addressing schemes are eliminated on the System 2400 computer.

REGISTERS - Four general purpose programmable registers are provided, three of which may be used for indexing. All memory reference, shift, and input-output instructions may be indexed using any of the three index registers.

INSTRUCTIONS - The System 2400 instruction set includes 100 basic commands organized into six instruction classes: Memory Reference, Immediate, Shift, Input-Output, Register-Register, and Control. Significant features of the instruction set include:

- Hardware Multiply and Divide
- Double Precision Arithmetic
- Signed 16-Bit Immediate Instructions
- Variable Shift Instructions
- Immediate, Word, and Double Word Comparsion Instructions
- Conditional Branching Instructions

INPUT-OUTPUT SYSTEM - The input-output system is organized around a multiple access buss structure that allows direct communication between two input-output devices, an input-output device and the main memory, the CPU and an input-output device, or the CPU and main memory. Buss priority is determined during each memory cycle resulting in a maximum latency time for high priority direct memory access input-output requests of 750 nanoseconds. The input-output system provides for byte or word transfers under program control and high speed block data transfer under direct memory access control. The basic input-output system is pre-wired to accept up to eight programmed I/O (PIO) controllers and up to five direct memory access (DMA) controllers. Block transfer rates of up to 32 million bits per second are possible under DMA control.

PRIORITY INTERRUPT SYSTEM - The basic System 2400 processor accepts up to 255 external interrupt signals. The priority interrupt system provides a fully nested interrupt structure with automatic source identification by hardware. Assignment of interrupt priorities is flexible and completely independent of the input-output system. Interrupt priorities may be reassigned without requiring corresponding modifications to the input-output addressing structure. The interrupt structure is specifically organized to facilitiate reentrant interrupt processing within a supervisory environment.

MEMORY PROTECTION - Memory protection is a standard system feature which permits a program to be restricted within an area of memory as defined by two memory address limits. An attempt to write or execute outside of the restricted area results in a system trap interrupt. The limits defining a protected area of memory may be altered under program control. Memory protection on the System 2400 computer is primarily intended for use in a supervisory multiprogrammed environment where memory protection is dynamically activated for each task at the onset of its execution.

REAL TIME CLOCK - The real time clock provides a low priority CPU interrupt each 16 2/3 milliseconds. The clock incorporates a watchdog timer which generates a system trap interrupt in case a clock interrupt is not acknowleged within 16 2/3 milliseconds.

POWER FAIL DETECTION - This feature insures orderly system shutdown and automatic restart in case of power failure.

PERIPHERALS - A complete line of peripheral equipment is available for the System 2400 computer. Representative peripheral include:

Keyboard Printers - 10 characters per second. Also available with integrated paper-tape reader at 10 characters per second and punch at 10 characters per second.

Card Readers - 300, 600 and 1000 cards per minute. Read 80-column cards in parallel by column, photoelectrically. Automatic conversion to ASCII.

Line Printers - Fully buffered, 300, 600 and 1000 lines per minute, with 80 and 132 print positions.

Magnetic-Tape Units - Nine-track: IBM-compatible, 37.5 inches per second, 800 bytes per inch, 30,000 bytes per second transfer rate. Seven-track: IBM-compatible, 37.5 inches per second, 200, 556 and 800 bytes per inch, transfer rates of 7,500, 20,850 and 30,000 characters per second.

Disc Files - Capacity 831,488 to 8,304,880 words per storage unit; transfer rate of 104,000 words per second; average access times between 12.5 and 20 milliseconds.

Paper-Tape Readers and Punches - Readers with speeds of 10 and 500 characters per second. Punches with speeds of 10 and 120 characters per second. Both use eight-level codes.

Analog-To-Digital Converter - Analog-to-digital converters and multiplexers with sampling rates that exceed 40,000 samples per second for 12 bit resolution; low-level to high-level.

Digital-To-Analog Converters - 14-bit resolution, low impedance output, fast settling time, updating rates exceeding 100,000 channels per second.

Digital Inputs - 24-bit contact closure and voltage level inputs.

Digital Outputs - 24-bit contact closure and voltage level outputs.

Process Inputs - 24-bit contact closure and voltage level inputs that generate processor interrupts in response to external state or level variations.

SOFTWARE - A comprehensive package of systems oriented software is available for the 2400 computer. Representative software includes:

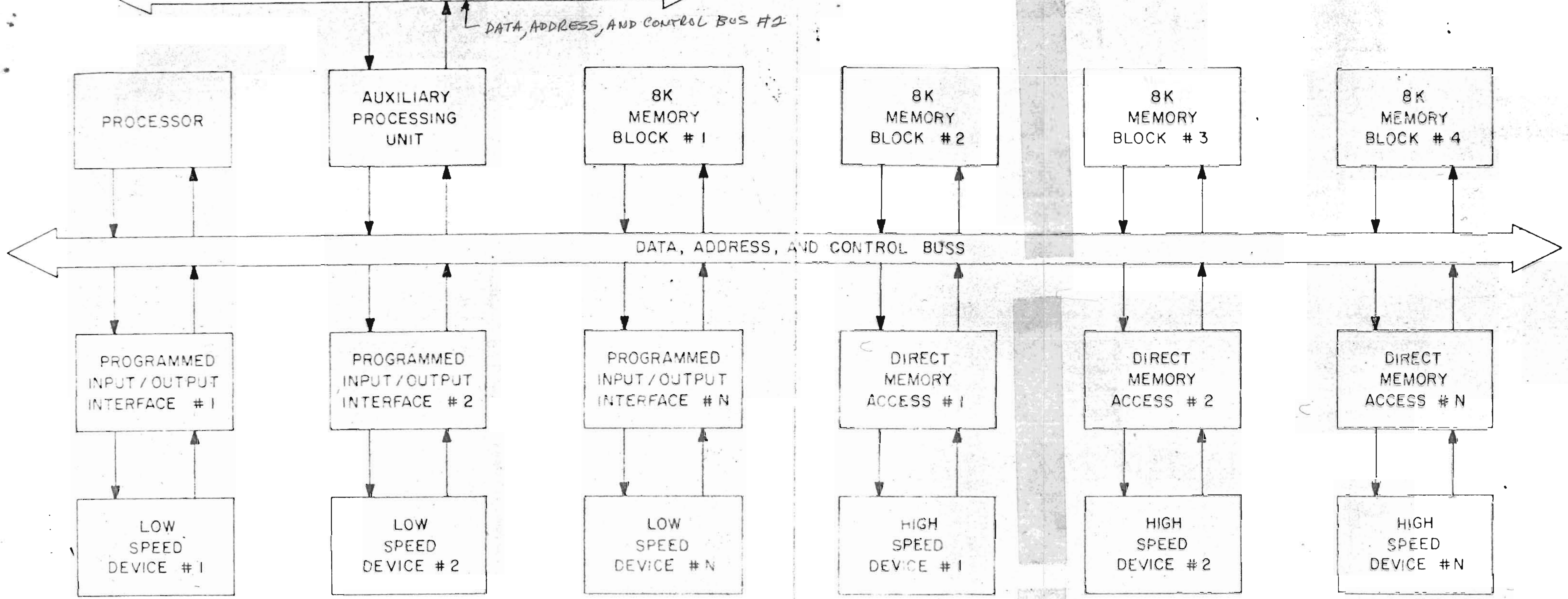
Multiprogramming Executive (MULTEX) - MULTEX is a disc oriented real time operating system designed to support priority oriented real time processing on the System 2400 computer. It provides for priority scheduling and functional support of multiple foreground tasks with concurrent background batch processing. Tasks may be permanently resident in memory or read from disc into memory for execution. A multiprogrammed mix of such tasks may be active at any given time with the highest priority task in actual execution.

