CONTROL DATA CORPORATION

CONTROL DATA®
CYBER 18 COMPUTER SYSTEMS

SYSTEM SUMMARY

REVISION RECORD

REVISION	DESCRIPTION
01	Preliminary edition released.
(3/76)	
02	Manual updated.
(4/76)	

Publication No.
96767850

Address comments concerning this manual to:

Control Data Corporation
Publications and Graphics Division
4455 Eastgate Mall
La Jolla, California 92037

or use Comment Sheet in the back of this manual.

CONTENTS

FIGURES

TABLES

The CYBER 18 is a series of general-purpose micro-programmable processors designed for application to a broad range of requirements. One basic hardware design, made up of modular elements, serves a variety of functions, permitting a high degree of standardization.

The operation of the CYBER 18 is controlled by a "micro" program in semiconductor memory, referred to as micro memory. The micro program reads macro instructions from main memory and decodes them for execution in the micro processor. The semiconductor memory, which is several times faster than core memory, executes several micro instructions during a single main memory cycle. In the CYBER 18, special micro-programming techniques are used to emulate an enhanced CDC 1700 system for lower cost, smaller size, and equal or better speed.

Micro programming provides a number of long-range benefits. In contrast to the conventional computer, whose control is an irregular assembly of specially designed logic functions, the micro-programmed machine embeds the control logic in a high-speed read-only memory, reducing the amount of irregular logic in a processor and allowing greater use of MSI/LSI circuitry. This approach allows high-volume production of nearly identical processors that can be tuned to a specific application by including a selected instruction set in the micro memory, thus specifying a different virtual machine. The micro-programming technology provides the flexibility to offer new architectures for general-purpose computers with emulation capabilities and for controllers where emulation is not an important factor.

Growth is a most important feature of the CYBER 18. Growth is available via the micro-code expansion capability of the micro processor. New functions can be added as applications require, or operating system functions can be included as part of the micro sequence to improve overall system performance. This growth potential is available without sacrifice of hardware commonality, so a common logistic base is maintained.

Figure 1-1 presents an overall view of the CYBER 18 hardware and software products.

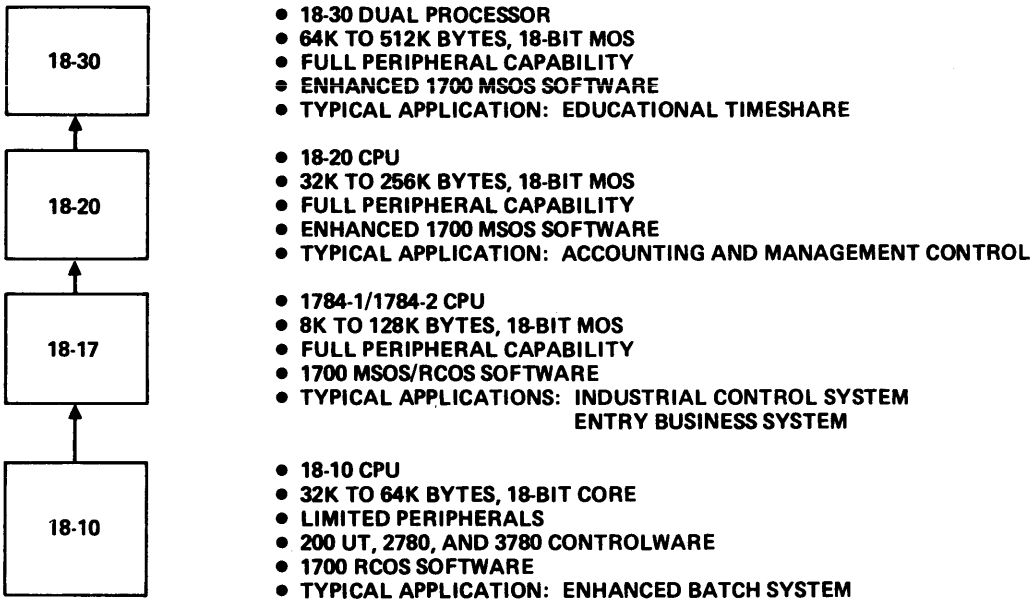


Figure 1-1. CYBER 18 Series Product Family

TABLE 1-1. CYBER 18 PRODUCT LINE GENERAL CHARACTERISTICS

Basic Configuration	
Processor	
Type	General-purpose, micro-programmable digital processor
Organization	Register oriented or file oriented.
Word length	16 bits
Micro-instruction word	32-bit format; two micro instructions per micro-memory address
Micro-memory type	Semiconductor read/write memory (RAM) and/or read-only memory (ROM)
Micro-memory size	512 words in 64-bit increments (on transform); maximum of 4,096 additional words available.
Micro-memory access time	70 nanoseconds
Arithmetic	Binary with dynamic selection of ones or twos complement mode Up to four parallel unrelated operations are possible in one micro instruction
Macro Memory	
Requirement	Variable, according to application
Type	Core memory: available in 16K byte stacks, with a maximum of 64K bytes (CYBER 18-10) MOS memory: available in 32K or 64K byte stacks, with a maximum of 256K bytes (CYBER 18-20) or 512K bytes (CYBER 18-30) Parity and protect bits are available in the standard stack.
Memory speed	750 nanoseconds average cycle
Input/Output (I/O)	
Interfaces	Display terminal (RS232-C compatible)
Mechanical	
Construction	RETMA 19-inch, rack mountable
Dimensions	Logic Chassis: Height — 18.5 inches (47 cm) Width — 17.5 inches (44.5 cm) Depth — 16.0 inches (40.64 cm) Power Supply Chassis: Height — 8.75 inches (22.25 cm) Width — 17.5 inches (44.5 cm) Depth — 16.0 inches (40.64 cm)
Weight	Logic Chassis: 40 pounds (approximately) (18 kg) Power Supply: 50 pounds (approximately) (45 kg)

TABLE 1-1. CYBER 18 PRODUCT LINE GENERAL CHARACTERISTICS (Continued)

Basic Configuration (Continued)	
Mechanical (Continued)	
Input power	115 volts, 50/60 Hz
Miscellaneous Features	
	Real-time clock Auto-data transfer Direct memory access (CYBER 18-20, 18-30)

The hardware of the CYBER 18 product set is complemented with an extensive set of software products, comprising operating systems (including program debugging and development aids), micro and macro assemblers, application-oriented software, and FORTRAN, BASIC, RPG II, and PASCAL compilers.

OPERATING SYSTEMS

MASS STORAGE OPERATING SYSTEM (MSOS 5)

This operating system is the most recent in the development of a sequence of mass-storage-based operating systems for the 1700, 1714, 1774, and 1784 computers. It takes advantage of the enhanced instructions, additional registers, and firmware capabilities of the CYBER 18 Series of computers. MSOS 5 is designed for use on the CYBER 18-20, having random-access mass memory.

The component parts of MSOS 5 are shown in figure 2-1.

MSOS 5 is a multiprogramming system designed to support a variety of applications requiring dedicated system utilization, batch processing, and program checkout features in a real-time environment.

MSOS 5 regulates all multiprogramming on the basis of the priority level assigned to a particular operation, whether the operation is program execution or input/output. The system queues requests for I/O data transmission and program execution by priority level, with no restriction on the number of requests that may be queued at a given time. The program task with the highest priority level is selected for execution. It remains in execution until completed, unless a higher priority level interrupt is scheduled. The lower priority level task is then suspended until the higher priority interrupt task is completed.

On interrupt, all eight registers of the CPU may be stored by a single command, making possible a very rapid change in the program environment as a result of interrupt.

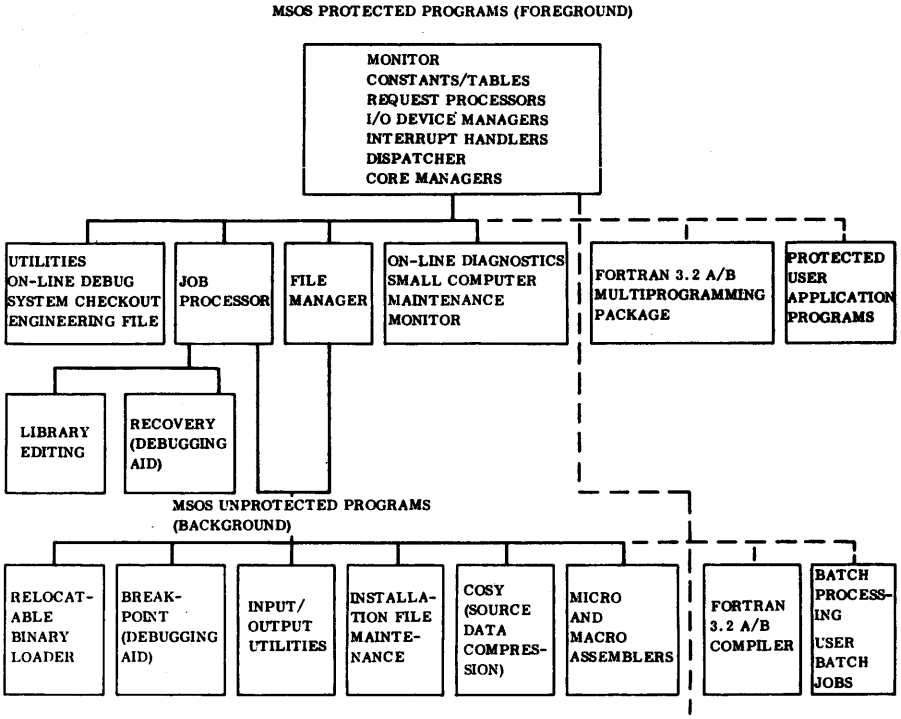


Figure 2-1. CYBER 18 Mass Storage Operating System (MSOS 5)

Sixteen hardware interrupts are utilized to maximize input/output efficiency and to allow concurrent input/output and computation.

The program protect feature of the hardware segregates central memory into two functional entities — protected memory and unprotected memory.

- Protected memory (the foreground) is reserved for executing all parts of the operating monitor and the user's on-line high priority applications programs.
- Unprotected memory (the background) is used for execution of batch job processing and program checkout at low priority levels. MSOS 5 has the capability to swap (move) the contents of unprotected memory to mass storage and to make the area protected memory for use by foreground programs.

MSOS 5 is extremely modular in design and provides the user considerable flexibility to perform on-line system modification and update.