MULTEX provides for automatic task scheduling based upon the system clock, the occurrence of an interrupt, or by task request. Interrupt scheduling may be performed in either direct or deferred mode. Direct interrupt scheduling is employed where critical response times are required and processing is performed at a priority level determined by hardware. Deferred interrupt scheduling is used where less critical response times are required and processing is performed at a priority level established by software.

MULTEX provides comprehensive input-output facilities at both the implicit and explicit levels which permit full I/O overlap with computing among separate tasks. Explicit I/O is inherently device dependent in nature as contrasted to implicit I/O which is device independent by definition. Implicit I/O is performed with respect to a system of file management which incorporates both disc files and I/O devices.

MULTEX provides for background batch processing at the lowest system priority level concurrent with foreground operation. The basic functions provided by the background system are indicated as follows:

- Assignment of Background I/O Units
- Assembly/Compilation
- On-line Creation of Foreground/Background Tasks
- Creation of Files


DATA, ADDRESS, AND CONTROL BUS #2
PROCESSOR
AUXILIARY PROCESSING UNIT
8K MEMORY BLOCK # 1
8K MEMORY BLOCK # 2
8K MEMORY BLOCK # 3
8K MEMORY BLOCK # 4
DATA, ADDRESS, AND CONTROL BUSS
PROGRAMMED INPUT / OUTPUT INTERFACE # 1
PROGRAMMED INPUT / OUTPUT INTERFACE # 2
PROGRAMMED INPUT / OUTPUT INTERFACE # N
DIRECT MEMORY ACCESS # 1
DIRECT MEMORY ACCESS # 2
DIRECT MEMORY ACCESS # N
LOW SPEED DEVICE # 1
LOW SPEED DEVICE # 2
LOW SPEED DEVICE # N
HIGH SPEED DEVICE # 1
HIGH SPEED DEVICE # 2
HIGH SPEED DEVICE # N

The list of background functions may be extended on-line to incorporate user written background tasks.

Assembler - The System 2400 assembler is available in both a stand-alone version and a version which operates as a MULTEX background task. The assembler provides the following basic features.

- User Defined Macro Procedures
- Conditional Assembly Capability
- Fixed and Floating Point Constants
- Absolute and/or Relocatable Object Code

Real Time Fortran - The System 2400 real time Fortran compiler operates as a MULTEX background task permitting foreground/background programs to be written in Fortran. The language exceeds USASI Basic Fortran standards and incorporates extensive file manipulation statements.

System Generation - The system generation program builds a version of MULTEX adapted to system configuration requirements. It operates as a stand-along program and accepts input from a master file of user specified system definition parameters.

Input-Output Drivers - Stand alone I/O drivers are provided for all system 2400 peripherals.

Diagnostic Programs - Stand-alone diagnostic programs are provided which permit comprehensive testing of selected System 2400 hardware components.

INSTRUCTION SET

CODE (HEXADECIMAL)	MNEMONIC	INSTRUCTION	CYCLES
ARITHMETIC			
B4	A	Add	2
F4	AD	Add Double	3
34	AIA	Add Immediate A	1
35	AIX	Add Immediate X	1
36	AIY	Add Immediate Y	1
37	AIZ	Add Immediate Z	1
BC	DM	Decrement Memory	3
E4	D	Divide	10
B0	IM	Increment Memory	3
EC	M	Multiply	10
20	NEGA	Negate A	1
E8	NEGD	Negate Double	2
21	NEGX	Negate X	1
22	NEGY	Negate Y	1
23	NEGZ	Negate Z	1
B8	S	Subtract	2
F8	SD	Subtract Double	3
38	SIA	Subtract Immediate A	1
39	SIX	Subtract Immediate X	1
3A	SIY	Subtract Immediate Y	1
3B	SIZ	Subtract Immediate Z	1
BRANCH			
90	B	Branch	1
14	BE	Branch if Equal	1
18	BG	Branch if Greater	1
98	BI	Branch Indirect	2
10	BL	Branch if Less	1
54	BNE	Branch if Not Equal	1
58	BNG	Branch if Not Greater	1
50	BNL	Branch if Not Less	1
5C	BNV	Branch if No Overflow	1
9C	BR	Branch and Restore	2
94	BS	Branch and Save	2
1C	BV	Branch if Overflow	1

INSTRUCTION SET

CODE (HEXADECIMAL)	MNEMONIC	INSTRUCTION	CYCLES
COMPARE			
70	CPA	Compare A	2
FC	CPD	Compare Double	3
3C	CIA	Compare Immediate A	1
3D	CIX	Compare Immediate X	1
3E	CIY	Compare Immediate Y	1
3F	CIZ	Compare Immediate Z	1
74	CPX	Compare X	2
78	CPY	Compare Y	2
7C	CPZ	Compare Z	2
LOGICAL			
A4	N	Logical And	2
24	NIA	Logical And Immediate A	1
25	NIX	Logical And Immediate X	1
26	NIY	Logical And Immediate Y	1
27	NIZ	Logical And Immediate Z	1
AC	X	Logical Exclusive Or	2
2C	XIA	Logical Exclusive Or Immediate A	1
2D	XIX	Logical Exclusive Or Immediate X	1
2E	XIY	Logical Exclusive Or Immediate Y	1
2F	XIZ	Logical Exclusive Or Immediate Z	1
A8	O	Logical Inclusive Or	2
28	OIA	Logical Inclusive Or Immediate A	1
29	OIX	Logical Inclusive Or Immediate X	1
2A	OIY	Logical Inclusive Or Immediate Y	1
2B	OIZ	Logical Inclusive Or Immediate Z	1
SHIFT			
C8	SLC	Shift Left Closed	2+N/3
CC	SLCD	Shift Left Closed Double	2+N/3
C0	SLL	Shift Left Logical	2+N/3
C4	SLLD	Shift Left Logical Double	2+N/3
D8	SRA	Shift Right Algebraic	2+N/3
DC	SRAD	Shift Right Algebraic Double	2+N/3
D0	SRL	Shift Right Logical	2+N/3
D4	SRLD	Shift Right Logical Double	2+N/3

INSTRUCTION SET

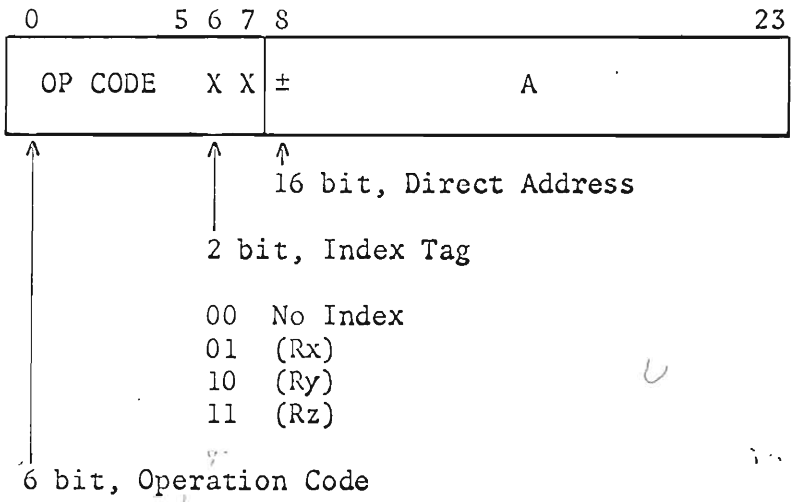
CODE (HEXADECIMAL)	MNEMONIC	INSTRUCTION	CYCLES
TRANSFER			
05	CAX	Copy A To X	1
06	CAY	Copy A To Y	1
07	CAZ	Copy A To Z	1
09	CXA	Copy X To A	1
0A	CYA	Copy Y To A	1
0B	CZA	Copy Z To A	1
0D	IAX	Interchange A and X	1
0E	IAY	Interchange A and Y	1
0F	IAZ	Interchange A and Z	1
60	LDA	Load A	2
F0	LDD	Load Double	3
30	LIA	Load Immediate A	1
31	LIX	Load Immediate X	1
32	LIY	Load Immediate Y	1
33	LIZ	Load Immediate Z	1
64	LDX	Load X	2
68	LDY	Load Y	2
6C	LDZ	Load Z	2
40	STA	Store A	2
E0	STD	Store Double	3
44	STX	Store X	2
48	STY	Store Y	2
4C	STZ	Store Z	2
INPUT/OUTPUT			
84	RD	Read Direct	2
80	RS	Read Status	2
8C	WD	Write Direct	2
88	WS	Write Status	2
MISCELLANEOUS			
04	CV	Clear Overflow	1
02	DSI	Disable Interrupts	1
03	ENI	Enable Interrupts	1
A0	EX	Execute	1+INSTR
00	HALT	Halt	1
01	NOP	No Operation	1
08	SV	Set Overflow	1
0C	SVC	Supervisor Call	1

INSTRUCTION FORMATS

Five instruction types are used in the dcs 2400 instruction set. All instructions occupy a single 24 bit word.

1. Memory Reference Instructions

   The most commonly used instruction is the memory reference type instruction. These instructions can directly address the entire memory (0 to 32K words) with or without indexing. The A field in the instruction specifies a 16 bit signed displacement memory address. The X field specifies optional indexing with the contents of index registers Rx, Ry, or Rz. The effective memory address (actual memory address) is the sum of the A value with the contents of the specified index register.



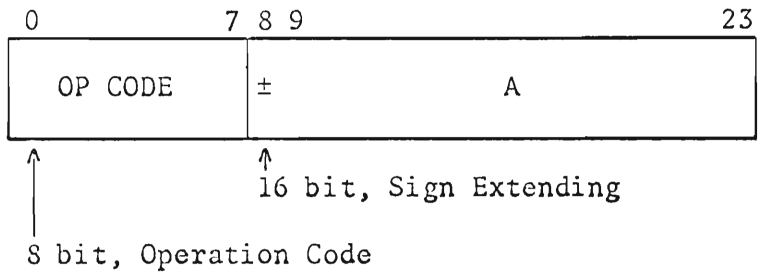
00	No Index
01	(Rx)
10	(Ry)
11	(Rz)

OP CODE	MNEMONIC	INSTRUCTION
1011 01XX	A	ADD
1111 01XX	AD	ADD DOUBLE
1001 00XX	B	BRANCH
0001 01XX	BE	BRANCH IF EQUAL
0001 10XX	BG	BRANCH IF GREATER
1001 10XX	BI	BRANCH INDIRECT
0001 00XX	BL	BRANCH IF LESS
0101 01XX	BNE	BRANCH IF NOT EQUAL
0101 10XX	BNG	BRANCH IF NOT GREATER
0101 00XX	BNL	BRANCH IF NOT LESS
0101 11XX	BNV	BRANCH IF NO OVERFLOW
1001 11XX	BR	BRANCH AND RESTORE
1001 01XX	BS	BRANCH AND SAVE
0001 11XX	BV	BRANCH IF OVERFLOW
0111 00XX	CPA	COMPARE A
1111 11XX	CPD	COMPARE DOUBLE

OP CODE	MNEMONIC	INSTRUCTION
0111 01XX	CPX	COMPARE X
0111 10XX	CPY	COMPARE Y
0111 11XX	CPZ	COMPARE Z
1011 11XX	DM	DECREMENT MEMORY
1110 01XX	D	DIVIDE
1010 00XX	EX	EXECUTE
1011 00XX	IM	INCREMENT MEMORY
0110 00XX	LDA	LOAD A
1111 00XX	LDD	LOAD DOUBLE
0110 01XX	LDX	LOAD X
0110 10XX	LDY	LOAD Y
0110 11XX	LDZ	LOAD Z
1010 01XX	N	LOGICAL AND
1010 11XX	X	LOGICAL EXCLUSIVE OR
1010 10XX	O	LOGICAL INCLUSIVE OR
1110 11XX	M	MULTIPLY
0100 00XX	STA	STORE A
1110 00XX	STD	STORE DOUBLE
0100 01XX	STX	STORE X
0100 10XX	STY	STORE Y
0100 11XX	STZ	STORE Z
1011 10XX	S	SUBTRACT
1111 10XX	SD	SUBTRACT DOUBLE

2. Immediate Instructions

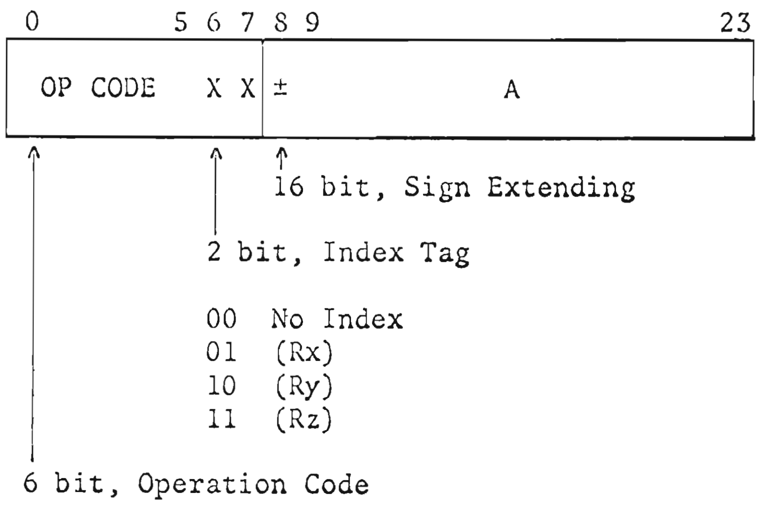
The immediate instructions (also called Literal instructions) use the A field in the instruction word as the second operand. The first operand is contained in one of the working registers (Ra, Rx, Ry, or Rz) and the result if saved replaces the previous contents of the register. The least significant sixteen bits of the instruction becomes a 24 bit operand when the sign bit (bit 8) is extended into bits 0 to 7.