The MSOS 5 system features the following main capabilities:

- The monitor is the real-time executive for MSOS 5. It serves as the interface between the system programs and the peripheral hardware via input/output drivers. It is highly modular and may be parameterized for a large variety of hardware and software configurations. It contains request processors to allow for input/output, program scheduling, memory allocation, and background operation. MSOS 5 provides for the execution of instructions in 128K bytes of contiguous memory, with the ability to access data in up to 512K bytes of memory.
- The job processor is responsible for monitoring background programs running in unprotected memory. Interface is provided for batch stream unattended jobs or for interactive operator-controlled jobs. The job processor controls the FORTRAN compiler, the macro assembler, and numerous background utility functions.
- The file manager is a general-purpose file management package. It creates and maintains both sequential and indexed files, and offers sequential, indexed, and direct methods of record retrieval, as well as variations of these. The file manager may be used by protected and unprotected programs and is FORTRAN-callable.
- The on-line diagnostics, in an on-line, real-time mode, have the capability of running peripheral hardware diagnostics, which facilitates problem detection and preventive maintenance.

The minimum hardware CYBER 18-20 requirements for MSOS 5 are:

- Computer with 32K bytes main memory
- Console device (teletypewriter, CRT)
- Input device (paper tape, cards, magnetic tape)
- Output device (paper tape, cards, magnetic tape)
- 1,024,000 bytes mass storage memory

MINIMUM OPERATING SYSTEM (RTOS 3)

The minimum operating system for the CYBER 18-10 is the Real-Time Operating System Version 3 (RTOS 3). It provides executive software and appropriate development tools in a minimum of main memory, without dependence on random-access mass memory.

RTOS 3 includes a monitor that is a compatible subset of MSOS 5. The monitor requires less than 3000 bytes of main memory, exclusive of peripheral-device drivers and optional features. Up to 16 program priority levels are provided, and input/output requests are also processed on a priority basis. The monitor is itself reentrant and minimizes the time for which interrupts are inhibited so as to provide very fast interrupt response times. It contains a request processor to handle the following standard requests:

READ
WRITE
FREAD
FWRITE
SCHDLE
INDIR
EXIT

Optional request processors are available to process requests for:

TIMER
MOTION
SPACE
RELEASE
CORE
STATUS
EBCDIC

Other optional features include a protect processor for use while debugging user-written programs, a core allocator, an alternate device handler and a diagnostic timer.

All MSOS 5 drivers are compatible with RTOS 3.

An RTOS 3 system with a minimum of 16K bytes of main memory supports the Assembler, which is compatible with the MSOS assembler (except that macros may not be used). Previously compiled FORTRAN programs may be run under RTOS 3; however, FORTRAN programs may not be compiled under RTOS 3.

A job processor provides for loading (*L) and executing (*X) non-resident absolute programs such as the relocatable binary loader and the assembler. An optional job processor provides expanded load and execute features (*L, *X), schedule (*S), and terminate (*Z).

Optional modules allow for batch mode control (*U, *V), dumping core (*D), inserting values into core (*I), punching core (*P), magnetic tape motion (*REW, etc.), assigning logical units (*K), marking logical units up/down (*M, *N), and copying and/or converting data (*T). The optional job processor modules may be either core-resident or system library-resident.

The system library resides on magnetic tape and contains system maintenance, debug, and utility programs. This includes programs in the RTOS 3 product set (relocatable loader, assembler, system initializer, LIBEDT, SMART, SETUP), all optional job processor modules, and the Small Computer Maintenance Monitor (SCMM).

RTOS 3 features an optional processor to load any of the programs from the system library by file name.

The library editor (LIBEDT) provides for maintenance of the system library labelled and unlabelled files in absolute or relocatable binary form on magnetic tape. A source tape editor (SETUP) and a relocatable binary tape editor (SMART) are also provided.

SCMM under RTOS 3 is functionally identical to SCMM under MSOS 5. The same tests and drivers are used in both systems.

RTOS 3 may be stored on magnetic tape or punched cards and installed or, if necessary, reloaded via a bootstrap loader.

The minimum hardware requirements for RTOS 3 are:

- CYBER 18-10 with 16K bytes of main memory
- Console device (conversational display terminal)
- Input device (card reader, magnetic tape)
- Output device (magnetic tape) — optional

LANGUAGES

MACRO ASSEMBLER

The macro assembler for the CYBER 18 Series provides a comprehensive instruction set that makes full use of the CPU capabilities and includes the ability to define, execute, and maintain a library of macro commands. A minimum system for execution of the macro assembler (under RTOS 3) requires 16K bytes of main memory.

CROSS ASSEMBLER (CLASS)

The CLASS cross assembler is designed for use on CYBER 170/70/6000/7000 computers. It duplicates all the functions of the CYBER 18 macro assembler, producing listing output and relocatable binary output for input by the CYBER 18 relocatable binary loader.

MICRO ASSEMBLER

The micro assembler provides the ability for the user to develop his own micro program using appropriate alphabetic and numeric designations of operations to be performed. As with the macro assembler, the micro assembler exists in two forms: one for micro assembly on a CYBER 18 Series machine, and the other for cross micro assembly on large CYBER Series computers (CYBER 170/70/6000/7000). Both forms of micro assemblers output in binary relocatable format, compatible with the loader of the MSOS 5 or RTOS 3 operating system.

A minimum system for execution of the micro assembler under MSOS 5 on a CYBER 18-20 requires 64K bytes of main memory.

FORTRAN

There are two FORTRAN compilers for the CYBER 18 Series. They provide the same run-time capability but differ in compilation speed and main memory requirements. The essential components and modes of operation are shown in figure 2-2.

The language is a subset of ANSI FORTRAN, so, most ANSI FORTRAN programs can be compiled with few changes; basic FORTRAN programs are immediately compiled.

As a product-set member of the CYBER 18 MSOS 5, Mass Storage FORTRAN offers a variety of compile-time options (such as run-anywhere code) and run-time packages.

CYBER 18 Mass Storage FORTRAN is a high-level language and, as such, is concise and provides data structuring capabilities and I/O independence. Increased program modularity and source code clarity also results.

The wide range of FORTRAN system configurations allows optimum usage of available storage resources for compilation and foreground/background execution.

Five basic elements comprise 1700 Mass Storage FORTRAN Version 3 — the A and B variations of the compiler and three run-time packages: re-entrant ENCODE/DECODE, non-re-entrant ENCODE/DECODE, and FORTRAN I/O.

The A variant compiler, having a larger number of overlays than the B variant, requires more compilation time but less core memory (16K vs. 32K bytes). The source format for both variants is the same, while the object code produced by the two variants is nearly identical.

The re-entrant ENCODE/DECODE run-time package runs in the foreground (protected core) and is the FORTRAN multiprogramming interface. It allows FORTRAN routines to execute at multiple levels without interference, and provides a greatly enhanced interface capability to the MSOS monitor.

Although the formatted I/O is nonstandard, all I/O capabilities are provided through the monitor and ENCODE/DECODE package calls.

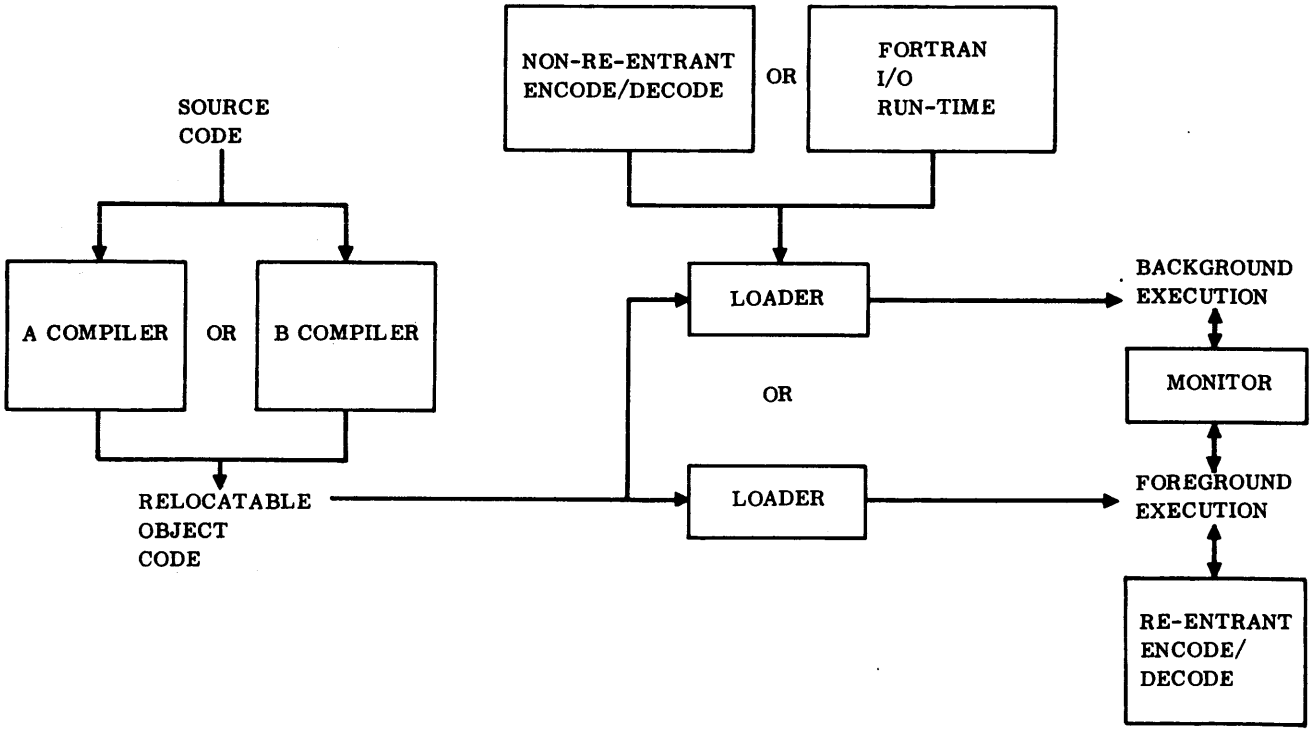


Figure 2-2. CYBER 18 Mass Storage FORTRAN

Given a list of variables and a FORMAT statement, ENCODE and DECODE translate the variables between internal hexadecimal format and ASCII. The other formatting routines do not use FORMAT statements, but translate either integer or floating-point variables. These formatting routines are as follows:

- HEXASC — Hexadecimal to ASCII
- HEXDEC — Hexadecimal to decimal ASCII
- ASCII — ASCII to hexadecimal
- DECHEX — Decimal ASCII to hexadecimal
- AFORM — Packed ASCII to unpacked ASCII (left-justified blank-filled)
- RFORM — Packed ASCII to individual ASCII characters (right justified, zero filled)
- FLOATG — Floating-point variable to ASCII

Monitor requests that are provided include Read/Write (extensive I/O capabilities), Scheduler/Timer (causes execution of another routine, optionally, after a specified time interval). Dispatcher (discontinue execution without terminating), and Release (return core allocated for execution).

10K bytes are required for the foreground ENCODE/DECODE package. If double-precision arithmetic is used, an additional 5K bytes are required.

Foreground programs can be checked out in the background (unprotected core) via the (identical) interface provided by the non-re-entrant ENCODE/DECODE package, without interfering with the protected programs, including the monitor.