OP CODE	MNEMONIC	INSTRUCTION
0011 0100	AIA	ADD IMMEDIATE A
0011 0101	AIX	ADD IMMEDIATE X
0011 0110	AIY	ADD IMMEDIATE Y
0011 0111	AIZ	ADD IMMEDIATE Z
0011 1100	CIA	COMPARE IMMEDIATE A
0011 1101	CIX	COMPARE IMMEDIATE X
0011 1110	CIY	COMPARE IMMEDIATE Y
0011 1111	CIZ	COMPARE IMMEDIATE Z
0011 0000	LIA	LOAD IMMEDIATE A
0011 0001	LIX	LOAD IMMEDIATE X
0011 0010	LIY	LOAD IMMEDIATE Y
0011 0011	LIZ	LOAD IMMEDIATE Z
0010 0100	NIA	LOGICAL AND IMMEDIATE A
0010 0101	NIX	LOGICAL AND IMMEDIATE X
0010 0110	NIY	LOGICAL AND IMMEDIATE Y
0010 0111	NIZ	LOGICAL AND IMMEDIATE Z
0010 1100	XIA	LOGICAL EXCLUSIVE OR IMMEDIATE A
0010 1101	XIX	LOGICAL EXCLUSIVE OR IMMEDIATE X
0010 1110	XIY	LOGICAL EXCLUSIVE OR IMMEDIATE Y
0010 1111	XIZ	LOGICAL EXCLUSIVE OR IMMEDIATE Z
0010 1000	OIA	LOGICAL INCLUSIVE OR IMMEDIATE A
0010 1001	OIX	LOGICAL INCLUSIVE OR IMMEDIATE X
0010 1010	OIY	LOGICAL INCLUSIVE OR IMMEDIATE Y
0010 1011	OIZ	LOGICAL INCLUSIVE OR IMMEDIATE Z
0011 1000	SIA	SUBTRACT IMMEDIATE A
0011 1001	SIX	SUBTRACT IMMEDIATE X
0011 1010	SIY	SUBTRACT IMMEDIATE Y
0011 1011	SIZ	SUBTRACT IMMEDIATE Z

3. Shift Instruction

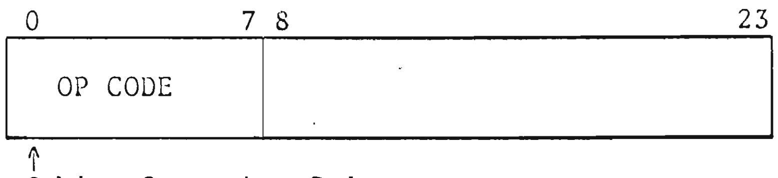
The shift instructions provide for arithmetic and logical manipulation of information contained in the Ra and Rd registers. The operation code specifies the register to be shifted and the type of shift operation to be performed. The number of bit positions to be shifted (N) is specified by the sum of the value A with the contents of the index register specified by X. After the shift count is modified by the index value, only the eight least significant bits are used. The shift count will be in the range $0 \leq N \leq 255$.



OP CODE	MNEMONIC	INSTRUCTION
1100 10XX	SLC	SHIFT LEFT CLOSED
1100 11XX	SLCD	SHIFT LEFT DOUBLE CLOSED
1100 00XX	SLL	SHIFT LEFT LOGICAL
1100 01XX	SLLD	SHIFT LEFT LOGICAL DOUBLE
1101 10XX	SRA	SHIFT RIGHT ALGEBRAIC
1101 11XX	SRAD	SHIFT RIGHT ALGEBRAIC DOUBLE
1101 00XX	SRL	SHIFT RIGHT LOGICAL
1101 01XX	SRLD	SHIFT RIGHT LOGICAL DOUBLE

4. Register-Register and Control Instructions

The register-register and control instructions use only the operation code field. Bits 8-23 in the instruction are ignored. The register-register instructions move information between the working registers. The control instructions control the CPU and interrupt system status.

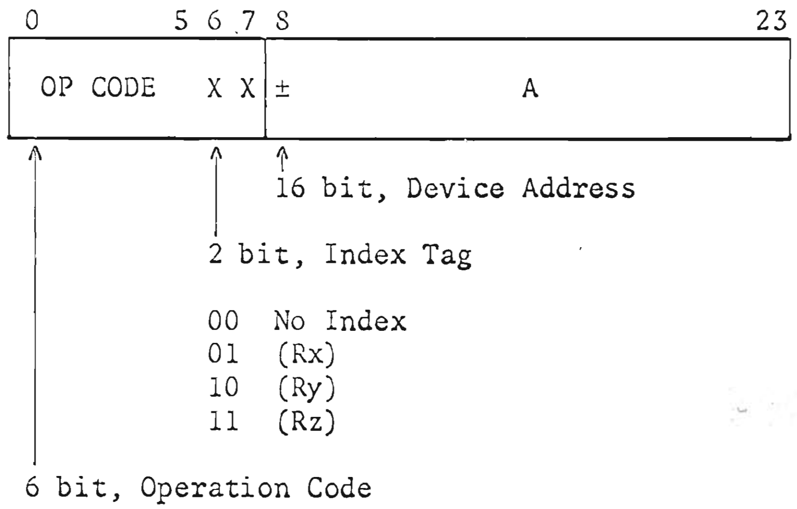


8 bit, Operation Code

OP CODE	MNEMONIC	INSTRUCTION
0000 0101	CAX	COPY A TO X
0000 0110	CAY	COPY A TO Y
0000 0111	CAZ	COPY A TO Z
0000 1001	CXA	COPY X TO A
0000 1010	CYA	COPY Y TO A
0000 1011	CZA	COPY Z TO A
0000 1000	CV	CLEAR OVERFLOW
0000 0010	DSI	DISABLE INTERRUPTS
0000 0011	ENI	ENABLE INTERRUPTS
0000 0000	HALT	HALT
0000 1101	IAX	INTERCHANGE A AND X
0000 1110	IAY	INTERCHANGE A AND Y
0000 1111	IAZ	INTERCHANGE A AND Z
0010 0000	NEGA	NEGATE A
1110 1000	NEGD	NEGATE DOUBLE
0010 0001	NEGX	NEGATE X
0010 0010	NEGY	NEGATE Y
0010 0011	NEGZ	NEGATE Z
0000 0001	NOP	NO OPERATION
0000 0100	SV	SET OVERFLOW

5. Input-Output Instructions

The input-output instructions transfer data, status and control information directly between the Ra register and the input-output system. Input-output device addresses are computed in the same way effective memory addresses are computed for memory reference instructions. The device address is the sum of the value A with the contents of the specified index register.



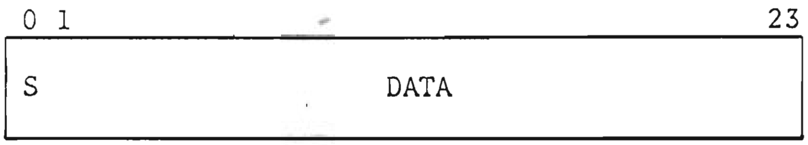
OP CODE	MNEMONIC	INSTRUCTION
1000 01XX	RD	READ DIRECT
1000 00XX	RS	READ STATUS
1000 10XX	WS	WRITE STATUS
1000 11XX	WD	WRITE DIRECT

DATA FORMATS

Three types of formats are used to represent numerical data: (1) Integer, (2) Double Precision, (3) Floating Point.

1. Integer

   The basic data format is a 24 bit signed 2's complement binary integer.



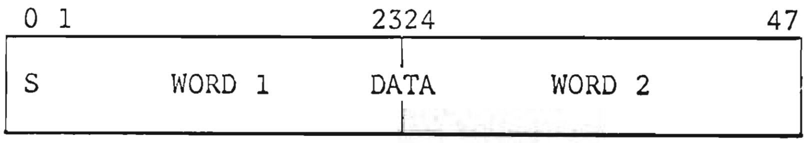
Sign Bit
0 = Positive
1 = Negative

The number represented is defined as a binary or hexadecimal integer, with bit 0 being the most significant position, and bit 23 the least significant.

The maximum range of signed integers which may be represented by a single word is $800000_{16} \leq i \leq 7FFFFF_{16}$ (decimal: $-8388608 \leq i \leq +8388607$). Negative quantities are expressed in 2's complement form. (The 2's complement of a number is obtained by inverting each bit of the binary number and adding one.)

2. Double Precision

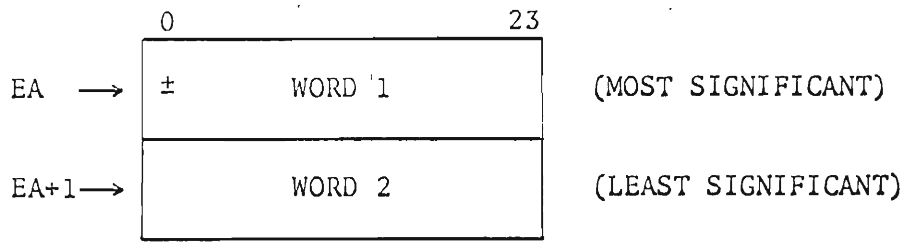
   Double Precision arithmetic utilizes two machine words to represent a 48 bit, signed binary integer with the following format:



Sign Bit
0 = Positive
1 = Negative

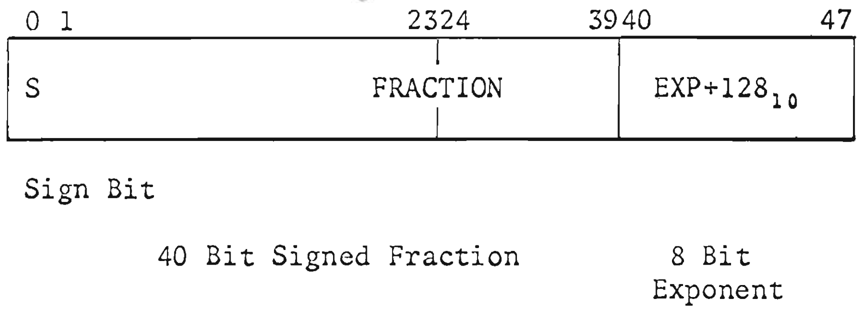
Numbers up to 2^{47} may be represented in this format. The minimum range is $800000000000_{16} \leq i \leq 7FFFFFFFFFFF_{16}$ (decimal:-140737488344328$\leq i \leq$+14073788355237). Negative double precision numbers are represented in 2's complement form.

The effective memory address of a double precision operand specifies the address of the most significant half of the operand. The least significant half is located at the effective memory address plus one.



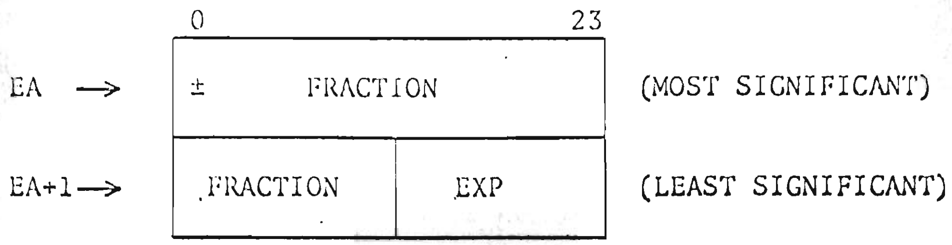
3. Floating Point

The floating point format used two machine words to represent the floating point number.



The 40 bit signed 2's complement fraction occupies bits 0-39 and the 8 bit excess 128 exponent occupies bit 40-47. The radix point of the fraction is assumed to be immediately to the left of the high order fraction digit.

The effective memory address of a floating point operand specifies the address of the most significant half of the fraction. The least significant half of the fraction and the exponent are located at the effective memory address plus one.


0
23
EA →
±
FRACTION
(MOST SIGNIFICANT)
EA+1 →
FRACTION
EXP
(LEAST SIGNIFICANT)

CONTROL PANEL

Controls and Indicators

Address Display - 16 bit indicator that displays and CPU address buss. When the CPU is in the halt mode this indicator displays the contents of the program counter.

Data Display - 24 bit indicator that displays the CPU data buss. When the CPU is in the halt mode this indicator displays the contents of the register or memory location selected by the display control switches.

Status Display - 8 bit indicator displays the program status.

I -	If lighted this display indicates that the interrupt system is ENABLED.
< -	If lighted this display indicates that the algebraic result of the last arithmetic or logical operation was LESS THAN ZERO.
> -	If lighted this display indicates that the algebraic result of the last arithmetic or logical operation was GREATER THAN ZERO.
	If both the < and > indicators are lighted, the last arithmetic or shift operation resulted in an ARITHMETIC OVERFLOW.
RUN -	If lighted this display indicates that the CPU is in the RUN mode.
HALT -	If lighted this display indicates that the CPU is in the HALT mode.
I/O -	If lighted this display indicates that an input or output operation is in progress.
PE -	If lighted this display indicates that a memory PARITY ERROR has occurred. The occurrence of a parity error will halt the CPU.
BP -	If lighted this display indicates that a program BREAKPOINT is set.

Data Switch Register - 24 bit switch register used to enter address and data information.

Run - This momentary contact switch places the CPU in the run mode. Instruction execution starts at the memory address stored in the program counter.

Halt - This momentary contact switch places the CPU in the halt mode. Instruction execution stops with the address of the next instruction to be executed stored in the program counter. If the CPU is in the halt mode, operation of the halt switch will cause a single instruction to be executed each time the switch is operated.

Reset - This momentary contact switch resets the CPU and all input-output control units. This switch is operational only when the CPU is halted.