The FORTRAN I/O run-time package, which also runs in the background, provides standard FORTRAN I/O capabilities more extensive than those of the ENCODE/DECODE packages.

The efficiency of the produced object code is mainly due to the compiler's extensive optimizing features, which include:

- Common arithmetic subexpressions, including subscripts, are recognized within and between expressions and are computed only once
- Index registers are optimally assigned
- One-word relative addressing is used wherever possible
- Relative addressing is maximized by appropriate storage allocation: some constants, for instance, are duplicated, and some arrays are inserted between code segments
- Simple FORTRAN functions such as AND and IABS are provided in-line
- IF statements are extensively analyzed, such as a transfer made to the next statement or to a GO TO
- Expressions in logical IFs are analyzed for the method of least computation
- Arithmetic expressions are analyzed to minimize code production and execution time
- Division by a real constant generates code for multiplication by the constant's reciprocal
- Integer multiplication or division by a (constant) power of 2 is accomplished by shifting

The following standard statements are recognizable by the FORTRAN compiler:

ASSIGN	FUNCTION
BACKSPACE	GO TO
BLOCK DATA	INTEGER
CALL	IF
COMMON	OPEN
CONTINUE	PAUSE
DIMENSION	PROGRAM
DO	READ
DOUBLE PRECISION	REAL
END	RETURN
ENDFILE	REWIND
EQUIVALENCE	SIGN
EXTERNAL	SUBROUTINE
FORMAT	STOP
	WRITE

Extensions to the ANSI standard are as follows:

- ASSEM — Allows inclusion of assembler statements in FORTRAN source code
- BYTE — Assigns a name to a bit field within an integer variable or array

- DATA — Enhanced to allow use of implied DO-loop to initialize arrays
- RELATIVE — Specifies externals to be linked via relative addressing
- SIGNED BYTE — Identical to BYTE but the high-order bit of the field is interpreted as a sign bit

The minimum main memory requirement for execution of the compiler (A or B version) on a CYBER 18-20 is 64K bytes.

PASCAL

PASCAL is a compiler patterned after ALGOL 60. It is available as a cross compiler executing on large CYBER computers (170/70/6000/7000), producing relocatable binary output for loading and execution by a CYBER 18 machine.

PASCAL is a high-level, algorithmic type language retaining the attractive features of ALGOL 60. Thus, the grammar of PASCAL is essentially context-free. Its syntax is unambiguous and simple to define, properties that are greatly appreciated by programmers. The block-oriented structure of ALGOL 60, which is particularly adaptable to the use of structured programming techniques, has also been preserved.

The basic constituents of a PASCAL program are statements and declarations or definitions. Statements indicate the various actions that are to be carried out by a program, and declarations/definitions describe the meaning attached to the various identifiers used in a program.

PASCAL provides some noteworthy extensions to the capabilities of ALGOL 60. Within the domain of statement constructs, PASCAL supports a broad variety of structured statements. Thus, repetitive or conditional actions may be coded in a concise and natural manner. More significantly, the ALGOL 60 deficit in the area of structured data has been remedied by the introduction of a set of data structuring techniques. Another feature of PASCAL is the provision of pointer-type variables, along with the ability to dynamically and explicitly allocate storage. These features extend the range of applicability of PASCAL, as compared with that of its more traditional ancestor.

PASCAL programs compiled on a CYBER 170/70/6000/7000 class of machine may be executed in a CYBER 18-10 under RTOS 3 with a minimum memory of 16K bytes.

BASIC

The BASIC compiler provided as a component of the CYBER 18 software product set meets the requirements of the proposed ANSI standard for minimal BASIC (ANSI -XJ32), and provides the following additional capabilities:

- Additional string manipulation functions
- Matrix operations
- Input/output formatting
- Binary and ASCII files
- Sequential and random (indexed) file accessing

The BASIC compiler may be executed interactively under TIMESHARE 3 on a CYBER 18-30, or in batch mode under MSOS 5 on a CYBER 18-30 or 18-20. Code generated by the BASIC compiler is usually executed under the TIMESHARE 3 system; if execution is performed under MSOS 5 in a CYBER 18-20 a minimum main memory of 96K bytes is required.

RPG II

The Report Program Generator II (RPG II) compiler provided as a member of the CYBER 18 software product set is closely compatible with IBM System 3/RPG II at the source language level. Differences from the System 3/RPG II primarily reflect differences in peripheral characteristics; the multifunction card reader, the dual tractor printer, bisync communication, card reader stacker selection, and inquiry mode are not supported by CYBER 18 RPG II.

RPG II provides a simple language by which a user can:

- Easily describe the formats, sources of input data, and destinations of output data
- Execute simple procedural programs with both conditional and unconditional statements
- Describe the formats and sources of data in an external file system

CYBER 18 RPG II requires a CYBER 18-20 for compilation under MSOS 5 but, depending on program size, may be executed on a CYBER 18-10 with 32K bytes of memory.

TIMESHARE 3

The TIMESHARE 3 system is designed to provide interactive computer capability for up to 64 concurrent users, together with communication with a remote large-scale CYBER system (70/170/6000/7000). TIMESHARE 3 requires the dual processor of the CYBER 18-30, one processor handling user programs, the other providing communication with a remote host machine and I/O for local peripherals. TIMESHARE 3 operates under MSOS 5.

The capabilities and characteristics of TIMESHARE 3 include:

- Up to 64 concurrent users on dedicated or dial-up lines
- Interactive execution of the BASIC compiler, with line-by-line syntax checking
- Log-in and sign-off procedures, with password security check
- A text editor for manipulation of user program and data files
- Simulation of a desk calculator, providing arithmetic operations with a minimum accuracy of six significant digits
- A formula calculator, where the user enters the algebraic formula and, under prompting by the computer, the values of independent variables
- A log of system events and file of system messages available to all terminal users
- Utility programs for use by the CYBER 18-30 operator for starting, stopping, and determining the status of the system, transmitting a message to all user terminals, establishing passwords, dumping, reloading, and auditing files.
- Ability to generate batch jobs at a remote terminal, for execution either by the CYBER 18-30 in batch mode or by the host processor (CYBER 170/70/6000/7000).

Minimum hardware requirements for TIMESHARE 3 are:

- CYBER 18-30, the timeshare processor with 128K bytes and the communications processor with 64K bytes of main memory
- Cassette tapes on both CPUs
- Storage module with 25M byte capacity
- Nine-track tape, card reader, and line printer on the timeshare CPU
- Synchronous communication interface for host computer link (optional)
- Asynchronous communications multiplexer, with facilities for both dial-up and dedicated lines
- Teletypewriter-compatible terminals (optional)
- Asynchronous communication line adapters (optional)

MANUFACTURING INDUSTRY ACCOUNTING SYSTEM

This application software for the CYBER 18, available in 1977, is designed to provide a broad range of business data processing capabilities for manufacturing and distribution industries. The application software is supplied in modular form, facilitating the matching of system capabilities to a particular customer's requirements. The modules of the system are:

- Order entry and invoice preparation
- Purchase orders and accounts payable
- Production and inventory control
- Inventory status and valuation
- Accounts receivable
- Payroll
- General ledger
- Sales analysis

The Manufacturing Industry Accounting System operates on a CYBER 18-20, supporting multiple local terminals operating on different tasks concurrently.

Minimum hardware requirements for the system are:

- CYBER 18-20 processor with 64K bytes of main memory (supporting two terminals), 96K bytes for up to six terminals, and 128K bytes for more than six terminals.
- Storage module with 25M bytes of random access mass storage
- Line printer
- Terminals with optional hard copy capability

BATCH TERMINAL CONTROLWARE

The CYBER 18-10 is available in the form of a batch terminal, with the ability to emulate a CDC 200 User Terminal or an IBM 2780 or 3780 terminal. This emulation is accomplished by software supplied in absolute binary form on card decks. A different deck is used for each emulation. The software is not intended to be accessible to the user for maintenance; the use of the term controlware reflects the "black-box" nature of the software.

With 32K bytes of memory and a magnetic tape unit, the 18-10 permits, in addition to batch terminal functions, local user program development and execution under RTOS 3.

The functions provided by the emulated terminals are described below.

CDC 200 USER TERMINAL EMULATION

The CDC 200 User Terminal provides the following features:

- Operator selection of Hollerith 026 or 029 card formats.
- CDC 200 User Terminal synchronous communication techniques, using modems conforming to RS232-C or CCITT-V24 specifications at rates from 1200 to 9600 baud.
- Half-duplex communication with a host computer, using a dedicated or dial-up line
- Compatibility with CDC CYBER 72, 73, 74, and 6000 Series computers using KRONOS with EXPORT/IMPORT or SCOPE with INTERCOM.
- Compatibility with CDC 3000 Series computers using MASTER with RESPOND-EXPORT/IMPORT, MARS III, or MCS III
- Operator selection of transmission-line code — external or internal BCD or ASCII
- Display of operator-selected operating parameters on the CRT
- Automatic notification of line and terminal error conditions on the CRT
- Selection of input and output devices (keyboard, magnetic tape, card reader, or line printer)
- Off-line functions, providing for transfer of data between peripherals on the terminal

A selection card loaded with the controlware permits matching the controlware to the hardware configuration and to the operating modes required.

Diagnostics are provided through the Operational Diagnostic System (ODS), loaded as card decks distinct from the controlware.

The minimum hardware configuration is:

- CYBER 18-10 with 16K bytes of memory (32K bytes if a tape unit is supported)
- Card reader
- Line printer
- CRT display and keyboard
- Operator panel
- Magnetic tapes (optional)

IBM 2780 EMULATION

The IBM 2780 emulation in the batch controlware provides:

- Compatibility with IBM host computers capable of communicating with 2780 terminals

- Emulation of IBM bisync communication techniques, with equipment and capabilities similar to those of the emulated 200 User Terminal.
- Operator selection of EBCDIC or ASCII code, with transparent text mode
- Selection of number (up to 30) of retransmit attempts
- Selection of print-line length and of use of end-of-record feature to eliminate trailing blanks on card
- CRT display of emulated IBM 2780 status functions
- ASA vertical forms control
- Off-line processing similar to that provided under the 200 User Terminal emulation.

IBM 3780 EMULATION

The batch terminal controlware provides capabilities similar to those of the IBM 2780 emulation, with the addition of:

- Compatibility with IBM's Telecommunications Access Method (TCAM) under OS, and with BTAM under OS or DOS.
- Interruption of terminal operation by host CPU
- Vertical and horizontal formatting of printed data
- Restart of terminal by host CPU

MODEL DESCRIPTIONS

Following are brief descriptions of the standard units. Descriptions of the optional peripherals are included in section 6, Standard Peripherals.