IPL - This momentary contact switch initiates an initial program load operation from a preselected input-output device. This switch is operational only when the CPU is halted.

Alter PC - This momentary contact switch causes the address set in bits 8-23 of the data switch register to be copied into the program counter and displayed in the address display. The contents of the memory location addressed by the program counter will be displayed in the data display.

Increment PC - This momentary contact switch causes one to be added to the contents of the program counter. The new contents of the program counter will be displayed in the address display. The contents of the memory location addressed by the program counter will be displayed in the data display.

Alter MEM - This momentary contact switch causes the data word set in the data switch register to be copied into the memory location addressed by the program counter. The new contents of the memory location will be displayed in the data display.

Display MEM - This momentary contact switch causes the contents of the memory location addressed by the program counter to be displayed in the data display.

Alter Ra - This momentary contact switch causes the data word set in the data switch register to be copied into the Ra register. The new contents of the Ra register will be displayed in the data display.

Display Ra - This momentary contact switch causes the contents of the Ra register to be displayed in the data display.

Alter Rx - This momentary contact switch causes the data word set in the data switch register to be copied into the Rx register. The new contents of the Rx register will be displayed in the data display.

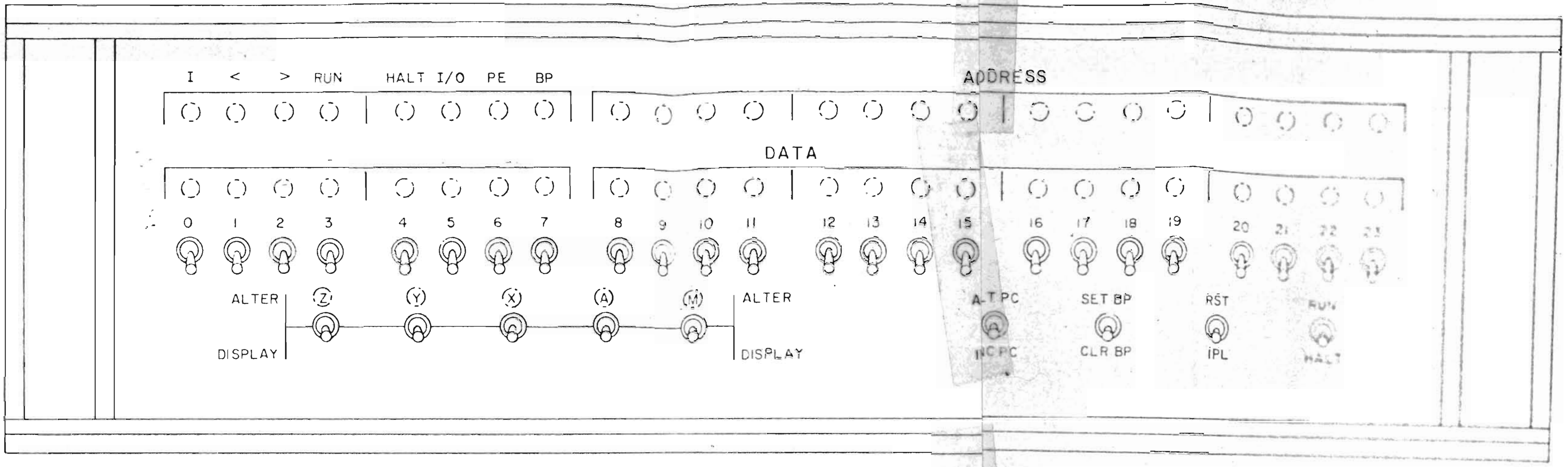
Display Rx - This momentary contact switch causes the contents of the Rx register to be displayed in the data display.

Alter Ry - This momentary contact switch causes the data word set in the data switch register to be copied into the Ry register. The new contents of the Ry register will be displayed in the data display.

Display Ry - This momentary contact switch causes the contents of the Ry register to be displayed in the data display.

Alter Rz - This momentary contact switch causes the data word set in the data switch register to be copied into the Rz register. The new contents of the Rz register will be displayed in the data display.

Display Rz - This momentary contact switch causes the contents of the Rz register to be displayed in the data display.



dcs 2400

CONTROL PANEL

ASSEMBLY LANGUAGE

GENERAL

The System 2400 Assembly Language is a symbolic programming language which facilitates the construction of machine language programs through the use of symbolic notation for machine instructions and data. The translation of an assembly language program into its machine language representation is performed by an assembler program. The assembly process is one of converting information expressed in assembly language into equivalent information expressed in machine language.

The fundamental program element is the statement. A program represented as a sequence of statements recorded on an external medium is referred to as a source program. The assembler reads and translates the source program into an equivalent machine language object program. The translation procedure requires two separate passes or readings of the source program. During pass one a table of program symbols is built for reference during pass two. Pass two provides a printed listing of the assembled program and an external record of the resulting object program which may be loaded into the computer for execution.

STATEMENTS

Statements are of three basic types: Comment, Instruction, and Directive.

Comment statements serve as program documentation aids and are simply reproduced on the program listing. A comment statement is indicated by an asterisk (*) in statement position one.

Instruction statements are translated directly by the assembler into corresponding machine instructions.

Directive statements are special instructions to the assembler which serve to direct the assembly process and generate various forms of data.

Instruction and Directive statements are subdivided into three fixed length statement fields:

Statement Position:	1-6	Label Field
	8-13	Operation Field
	15-72	Operand Field

Following the first blank operand position, the remainder of the statement is ignored and is available for comments.

Label Field

The Label Field optionally contains a statement label consisting of from one to six alphanumeric characters with the first character alphabetic.

Operation Field

The Operation Field contains an operation mnemonic associated with an instruction or assembler directive.

Operand Field

The Operand Field contains one or more operands associated with the operation to be performed.

LOCATION COUNTER

The Location Counter is used by the assembler to assign consecutive addresses to statement derived object code. The location counter is absolute if it represents an actual memory address and relocatable if it represents a relative displacement from the beginning of the program. In the relocatable case, the initial value of the location counter is specified at load time as a relocation bias corresponding to the address in memory at which the program is to originate. The location counter is initialized by the assembler to relocatable zero and is controlled by the ORG assembler directive.

TERMS

Terms are the basic units required in building expressions. A term may be absolute or relocatable depending upon its type and the mode of the location counter at the time it is defined.

Asterisk

The Asterisk (*) is a term representing the current value and mode of the location counter.

Symbol

A Symbol consists of from one to six alphanumeric characters with the first character alphabetic. A symbol is usually defined by its appearance as a statement label and is assigned the value and mode of the location counter at the time it is encountered.

Constant

Constant terms are absolute and are assigned their indicated values. There are three types of constant terms: Hexadecimal, Decimal, and Character.

Hexadecimal Constant

A Hexadecimal Constant consists of a single hexadecimal number, consisting of at most six hexadecimal digits, enclosed in apostrophes and preceded by the letter X.

```
Examples:     X'3AF'
              X'1000'
              X'FFFFF0'
```

Decimal Constant

A Decimal Constant consists of from one to seven decimal digits in the numeric range 0 to 8,388,607.

```
Examples:     100
              10000
              6999500
```

Character Constant

A Character Constant consists of a single ASCII character enclosed in apostrophes and preceded by the letter C.

```
Examples:     C'A'
              C'+'
              C'''
```

EXPRESSIONS

Expressions are formed by combining terms from left to right using the following arithmetic operators:

```
+  Addition
-  Subtraction
*  Multiplication
/  Division
```

The first term of an expression may be preceded by a minus sign to indicate negation. An expression is absolute if the net number of relocatable terms is zero and relocatable if the net number of relocatable terms is one. Each operand address generated by a relocatable expression must be modified at load time by adding the relocation bias in order to obtain a corresponding memory address. This function is performed by the loader in addition to program storage allocation.

DIRECTIVES

Directives are special instructions to the assembler relating to assembly control and data generation. Each directive is identified by a symbolic operation mnemonic which is written in the operation field of the statement. With the exception of the EQU directive, a label appearing within the label field of an assembler directive is assigned the value and mode of the location counter prior to processing the directive.

Define Origin (ORG)

The location counter is set to the value and mode of the expression appearing in the operand field.

```
Examples:     ORG  0
              ORG  X'1000'
              ORG  *+16
              ORG  BEGIN+6
```

Define Equivalence (EQU)

The label appearing in the label field is assigned the value and mode of the expression appearing in the operand field.

```
Examples:     TEN     EQU  10
              TEMP2   EQU  TEMP+2
              CON6    EQU  X'4AF0'
```

Define Constant (DC)

The constants appearing in the operand field are evaluated and generated into consecutive storage words. Each line of constant data is identified by a single character type mnemonic followed by one or more constant items separated by commas and enclosed in apostrophes. The type mnemonic may be preceded by an optional integer repeat constant which specifies the number of line copies to be generated.

Hexadecimal, Type X

An X type constant consists of the letter X followed by one or more hexadecimal numbers separated by commas and enclosed in apostrophes. One storage word is generated for each constant item.

```
Examples:    DC    X'400000'
             DC    X'16FC,12AB6'
             DC   4X'20'
             DC  10X'0,0,FFFFFF'
```

Single Word, Type S

A type S constant consists of the letter S followed by one or more expressions separated by commas and enclosed in apostrophes. One storage word is generated for each item specified.

```
Examples:    DC    S'-4'
             DC    S'TEMP+2'
             DC  10S'0'
             DC    S'1,10,100,1000'
```

Double Word, Type D

A type D constant consists of the letter D followed by one or more double word constants separated by commas and enclosed in apostrophes. Two types of double word constants are permitted:

```
Fixed Point:     ±M
Floating Point:  ±M.N±E
```

M, N, and E are strings of decimal digits representing the integer, fraction, and exponent parts of the constant, respectively, and may be omitted when zero. Fixed point constants are distinguished from floating point constants by the absence of both a decimal point and exponent. The repeat constant, if specified, is interpreted as a signed fixed point binary scale factor. In this case, the constant is converted to its fixed point representation at the specified scale. Two storage words are generated for each constant item.

```
Examples:    DC    D'.125'
             DC    D'-.2554-3'
             DC    D'125000'
             DC  23D'45.99656'
```

ASCII, Type A

A type A constant consists of the letter A followed by a string of characters enclosed in apostrophes. The characters are generated into consecutive storage words and packed three characters per word beginning with the left byte of the first word. Additionally, a one or two digit hexadecimal number enclosed in apostrophes appearing within the string is evaluated and generated into the next available character position.

```
                 Examples:      DC     A'CHARACTER STRING'
                                DC     A'MESSAGE'OD''OA''
                                DC   20A' '
```

Character, Type C

A type C constant consists of the letter C followed by a string of characters enclosed in apostrophes. The characters are generated into consecutive storage words with one storage word allocated for each character.

```
                 Examples:      DC    C'A'
                                DC    C'ABCDE'
                                DC   4C' '
```

Define Storage (DS)

The number of words indicated by the value of the absolute operand expression are reserved in storage.

```
                 Examples:      DS   10
                                DS   X'64'
                                DS   0
```

Define Entry Symbols (ENTRY)

The symbols appearing in the operand field correspond to statement labels which may be referenced by other programs to permit symbolic interprogram linkage.

```
                 Examples:      ENTRY  FADD,FSUB
                                ENTRY  SQRT
```

Define External Symbols (EXTRN)

The symbols appearing in the operand field correspond to entry symbols in one or more external programs which are to be loaded following the current program. An external symbol may appear only as an isolated term within the operand of a memory reference instruction or type S DC directive. Evaluation of address operands generated by external symbols is performed by the loader at load time.

```
                 Examples:      EXTRN  FMUL,FDIV
                                EXTRN  SIN
```

Conditional Assembly (IFZ,IFP,IFN,IFNZ,IFNP,IFNN)

The first operand expression is evaluated and tested against the selected IF condition. If the condition is true, the number of successive statements indicated by the value of the absolute second operand expression are ignored by the assembler. If a second operand is not specified, all successive statements are ignored until an END or ENDIF directive is encountered. The permissable IF conditions are indicated as follows:

IFZ -	IF Zero (=0)
IFP -	IF Positive (>0)
IFN -	IF Negative (<0)
IFNZ -	IF Not Zero ($\neq$0)
IFNP -	IF Not Positive ($\leq$0)
IFNN -	IF Not Negative ($\geq$0)

```
Examples:     IFZ    COND
              IFNP   OPTION-1,4
```

End Conditional Assembly (ENDIF)

Assembly unconditionally proceeds with the next source statement.

Define Macro (MACRO)

A macro procedure is defined and assigned the name appearing in the label field. The first absolute operand expression specifies the number M of statements comprising the macro and the second absolute operand expression specifies the number N of macro parameters. The macro statement is followed by M statements which comprise the macro skeleton. Each statement may specify dummy macro parameters which are to be replaced by specific arguments when the macro is referenced. Each dummy parameter consists of the '#' character followed by an integer i in the range $0 \leq i \leq N$. A macro is referenced by writing the macro name in the operation field and a list of arguments separated by commas in the operand field which are to be substituted for the dummy parameters in the order: first argument-#0, second argument-#1, etc. The single parameter '#' represents the number of arguments present in the last macro reference and may be used in conjunction with conditional assembly directives th facilitate the construction of variable argument macros. A macro definition may incorporate any number of macro references; however, a macro must be defined prior to being referenced.

Example 1:   Storage to Storage MOVE Macro.

Macro Definition:

```
MOVE   MACRO   2,2
       LDA     #0
       STA     #1
```

Macro Reference:

```
       MOVE    TEMP,ARG+4
```

Generated Statements:

```
       LDA     TEMP
       STA     ARG+4
```

Example 2:   Variable Argument CALL Macro.