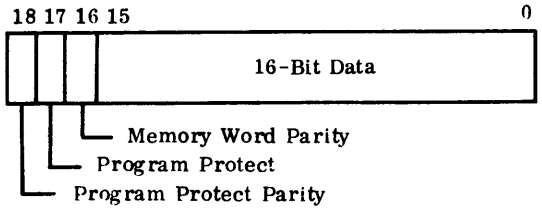
CYBER 18-10 Processor

The CYBER 18-10 processor is a micro-programmable processor unit that can accommodate up to 65,536 eight-bit bytes (one protect and one parity bit for each two bytes) of core main memory with an effective read/write cycle time of 750 nanoseconds. The 18-10 emulates the basic as well as an enhanced 1700 instruction set. Basic features are: hardware multiply/divide, 16 levels each of micro and macro interrupts, auto-data transfer, a real-time clock, and deadstart facility from peripherals. The processor supports only A/Q-type peripherals. Feature options include a programmable micro/macro breakpoint controller and breakpoint panel. The 18-10 includes a cabinet, an operators panel, power supplies, and an I/O controller for the 1810-1 communications console. No main memory is included, but 32K eight-bit bytes of core memory is required as a minimum.

1882-8 Core Main Memory Storage

The 1882-8 Core Main Memory Storage provides 16,384 eight-bit bytes of read/write core memory. One protect and one parity bit are provided for each two bytes. The effective read/write cycle time is 750 nanoseconds. The core memory occupies one memory position within the processor unit.

The memory word format is:



CENTRAL PROCESSORS

The CYBER 18-10 central processing unit (CPU) configuration consists of an ALU card, a status mode interrupt (SMI), two control cards, the I/O-TTY card and the standard CYBER 18-10 transform card. The detailed organization of the CYBER 18-10 is shown in figure 4-1. This diagram shows registers interconnected primarily by selectors. A selector is a multiplexer that transfers one of several inputs to an output. They are either one, eight, 12, 16, or 32 bits wide.

ARITHMETIC/LOGICAL UNIT (ALU) AND DATA TRANSFER ORGANIZATION

The ALU provides the arithmetic and logical capabilities of the CPU. This unit combines two 16-bit input words according to the function code specified in the micro instruction. The result is immediately available at the output of the ALU for possible shifting via selector S3 and delivery to the destination register, memory interface, panel interface, and I/O. The unshifted output of the ALU is delivered to the status mode and mask registers. The ALU output can be ignored on an operation. The results of the ALU operation regarding sign, zero, and magnitude (by means of carryout test) are available to the test bit logic for instruction sequencing.

The data transfer organization of the CYBER 18-10 provides for storing data in one of six working registers and two files, and for selecting data for processing through the ALU. ALU results are transferred back to one of the registers or out of the organization to control external equipment.

The primary data registers are I, P, A, F, X, and Q.

The following are brief descriptions of the primary registers.

- I Register — A 16-bit register whose only input and output is the selector S1. This register should not

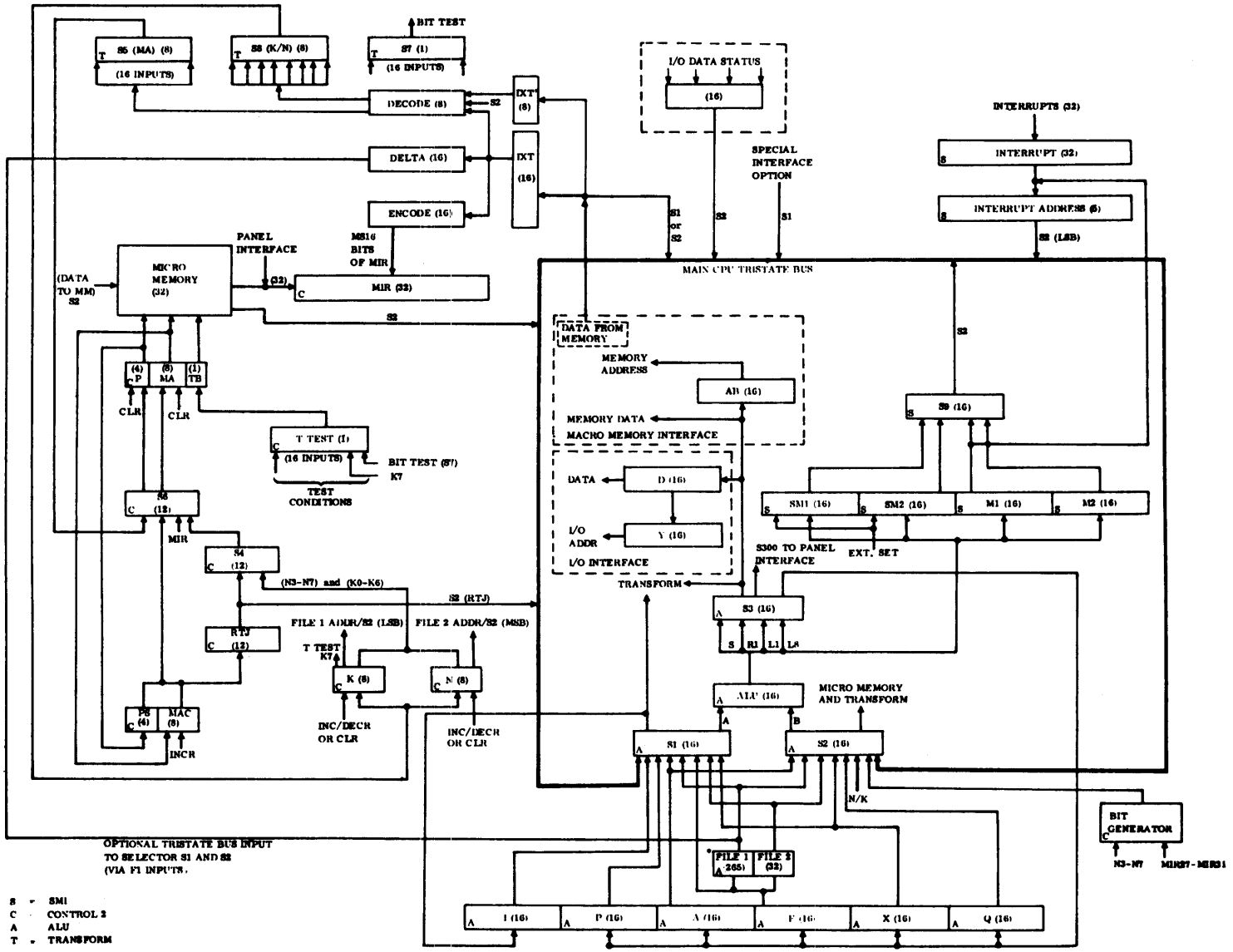


Figure 4-1. Detailed Block Diagram of CYBER 18-10 Processor

be confused with the 1700 I register (location $00FF_{16}$).

- P Register† — A 16-bit, general-purpose register that receives data from the ALU and provides output to S1. Normally it is used to hold the software instruction counter.
- A Register† — A 16-bit, general-purpose register that receives data from the ALU and provides output to S1. The A register is mechanized as a shifting register, and can be shifted left or right without using the ALU. The A register may also be combined with the Q register to form a double-length shifting register that operates independently of the ALU.
- F Register — A 16-bit, general-purpose register that receives data from the ALU and provides data to S1 or S2 as ALU input. This register is also used as the file entry register and contains information written into the files when they are used as the destination of an ALU operation.
- X Register — A 16-bit, general-purpose register that receives data from the ALU and provides data to S1 or S2.
- Q Register† — A 16-bit, general-purpose register that receives data from the ALU and provides output to S2. The Q register is mechanized as a shifting register. It may be shifted left or right in conjunction with the A register without using the ALU.

Other major portions of the standard CYBER 18-10 are:

- File 2 — A 32-word scratchpad file that may be used as a general-purpose, word-sized register. It delivers its output to S1 and S2; data input is provided by the F register. File 2 is reserved for the emulator, except for registers R1, R2, R3, and R4, which are available to the 1700 programmer through enhanced instructions.
- Bit Generator (BG) — The BG circuit generates one bit at any position in a word as input to the B side of the ALU. Control to drive the bit generator is derived from either the micro instruction (bits 27 to 31) or the lower five bits of the N register. Control is usually obtained from the micro instruction. A bit setting in an SM register determines the input that will drive the bit generator.

†Available to the 1700 programmer.

- Status/Mode Register (SM) — The SM register allows the micro program to control the mode of operation and also allows the micro program to examine the status of certain internal and external conditions. The CPU can access one of two SM registers, SM1 and SM2.

  The SM register module contains 16 bits of SM1 and 16 bits of SM2. All 32 bits of an SM module can be set or reset by the micro program by transferring information to the SM register from the output of the ALU. Master clear will also clear SM1 and SM2.
- Interrupts and Mask Register — The interrupt system is implemented as a sampled data system at the micro-program level, instead of a true vectored interrupt system as used in conventional computers.

  The mask register enables the micro processor to disable/enable interrupts. The CPU can access two mask registers, M1 or M2. For each mask bit there is a corresponding bit in the interrupt register.

  M1 is available to the 1700 programmer through the DMI instruction, while M2 (referred to as M) is available through the basic inter-register instruction.

  Interrupts are identified by their corresponding mask bits, which are assigned to control the interrupt recognition. The bits in the mask registers are identified as follows:

  -Mask Register 1 (M1): M100 through M115

  -Mask Register 2 (M2): M200 through M215

  Interrupt addresses are generated by the interrupt address encoder, according to the assignments given in table 4-1.

  The interrupt priorities correspond to the interrupt address generated; that is, interrupt address 00 is associated with the highest priority interrupt line and interrupt address 31 is associated with the lowest priority interrupt line. For example, an interrupt associated with M112 would have priority over an interrupt associated with M111, and an interrupt address of 3 would be developed by the interrupt address encoder.
- K Register — An eight-bit counter that can be cleared, incremented, or decremented. It is used to address file 1 in addition to any program usage as a counter.

TABLE 4-1. MASK REGISTER/INTERRUPT ADDRESSES

Mask Bit	Interrupt Address Mask Register 1	
M100	15	Lowest Priority (M1)
M101	14	
M102	13	
M103	12	
M104	11	
M105	10	
M106	09	
M107	08	
M108	07	
M109	06	
M110	05	
M111	04	
M112	03	
M113	02	
M114	01	
M115	00	Highest Priority (M1)
Mask Bit	**Interrupt Address Mask Register 2**	
M200	31	Lowest Priority (M2)
M201	30	
M202	29	
M203	28	
M204	27	
M205	26	
M206	25	
M207	24	
M208	23	
M209	22	
M210	21	
M211	20	
M212	19	
M213	18	
M214	17	
M215	16	Highest Priority (M2)

Note: The interrupt address generated is the same as its priority level; i.e., the highest priority interrupt generates a 0 interrupt address and the lowest priority interrupt generates a 31 interrupt address.

- N Register — An eight-bit counter that may be cleared, incremented, or decremented. It is used to address file 2, control shifts, control the scale operations, and may be used as an iteration counter that controls micro-instruction execution.
- N/K Register — The N and K registers may be combined to provide operand addresses outside the current operating micro page.