Macro Definition:

```
CALL   MACRO   5,3
       BS      #0
       IFN     #-2,3
       DC      S'#1'
       IFN     #-3,1
       DC      S'#2'
```

Macro Reference:

```
       CALL    SUB
```

Generated Statements:

```
       BS      SUB
```

Macro Reference:

```
       CALL    SUB,ARG1,ARG2
```

Generated Statements:

```
       BS      SUB
       DC      S'ARG1'
       DC      S'ARG2'
```

Program End (END)

The END statement indicates the end of the source program. The operand expression specifies the execution entry address of the program.

Listing Control Directives

The following assembler directives control the format of the program listing and are not printed on the list.

Program Title (TITLE)

The character string appearing in the operand field is printed at the top of each page of the listing.

New Page (PAGE)

The print device is advanced to a new page prior to printing the next statement.

Space Listing (SPACE)

The listing is spaced forward the number of lines indicated by the value of the absolute operand expression.

Suppress Listing (NOLIST)

Listing of statements following the NOLIST directive is suppressed.

Resume Listing (LIST)

Listing of statements following the LIST directive is resumed.

Suppress Generated Data Listing (NODATA)

Listing of constant data generated by DC directives is suppressed.

Resume Generated Data Listing (DATA)

Listing of DC generated constant data is resumed.

Suppress Macro Statement Listing (NOSTAT)

Listing of symbolic macro statements is suppressed.

Suppress Conditional Statement Listing (NOCOND)

Listing of conditional assembly statements and all statements which are not assembled is suppressed.

INSTRUCTIONS

Instruction statements are translated by the assembler into corresponding machine instructions. Each instruction statement is identified by a symbolic instruction mnemonic which is written in the operation field of the statement. A label appearing in the label field of an instruction statement is assigned the current value and mode of the location counter.

Class I, Memory Reference

A	Add	DM	Decrement Memory
AD	Add Double	EX	Execute
B	Branch	IM	Increment Memory
BE	Branch if Equal	LDA	Load A
BG	Branch if Greater	LDD	Load Double
BI	Branch Indirect	LDX	Load X
BL	Branch if Less	LDY	Load Y
BNE	Branch if Not Equal	LDZ	Load Z
BNG	Branch if Not Greater	M	Multiply
BNL	Branch if Not Less	N	Logical AND
BNV	Branch if No Overflow	O	Logical Inclusive OR
BR	Branch and Restore	S	Subtract
BS	Branch and Save	SD	Subtract Double
BV	Branch if Overflow	STA	Store A
CPA	Compare A	STD	Store Double
CPD	Compare Double	STX	Store X
CPX	Compare X	STY	Store Y
CPY	Compare Y	STZ	Store Z
CPZ	Compare Z	X	Logical Exclusive OR
D	Divide		

Operands:	A	Direct, No Indexing
	A,X	Indexed using Rx
	A,Y	Indexed using Ry
	A,Z	Indexed using Rz

A: An absolute or relocatable expression in the numerical range $-32768 \leq A \leq 32767$ which specifies a memory or displacement address.

Class II, Immediate

AIA	Add Immediate A	NIY	AND Immediate Y
AIX	Add Immediate X	NIZ	AND Immediate Z
AIY	Add Immediate Y	OIA	Inclusive OR Immediate A
AIZ	Add Immediate Z	OIX	Inclusive OR Immediate X
CIA	Compare Immediate A	OIY	Inclusive OR Immediate Y
CIX	Compare Immediate X	OIZ	Inclusive OR Immediate Z
CIY	Compare Immediate Y	SIA	Subtract Immediate A
CIZ	Compare Immediate Z	SIX	Subtract Immediate X
LIA	Load Immediate A	SIY	Subtract Immediate Y
LIX	Load Immediate X	SIZ	Subtract Immediate Z
LIY	Load Immediate Y	XIA	Exclusive OR Immediate A
LIZ	Load Immediate Z	XIX	Exclusive OR Immediate X
NIA	AND Immediate A	XIY	Exclusive OR Immediate Y
NIX	AND Immediate X	XIZ	Exclusive OR Immediate Z

Operands: A

A: An absolute or relocatable expression in the numerical range $-32768 \leq A \leq 32767$ which specifies an immediate operand.

Class III, Shift

SLC	Shift Left Closed	SRA	Shift Right Algebraic
SLCD	Shift Left Closed Double	SRAD	Shift Right Algebraic Double
SLL	Shift Left Logical	SRL	Shift Right Logical
SLLD	Shift Left Logical Double	SRLD	Shift Right Logical Double

Operands:

A	Direct
A,X	Indexed using Rx
A,Y	Indexed using Ry
A,Z	Indexed using Rz

A: An absolute expression in the numerical range $-32768 \leq A \leq 32767$ which specifies a shift or displacement count.

Class IV, Input-Output

RD	Read Direct	WD	Write Direct
RS	Read Status	WS	Write Status

Operands:	A	Direct
	A,X	Indexed using Rx
	A,Y	Indexed using Ry
	A,Z	Indexed using Rz

A: An absolute expression in the numerical range $-32768 \le A \le 32767$ which specifies a device or displacement address.

Class V, Register-Register and Control

CAX	Copy A to X	IAX	Interchange A and X
CAY	Copy A to Y	IAY	Interchange A and Y
CAZ	Copy A to Z	IAZ	Interchange A and Z
CXA	Copy X and A	NEGA	Negate A
CYA	Copy Y and A	NEGD	Negate Double
CZA	Copy Z and A	NEGX	Negate X
CV	Clear Overflow	NEGY	Negate Y
DSI	Disable Interrupts	NEGZ	Negate Z
ENI	Enable Interrupts	NOP	No Operation
HALT	Halt	SV	Set Overflow

Operands: None

Class VI, Supervisor Call

SVC Supervisor Call

Operands: A

A: An absolute expression in the numerical range $-32768 \le A \le 32767$ which specifies a supervisor call index.

ASSEMBLY ERRORS

Errors detected during assembly are indicated on the program lsiting by a single error flag. The error flags and their meanings are indicated as follows:

P -	Pass Error. A statement label has a calculated pass two value different from the value assigned during pass one. This condition may result from data loss during one of the source passes or as the result of a symbol not being defined prior to its reference within the operand of an ORG or DS assembler directive.
O -	Undefined Op Code. The operand field does not contain a recognized instruction or directive mnemonic.
L -	Label Field Error. The contents of the label field violate label syntax requirements.
D -	Doubly Defined Label. The label appearing in the label field has appeared elsewhere as a statement label. The value assigned at the first occurance is used.
S -	Statement Error. The contents of the operand field violate statement syntax requirements.
U -	Undefined Symbol. A symbol appearing in the operand field has not been defined in any program statement.
C -	Constant Error. An operand constant has been incorrectly specified or does not lie in the correct numerical range.
E -	Expression Error. An expression appearing in the operand field has an improper mode or does not lie in the correct numerical range.

PROGRAM LISTING

The program listing is printed during pass two with listing fields for error flag, location counter, instruction, and statement number in the indicated print positions:

1	Error Flag
3- 6	Location Counter (Hexadecimal)
8-13	Instruction (Hexadecimal)
15-18	Statement Number (Decimal)
20-	Source Statement

The listing is printed with 56 lines per page. Each page is numbered in decimal.