TRANSFORMS

Transforms enable quick and efficient decoding of an emulated instruction. A transform can be designed to extract bits from a register or registers, shift the bits to the required position, and add a base address or constant bits. This result can then be transferred to the micro-memory address register (transform jump) or to the K or N register (transform register load). For example, when a 1700 instruction is read from macro memory, one micro-instruction transform jump transfers control to one of 108 micro-memory locations. Without the transform feature the above operation would require many micro instructions.

The transform hardware is packaged on a separate printed circuit card and is implemented using three selectors. The transform module includes 1,024 micro instructions (512 words) in read-only memory. The majority of these instructions are used to execute the 1700 emulator. The ROM also contains instructions for the panel interface simulation via the I/O-TTY board.

I/O-TTY MODULE

The I/O-TTY module includes the following components:

- Real-Time Clock — In conjunction with the micro code it appears as a 1700 peripheral to the macro-level programmer.
- I/O Teletypewriter/Display Control — This controller is an integral part of the module. It interfaces to Control Data RS232-C-compatible conversational display terminals.

- Internal Peripheral Controller Bus — Provides all I/O data lines, interrupts, and control signals necessary to generate, in conjunction with the micro code, an internal CDC 1700 A/Q (input/output) bus. This TTL-level bus is intended to interface with controllers located in the basic chassis.
- Panel Interface Simulation — A logic section that is required when a panel/program device is used for operator input in the panel mode.

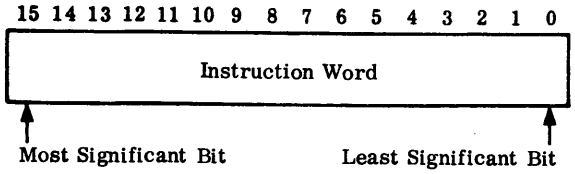
The CPU is interfaced to the I/O module as follows:

- ALU Output — All output data and address information is provided from the output of the ALU via S3.
- SM Register — All commands to peripheral controllers are generated by micro code manipulation of the status mode register.
- CPU Control — Timing and control information for controlling internal I/O module data gating is provided from the CPU control signals.

CYBER 18-10 INSTRUCTION DESCRIPTION

INSTRUCTION FORMAT

The CYBER 18-10 computer instruction word shown in the following example consists of 16 bits, numbered right to left as 0 to 15, with the leftmost bit, 15 being the most significant and the rightmost bit, 0, being the least significant.



Hexadecimal (base 16) notation is used in this computer.

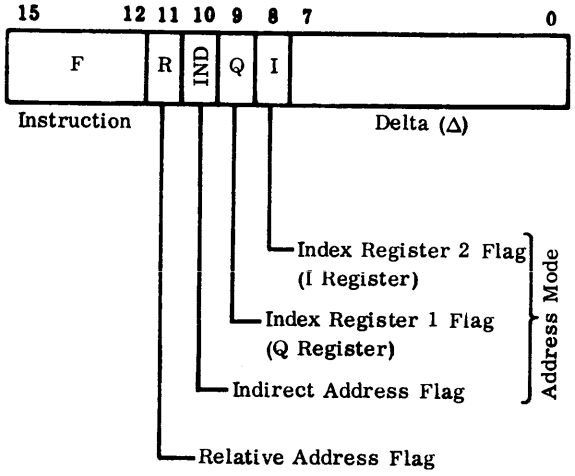
The computer is composed of a basic and an enhanced instruction set. The basic set is 1700-compatible and is divided into storage reference, register reference, inter-register, skip, and shift instructions. The enhanced instruction set is divided into the enhanced storage reference, field reference, enhanced inter-register, enhanced skip, decrement and repeat, and miscellaneous instructions.

BASIC INSTRUCTION SET

Storage Reference

The storage reference instructions shown in the following illustration contain three fields: instruction, address mode, and delta. The instruction field contains the operation code.

The address mode field contains flags for indexing, indirect addressing, and relative addressing. The delta field is a signed eight-bit address modifier in which the most significant bit is the sign bit. Storage reference instructions have the following format:



Five types of addresses and/or address methods are created by these instructions:

- Instruction Address — The address of the instruction being executed; also called P
- Indirect Address — A storage address that contains an address rather than an operand

- Base Address — The operand address after all indirect addressing but before modification by the index registers. The base address is the effective address when no indexing is specified.
- Effective Address — The final address of the operand. At certain times the effective address equals the operand for read-operand type instructions (refer to table 4-2).
- Indexing — The computer has two index registers. Index register 1 is the Q register; index register 2 is storage location $00FF_{16}$ (I register). The base address may be modified by either or both of the index registers. If the index 1 flag is set, the contents of the Q register are added to the base address to form the effective address. If the index register 2 flag is set, the contents of storage location $00FF_{16}$ (I register) are added to the base address to form the effective address. If both index register flags are set, the contents of Q are added to the base address; then the contents of $00FF_{16}$ are added to the result to form the effective address. B (for both) is used for indexing both the Q and I register. Indexing occurs after completion of indirect addressing.

The computer uses the 16-bit ones complement adder during indexing operations. Consequently, the index register contents are treated as signed quantities (bit 15 is the sign bit).

The storage reference instructions (refer to table 4-2) have eight different types of addressing modes: eight-bit absolute, eight-bit absolute indirect, eight-bit relative, eight-bit relative indirect, absolute constant, 16-bit storage, 16-bit relative, and 16-bit relative indirect.

- Eight-Bit Absolute (address mode bits = 0, 1, 2, or 3) — Both relative and indirect flags are set to 0 and delta is not set to 0. The base address equals delta. Delta has no sign bit. The contents of the index registers, when specified, are added to the base address to form the effective address.
- Eight-Bit Absolute Indirect (address mode bits = 4, 5, 6, or 7) — The relative address flag is set to 0, the indirect flag is set to 1, and delta is not set to 0. The eight-bit value of delta is an indirect address. Delta is a magnitude quantity for this operation (no sign bit).
- Eight-Bit Relative (address mode bits = 8, 9, A, or B) — The relative flag is set to 0, and delta is not set to 0. The base address is equal to the instruction address P plus the value of delta with sign extended. The contents of the index registers, when specified, are added to the base address to form the effective address.
- Eight-Bit Relative Indirect (address mode bits = C, D, E, or F) — Both relative and indirect flags are set to 1. If delta is not set to 0, the value of the instruction address P plus the value of delta with sign extended is an indirect address. If bit 15 of the contents of this indirect address is 0, the contents of this indirect address is the base address. If bit 15 of the contents of the indirect address is set when the computer is in 32K mode, another indirect address is indicated.
- Absolute Constant (address mode bits = 0, 1, 2, or 3) — Both relative and indirect flags and delta are set to 0.

  When the address mode bits are set to 0, P + 1 is the effective address. When the address mode bits are set to 1, 2, or 3, the contents of P + 1 plus the contents of one or both index registers form the effective address. The effective address is taken as the operand for read-operand type instructions.
- 16-Bit Storage (address mode bits = 4, 5, 6, or 7) — The relative address flag and delta are set to 0 and the indirect flag is set to 1. The contents of location P + 1 is an indirect address. When the base address is formed (indirect addressing complete), the contents of one or both index registers, if specified, are added to form the effective address.
- 16-Bit Relative (address mode bits = 8, 9, A, or B) — The relative address flag is set to 1, and the indirect address flag and delta are set to 0. If no indexing is specified, the instruction address P + 1 plus the contents of location P + 1 form the base address or effective address. If indexing is specified, the contents of the specified index register(s) are added to the base address to form the effective address.
- 16-Bit Relative Indirect (address mode bits = C, D, E, or F) — Both relative and indirect flags are set to 1. In 65K mode, the contents of P + 1 + $(P + 1)^{\dagger}$ is the base address. Then the contents

---

†( ) Denotes contents of expression

TABLE 4-2. STORAGE REFERENCE INSTRUCTION ADDRESSING

Mode	Binary 11 10 9 8	Hex.	Δ Delta	Effective Address	Address of Next Instruction
8-Bit Absolute	0000	0	≠0	Δ	P+1
	0001	1		Δ+(00FF)	
	0010	2		Δ+(Q)	
	0011	3		Δ+(Q)+(00FF)	
8-Bit Absolute Indirect††	0100	4		(Δ)	
	0101	5		(Δ)+(00FF)	
	0110	6		(Δ)+(Q)	
	0111	7		(Δ)+(Q)+(00FF)	
8-Bit Relative	1000	8		P+Δ	
	1001	9		P+Δ+(00FF)	
	1010	A		P+Δ+(Q)	
	1011	B		P+Δ+(Q)+(00FF)	
8-Bit Relative Indirect††	1100	C		(P+Δ)	
	1101	D		(P+Δ)+(00FF)	
	1110	E		(P+Δ)+(Q)	
	1111	F		(P+Δ)+(Q)+(00FF)	
Absolute Constant	0000	0	=0	P+1	P+2
	0001	1		(P+1)+(00FF)†	
	0010	2		(P+1)+(Q)†	
	0011	3		(P+1)+(Q)+(00FF)†	
16-Bit Storage††	0100	4		(P+1)	
	0101	5		(P+1)+(00FF)	
	0110	6		(P+1)+(Q)	
	0111	7		(P+1)+(Q)+(00FF)	
16-Bit Relative	1000	8		P+1+(P+1)	
	1001	9		P+1+(P+1)+(00FF)	
	1010	A		P+1+(P+1)+(Q)	
	1011	B		P+1+(P+1)+(Q)+(00FF)	
16-Bit Relative Indirect††	1100	C		(P+1+(P+1))	
	1101	D		(P+1+(P+1))+(00FF)	
	1110	E		(P+1(P+1))+(Q)	
	1111	F		(P+1+(P+1))+(Q)+(00FF)	

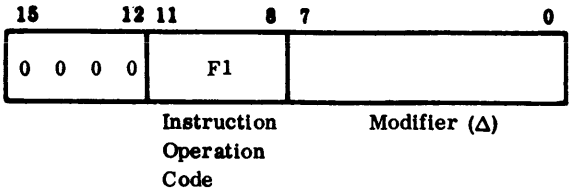
†Effective address is the operand for read-operand type instructions.

††Multilevel only in 32K word mode

of the index registers, when specified, are added to the base address to form the effective address. In 32K mode, P + 1 + (P + 1) = base address if bit 15 of (P + 1) is 0; if 1, then P + 1 + (P + 1) forms an indirect address. This process continues until bit 15 = 0.

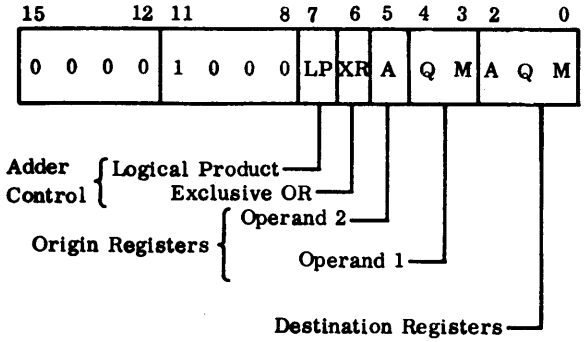
Register Reference

Register reference instructions use the address mode field for the operation code. These instructions are identified by 0s in the upper four bits of an instruction and the F1 instruction operation code (address mode field) cannot be a one, eight, or 15:



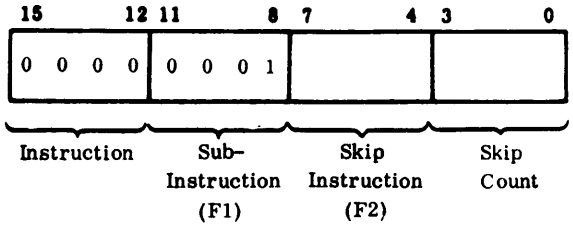
Inter-Register

Inter-register instructions (F1 = 8) are identified by an 8 in the address mode field and a 0 in the instruction mode field. These instructions cause data from certain combinations of origin registers to be sent through the adder to any combination of destination registers. Various operations, selected by the adder control lines, are performed on the data as it passes through the adder. The inter-register instruction format is:



Skip

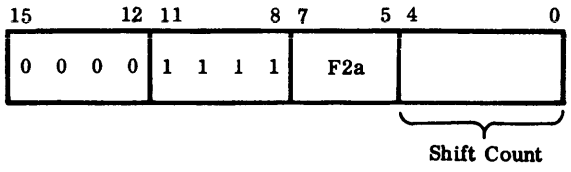
Skip instructions (F1=1) are identified by a 1 in the address mode field and a 0 in the instruction mode field:



When the skip condition is met, the contents of the skip count + 1 is added to P to obtain the address of the next instruction (e.g., when the skip count is 0, go to P + 1). When the skip condition is not met, the address of the next instruction is P + 1 (skip count ignored). The skip count does not have a sign bit.

Shift

Shift instructions are identified by a 15 in the address mode field and a 0 in the instruction mode field. These instructions shift A or Q, or QA left or right for the number of places specified by the five-bit shift count. The sign is extended on right shifts. Left shifts are end-around. This instruction has the following format:

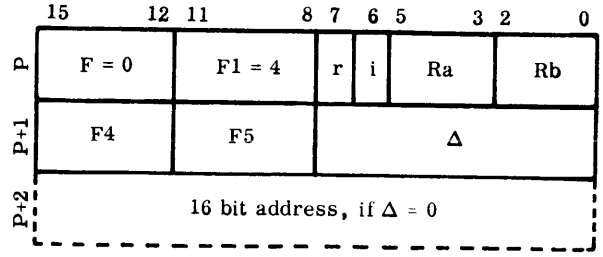


ENHANCED INSTRUCTIONS

Instruction formats for enhancements to the 1700 instruction repertoire are upward-compatible with existing 1700 computers. They make use of previously undefined instruction formats.

Enhanced Storage Reference

These instructions have the following format:



The enhanced storage reference instructions are identified when the F field is 0, the F1 field is equal to 4, and the r, i, Ra, and Rb fields are not all 0. (If these fields are all 0, the instruction is an EIN.) This instruction is made up of two (or three, if delta is 0) words.

The enhanced storage reference instructions are similar to the basic storage references in that they contain four parts: instruction field (F4), instruction mode field (F5), addressing mode fields (delta, r, i, and Ra), and register Rb. Two operands (A and B) are specified for executing the instruction.

The F4 field determines the instruction (e.g., add, subtract, etc.). The F5 field determines the instruction mode:

F5 = 0	Word processing; register destination
1	Word processing; memory destination
2	Character processing; register destination
3	Character processing; memory destination

NOTE

F5 is not used for subroutine jumps and subroutine exit. The register/memory destination bit of F5 is not used for compare instructions (see below).

The addressing mode fields contain four fields:

1. Delta determines eight- or 16-bit addressing. If delta is 0, a third word will be required to specify a 16-bit address.
2. Flag r is the relative address flag.
3. Flag i is the indirect address flag.
4. Register Ra is the index register.

The addressing modes are similar to the basic storage instructions. The basic set allows indexing by one or two registers (I and Q); while the enhanced set allows indexing by any one of seven registers (1, 2, 3, 4, Q, A, or I). Table 4-3 specifies the addressing modes, the effective address, and the address of the next instruction.

The addressing mode fields determine the effective address for operand A. Register Rb and the instruction mode field (F5) determine the address for operand B. Note that for character addressing, the effective address (operand A and register Rb) are combined to ascertain the actual character effective address. Operand B is always the A register for character addressing.

CAUTION

For character addressing, selection of absolute (r = 0), no indirect (i = 0), no index register (Ra = 0), and no character register (Rb = 0) will result in an EIN instruction.

Any unspecified combinations of F4, F5, and Rb are reserved for future expansion.

The following definitions apply to the description of addressing modes:

- Instruction Address — The address of the instruction being executed, also called P.
- Indirect Address — A storage address that contains an address rather than an operand. Note that there is no multilevel indirect addressing for enhanced storage reference instructions.
- Base Address — The operand address after all indirect addressing but before modification by an index register. The base address is the effective address if no indexing is specified.

TABLE 4-3. ENHANCED STORAGE REFERENCE INSTRUCTION ADDRESSES

Addressing Mode	Delta	r	i	Ra	Effective Address (EA)	Address of Next Instruction
8-Bit Absolute	$\Delta \neq 0$	0	0	0	Δ	P + 2
		0	0	1	Δ + (1)	P + 2
		0	0	2	Δ + (2)	P + 2
		0	0	3	Δ + (3)	P + 2
		0	0	4	Δ + (4)	P + 2
		0	0	5	Δ + (Q)	P + 2
		0	0	6	Δ + (A)	P + 2
		0	0	7	Δ + (I)	P + 2
8-Bit Absolute Indirect	$\Delta \neq 0$	0	1	0	(Δ)	P + 2
		0	1	1	(Δ) + (1)	P + 2
		0	1	2	(Δ) + (2)	P + 2
		0	1	3	(Δ) + (3)	P + 2
		0	1	4	(Δ) + (4)	P + 2
		0	1	5	(Δ) + (Q)	P + 2
		0	1	6	(Δ) + (A)	P + 2
		0	1	7	(Δ) + (I)	P + 2
8-Bit Relative†	$\Delta \neq 0$	1	0	0	P + 1 + Δ	P + 2
		1	0	1	P + 1 + Δ + (1)	P + 2
		1	0	2	P + 1 + Δ + (2)	P + 2
		1	0	3	P + 1 + Δ + (3)	P + 2
		1	0	4	P + 1 + Δ + (4)	P + 2
		1	0	5	P + 1 + Δ + (Q)	P + 2
		1	0	6	P + 1 + Δ + (A)	P + 2
		1	0	7	P + 1 + Δ + (I)	P + 2
8-Bit Relative Indirect†	$\Delta \neq 0$	1	1	0	(P + 1 + Δ)	P + 2
		1	1	1	(P + 1 + Δ) + (1)	P + 2
		1	1	2	(P + 1 + Δ) + (2)	P + 2
		1	1	3	(P + 1 + Δ) + (3)	P + 2
		1	1	4	(P + 1 + Δ) + (4)	P + 2
		1	1	5	(P + 1 + Δ) + (Q)	P + 2
		1	1	6	(P + 1 + Δ) + (A)	P + 2
		1	1	7	(P + 1 + Δ) + (I)	P + 2

†For these addressing modes, delta is sign extended.

Note: ( ) Denotes contents of expression.

TABLE 4-3. ENHANCED STORAGE REFERENCE INSTRUCTION ADDRESSES (Continued)

Addressing Modes	Delta	r	i	Ra	Effective Address (EA)	Address of Next Instruction
Absolute Constant	Δ = 0	0	0	0	P + 2	P + 3
		0	0	1	P + 2 + (1)	P + 3
		0	0	2	P + 2 + (2)	P + 3
		0	0	3	P + 2 + (3)	P + 3
		0	0	4	P + 2 + (4)	P + 3
		0	0	5	P + 2 + (Q)	P + 3
		0	0	6	P + 2 + (A)	P + 3
		0	0	7	P + 2 + (I)	P + 3
16-Bit Storage	Δ = 0	0	1	0	(P + 2)	P + 3
		0	1	1	(P + 2) + (1)	P + 3
		0	1	2	(P + 2) + (2)	P + 3
		0	1	3	(P + 2) + (3)	P + 3
		0	1	4	(P + 2) + (4)	P + 3
		0	1	5	(P + 2) + (Q)	P + 3
		0	1	6	(P + 2) + (A)	P + 3
		0	1	7	(P + 2) + (I)	P + 3
16-Bit Relative	Δ = 0	1	0	0	P + 2 + (P + 2)	P + 3
		1	0	1	P + 2 + (P + 2) + (1)	P + 3
		1	0	2	P + 2 + (P + 2) + (2)	P + 3
		1	0	3	P + 2 + (P + 2) + (3)	P + 3
		1	0	4	P + 2 + (P + 2) + (4)	P + 3
		1	0	5	P + 2 + (P + 2) + (Q)	P + 3
		1	0	6	P + 2 + (P + 2) + (A)	P + 3
		1	0	7	P + 2 + (P + 2) + (I)	P + 3
16-Bit Relative Indirect	Δ = 0	1	1	0	(P + 2 + (P + 2))	P + 3
		1	1	1	(P + 2 + (P + 2)) + (1)	P + 3
		1	1	2	(P + 2 + (P + 2)) + (2)	P + 3
		1	1	3	(P + 2 + (P + 2)) + (3)	P + 3
		1	1	4	(P + 2 + (P + 2)) + (4)	P + 3
		1	1	5	(P + 2 + (P + 2)) + (Q)	P + 3
		1	1	6	(P + 2 + (P + 2)) + (A)	P + 3
		1	1	7	(P + 2 + (P + 2)) + (I)	P + 3

Note: ( ) denotes contents of expression

- Effective Address — The final address of the operand.
- Indexing — If specified, the contents of register Ra is added to the base address to form the effective address. Indexing occurs after addressing is completed.

  The computer uses the 16-bit ones complement adder during indexing operations. Consequently, the index register contents are treated as signed quantities (bit 15 is the sign bit).
- Registers — Registers Ra and Rb are defined as follows:

Register	Value
None	0
1	1
2	2
3	3
4	4
Q	5
A	6
I	7

Enhanced storage reference instructions (table 4-8) have the following types of addressing modes:

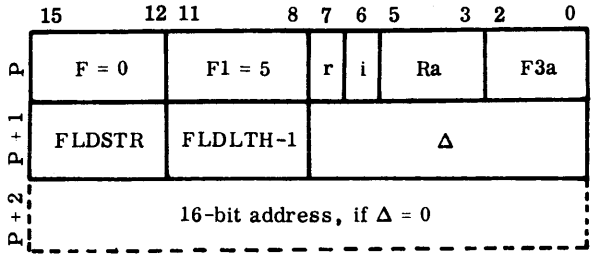
- Eight-Bit Absolute — (r = 0, i = 0, and $\Delta \neq 0$) — The base address equals delta and the sign bit of delta is not extended. The contents of index register Ra, when specified, are added to the base address to form the effective address.
- Eight-Bit Absolute Indirect (r = 0, i = 1, and $\Delta \neq 0$) — The eight-bit value of delta is an indirect address. The sign bit of delta is not extended. The content of this address in low core (addresses 0001_{16} to $00FF_{16}$) is the base address. The contents of index register Ra, when specified, are added to the base address to form the effective address.
- Eight-Bit Relative (r = 1, i = 0, and $\Delta \neq 0$) — The base address is equal to the instruction address plus one, P + 1, plus the value of delta with sign extended. The contents of index register Ra (when specified) are added to the base address to form the effective address.

  If no indexing takes place, the addresses that can be referenced in the eight-bit relative mode are restricted to the program area. Delta is eight bits long, thus the computer references a location between P - $7E_{16}$ and P + 80_{16} inclusive.
- Eight-Bit Relative Indirect (r = 1, i = 1, and $\Delta \neq 0$) — The address of the second word of the instruction, P + 1, plus the value of delta with sign extended is an indirect address. The content of this address is the base address. The contents of index register Ra, when specified, are added to the base address to form the effective address.
- Absolute Constant (r = 0, i = 0, and $\Delta = 0$) — The address of the third word of the instruction, P + 2, is the base address. The contents of the index register Ra, when specified, are added to the base address to form the effective address. Thus, when Ra is not specified, the contents of P + 2 is the value of the operand.

  Note that there is no immediate operand condition (i.e., indexing is specified and the instruction is a read-operand type) as there is for basic storage reference addressing.
- 16-Bit Storage (r = 0, i = 1, and $\Delta = 0$) — The base address equals the contents of P + 2. The contents of index register Ra, when specified, are added to the base address to form the effective address.
- 16-Bit Relative (r = 1, i = 0, and $\Delta = 0$) — The base address equals the contents of P + 2 plus P + 2. The contents of index register Ra, when specified, are added to the base address to form the effective address.
- 16-Bit Relative Indirect (r = 1, i = 1, and $\Delta = 0$) — The address of the third word of the instruction, P + 2, plus the contents of the third word of the instruction is an indirect address. The content of this address is the base address. The contents of the index register Ra, when specified, are added to the base address to form the effective address.

Field Reference

These instructions have the following format:



Field reference instructions are identified when the F field is 0, the F1 field is equal to 5, and the r, i, Ra, and F3a fields are not all 0. (If these fields are all 0, the instruction is an IIN.)

Field reference instructions contain four parts: operation field (F3a), addressing mode fields (Δ, r, i, and Ra), FLDSTR, and FLDLTH-1 fields. The F3a field determines the operation (e.g., load, store). The addressing mode fields are defined exactly as the enhanced storage reference instructions.

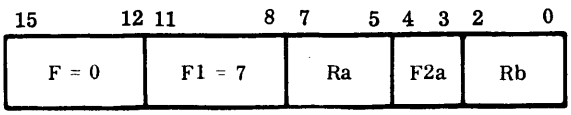
FLDSTR defines the starting bit of the field. For example, FLDSTR = 0 indicates that the field starts at bit 0. FLDLTH-1 defines the length of the field minus one. FLDLTH-1 = 0 indicates that the field is one bit long. If FLDLTH-1 = 0, the field reference instructions become bit reference instructions.

A field starts at the bit specified by FLDSTR and includes the contiguous FLDLTH bits to the right of that bit. No field may cross a word boundary (i.e., FLDSTR-FLDLTH-1 must be greater than or equal to 0). If FLDSTR = 0, the field length must be one bit long (FLDLTH-1 = 0).

Note that F3a = 0, F3a = 1, and FLDSTR-FLDLTH-1 < 0 are reserved for future expansion.

Enhanced Inter-Register

These instructions have the following format:



Enhanced inter-register instructions are identified when the F field is 0, the F1 field is 7, and the F2a, Ra, and Rb fields are not all 0. (If these fields are all 0, the instruction is CPB.)

Enhanced inter-register instructions (similar to the basic) inter-register instructions, such as TRA Q) contain three parts: operation field (F2a) and two register fields (Ra and Rb). The F2a field determines the operation (e.g., transfer). The Ra and Rb fields specify two operands.

Note that F2a = 1, F2a = 2, F2a = 3, Ra = 0, and Rb = 0 are reserved for future expansion.

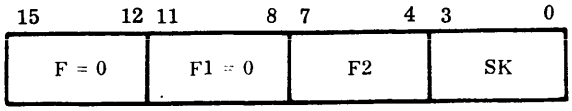
The instruction description is:

Transfer Register XFr R
F2a = 0
Ra = 1, 2, 3, 4, 5, 6, or 7
r = 1, 2, 3, 4, Q, A, or 1

Transfer the contents of register r to register R. Note that R = 1, 2, 3, 4, Q, A, or I implies that Rb = 1, 2, 3, 4, 5, 6, or 7.

Enhanced Skip

The skip instructions have the following format:



Enhanced skip instructions are identified when the F and F1 fields are both 0, and the F2 and SK fields are not both 0. (If these fields are both 0, the instruction is an SLS.)

Enhanced skip instructions (similar to the basic skips; such as SAZ) contain two parts: operation field (F2) and skip count (SK). The F2 field determines the operation (i.e., skip on register 1, 2, 3, or 4 if zero, nonzero, positive, or negative). The skip count specifies how many locations to skip if the skip condition is met.

When the skip condition is met, the skip count plus one is added to the P register to obtain the address of the next instruction (e.g., when the skip count is one, go to

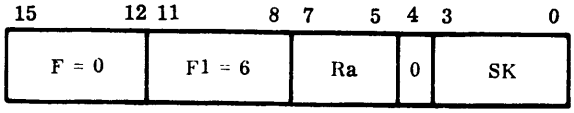
P + 2). When the skip condition is not met, the address of the next instruction is P + 1 (skip count ignored). The skip count does not have a sign bit.

If F2 = 0 (S4Z), the skip count cannot be 0 because the instruction would be an SLS.

The instruction descriptions are given in table 4-11.

Decrement and Repeat

These instructions have the following format:



Decrement and repeat instructions are specified when the F field is 0, the F1 field is 6, bit 4 is 0, and the Ra and SK fields are not both 0. (If these fields are both 0, the instruction is a SPB.)

Decrement and repeat instructions contain two parts: register field (ra) and skip count (SK). The register field specifies which register is to be decremented by one and checked for the skip condition. The skip count specifies how many locations to repeat (go backwards) if the skip condition is met.

When the skip condition is met, the skip count is subtracted from the P register to obtain the address of the next instruction (e.g., when the skip count is one, go to P - 1). When the skip condition is not met, the address of the next instruction is P + 1. The skip count does not have a sign bit.

Note that Ra = 0 and bit 4 = 1 are reserved for future expansion.

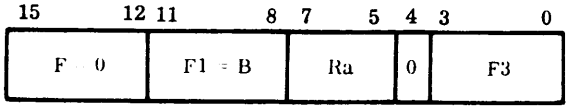
The instruction description is:

Decrement and Repeat if Positive DrP SK
Ra = 1, 2, 3, 4, 5, 6, or 7
r = 1, 2, 3, 4, Q, A, or I

Decrement the contents of register r by one. Operation on overflow is the same as for the ADD instruction. Repeat (go backwards) SK locations if the contents of register r are positive (bit 15 is 0), otherwise execute the next instruction.

Miscellaneous

Miscellaneous instructions have the following format:



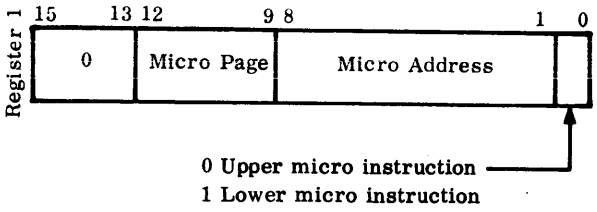
Miscellaneous instructions are specified when the F field is 0, the F1 field is equal to a decimal 11 (hexadecimal B), bit 4 is 0, and Ra and F3 fields are not both 0. (If these fields are both 0, the instruction is an NOP.) All of the miscellaneous instructions are privileged instructions; i.e., if they are executed by an unprotected program, they will cause a program protect violation.

Miscellaneous instructions contain two parts: operation field (F3) and register field (Ra).

If Ra is nonzero, the F3 operation field can select up to 16 miscellaneous instructions with register Ra used to specify an operand. If Ra is 0, the F3 operation field can select up to 15 more miscellaneous instructions without any explicit operand specified.

The miscellaneous instruction formats are:

1. Load Micro Memory



Initially, the Q register contains the number of 32-bit micro-memory instructions to be transferred (if Q = 0, no instructions will be transferred). Register 1 contains the starting address of micro memory. Register 2 contains the starting address of the macro memory.

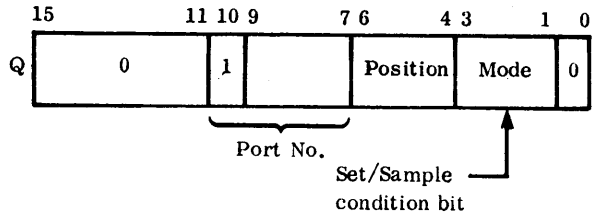
The most significant bit (15) of the contents of the starting address will be transferred to the most significant bit of the first micro instruction. The least significant bit (0) of the contents of the starting address

plus one will be transferred to the least significant bit. This instruction is interruptible after storing each 32-bit micro memory instruction, and when registers 1, 2, and Q are incremented/decremented to allow the instruction to be restarted after any interruption. When the instruction is completed, these registers will contain the following rather than their original values:

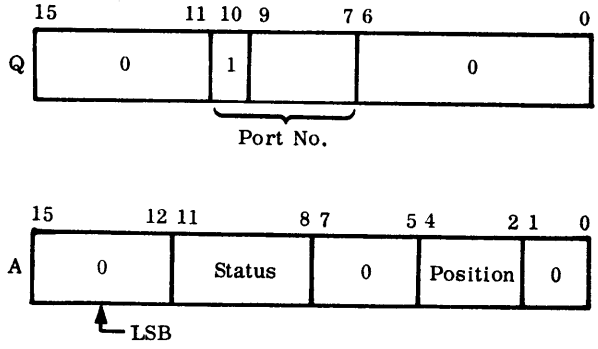
Q ← 0
R1 ← (R1)i+(Q)i
R2 ← (R2)i+2*(Q)i

Where: i is the initial value before execution.

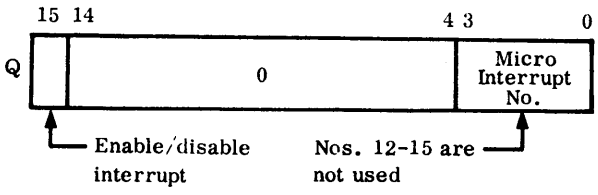
2. Set/Sample Output or Input

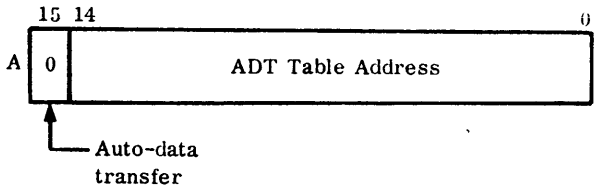


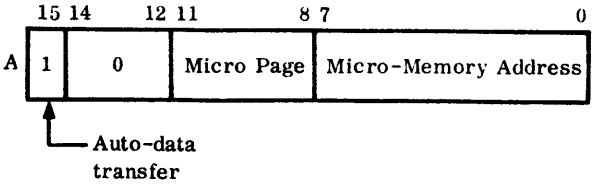
3. Sample Position Status



4. Define Micro Interrupt





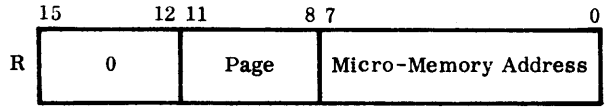


When bit 15 of the A register is set to a 1, a jump is made to the upper micro instruction of the page/micro memory in bits 0 to 14. A section of micro memory is assumed to have been previously loaded, and it must process the micro interrupt properly and return control to the current macro instruction address (P) by jumping to the lower micro instruction of micro-memory address $3E_{16}$ in micro page zero. Registers P, A, Q and all of file 2 should not be altered, and return must be within 12.5 microseconds.

CAUTION

The micro function, SUB-, must not be used. Extreme caution should be exercised in using this option, since it provides an escape from the 1700 emulation being performed.

5. Execute Micro Sequence



An EMS to non-existant micro memory may cause an indeterminate result. Control should be returned to the next macro-instruction address (P + 1) by jumping to the lower micro instruction of micro-memory address $3E_{16}$ in micro page zero. Registers P, A, Q, and all of file 2 should not be altered, and return must be within 12.5 microseconds (or the micro sequence must be interruptible).

CAUTION

Extreme caution should be exercised in using this option, since it provides an escape from the 1700 emulation being performed.

INTERRUPT SYSTEM

The interrupt system enables the program to establish an interrupt priority so that an interrupt of high priority can interrupt the machine while it is processing an interrupt of a lower priority. The return path to the interrupted program is clearly established and saved.

INTERRUPT TRAP LOCATIONS

Trap locations are established for each interrupt line. They are in the range of addresses 0100 through 013C. These addresses are reserved for interrupts unless that particular interrupt is not being used. The assignment for each interrupt state or line is shown in table 4-4.

TABLE 4-4. INTERRUPT STATE DEFINITIONS

Interrupt State	Value of Δ to Exit State	Location of Return Address	Location of First Instruction after Interrupt Occurs
00	00	0100	0101
01	04	0104	0105
02	08	0108	0109
03	0C	010C	010D
04	10	0110	0111
05	14	0114	0115
06	18	0118	0119
07	1C	011C	011D
08	20	0120	0121
09	24	0124	0125
10	28	0128	0129
11	2C	012C	012D
12	30	0130	0131
13	34	0134	0135
14	38	0138	0139
15	3C	013C	013D

MASK REGISTER

The mask register is the enable for each interrupt state or line. Bit 0 of the mask register corresponds to the interrupt line 0, bit 1 to line 1, etc. To enable an interrupt line, its corresponding bit in the mask register must be set. The mask register is set by the inter-register instruction.

PRIORITY

The computer program controls the interrupt priority by establishing an interrupt mask for each interrupt state, which enables all higher priority interrupts and disables all lower priority interrupts. When an interrupt state is entered, the mask for that state is placed in the mask register. Therefore, there may be up to 16 levels of priority. It is possible to change priority during execution of a program.

INTERNAL INTERRUPTS

Interrupts are also generated by certain conditions arising within the computer. These are called internal interrupts. If such a condition occurs, it generates interrupt 00 (interrupt mask bit 00). Normally, internal interrupts are assigned the highest priority. The internal interrupts are:

- Storage Parity Error
- Program Protect Fault
- Power Failure

OPERATION

The computer can distinguish between up to 16 (1 internal, 15 external) macro interrupts. Each of these interrupts has its respective address to which control is transferred when the interrupt is recognized.

When the computer is processing a particular interrupt, it will be defined as being in that interrupt state (state 00 through 15). Thus, the interrupts and their respective bits in the interrupt mask register are numbered 00 through 15. An interrupt in bit 7 will put the computer in interrupt state 7, etc.

Before the computer can recognize any interrupt, the mask bit for that interrupt must be set and the interrupt system must be activated. The mask register may be set by an inter-register command and the interrupt system can be activated by an enable interrupt command.

When an interrupt is recognized, the computer automatically stores the return address in the storage location reserved for that interrupt state. If 32K multilevel indirect mode has been selected, bit 15 of the storage location is set or cleared to record the current state of the overflow indicator. If 65K multilevel indirect mode has been selected, all 16 bits are required to save the return address. Thus, the program must check for an overflow condition with an SOV or SNO instruction and record this condition for restoration of the overflow indicator. In both 32K and 65K modes the interrupt system is de-activated and control is transferred when the interrupt occurs. In 32K mode the overflow is cleared, while in 65K mode the SOV or SNO instruction must first be executed. The program then stores all registers, including the mask register, in addresses reserved for this interrupt state and loads the mask register with the mask to be used in this state. The 1s in the mask indicate the interrupts that have a higher priority than the interrupt being processed. The mask should not have a 1 in the position of the interrupt being processed; this would lose the return link. The program then activates the interrupt system and processes the interrupt.

The computer exits from an interrupt state when the program inhibits the interrupt and restores the registers (including the mask register). After loading the register, the program executes the exit interrupt command with delta equal to the lower eight bits of the base address of the interrupt state. This command reads the storage location where the return address is stored. The overflow indicator is set or cleared as specified by bit 16. The interrupt system is activated and control is transferred to the return address.

EXAMPLE

The following listing and sample program steps apply if there were five different possible interrupts and three levels of priority:

Interrupt 01	High priority
02, 05	Mid-priority
03, 04	Low priority

Bit	5 4 3 2 1 0	
Mask 1	1 1 1 1 1 1	Mask used for main program
Mask 2	1 0 0 1 1 1	Mask used for state 03, 04
Mask 3	0 0 0 0 1 1	Mask used for state 02, 05
Mask 4	0 0 0 0 0 1	Mask used for state 01

Main Program

Set mask register to Mask 1
Enable interrupt
.
.
.

State 01 Program

Store registers
Set mask to Mask 4
Enable interrupt
.
.
.
Inhibit interrupt
Exit interrupt 01

State 02 Program

Store registers
Set mask to Mask 3
Enable interrupt
.
.
Inhibit interrupt
Replace registers
Exit interrupt 02

State 03 Program

Store registers
Set mask to Mask 2
Enable interrupt
.
.
.
Inhibit interrupt
Replace registers
Exit interrupt 03

State 04 Program

Store registers
Set mask to Mask 2
Enable interrupt
.
.
.
Inhibit interrupt
Replace registers
Exit interrupt 04

State 05 Program

Store registers
Set mask to Mask 3
Enable interrupt
.
.
.
Inhibit interrupt
Replace registers
Exit interrupt 05

PROGRAM PROTECT

The CYBER 18-10 computer has a program protect system to protect a program in the computer from any other nonprotected program also in the computer. The system is built around a program protect bit contained in each word of storage. If the bit is set, the word is an operand or an instruction of the protected program. All operand and instruction locations of the protected program must have the program protect bit set. None of the instructions or operands of the nonprotected program can have the program protect bit set.

Whenever a violation of the program protect system, other than a direct storage access violation, is detected, the program protect fault flip-flop is set and an internal interrupt is generated. A violation indicates that the nonprotected program has attempted an operation that could harm the protected program.

PROGRAM PROTECT VIOLATIONS

The following are the program protect violations:

- A nonprotected instruction attempts to write in a protected storage location. The contents of the storage location are not changed.
- An attempt is made to write into a protected storage location via external storage access when a nonprotected instruction was the ultimate source of the attempt. The contents of the storage location are not changed.
- An attempt is made to execute a protected instruction following execution of a nonprotected instruction. The protected instruction is executed as a nonprotected selected stop instruction. However, it is not a violation if an interrupt caused this sequence of instructions.
- An attempt is made to execute the following instructions when they are not protected: any interregister instructions with bit 0 = 1, EIN, IIN, EXI, SPB, CPB, or any miscellaneous instructions (0Bxx). Those instructions become a nonprotected selective stop instruction under these circumstances.

Program protect is enabled by setting bit 8 in the function control register. If this bit is not set, then none of the above violations are recognized, with the exception of the external storage access protect violation.

STORAGE PARITY ERRORS RELATED TO PROGRAM PROTECTION

If a nonprotected instruction is attempting to write into storage and a storage parity error is present or occurs, the word in storage is not altered and a Storage Parity Error interrupt is enabled.

If a protected instruction is attempting to write into storage and a storage parity error occurs, the word is written into storage and a Storage Parity Error interrupt is enabled.

If the computer attempts to execute a SPB or CPB instruction and a storage parity error occurs, these become Pass instructions and a storage parity error interrupt is enabled.

SET/CLEAR PROGRAM PROTECT BIT

The program protect instructions (SPB or CPB) and the bounds instructions (LUB and LLB) are the only way in which the program protect bit may be set or cleared in each word of storage.

PROGRAMMING REQUIREMENTS

The following program requirements must be met:

- The program package that handles all interrupts for the nonprotected program must be completely checked out. This program must also be part of the protected program.
- The protected program must be a completely checked-out program.

PERIPHERAL EQUIPMENT PROTECTION

All peripheral equipment that is essential to the operation of the protected program must have a bit in the FCR to designate if the device is protected. If the bit is set, the peripheral device responds with a reject to all nonprotected commands (except status request) addressed to it. All protected commands have a normal response. If the bit is not set, the peripheral device responds in the normal manner to protected and nonprotected commands.

INPUT/OUTPUT

When referencing the input/output devices, the Q register should contain either 0090_{16} or 0091_{16} according to the following table:

Q Register	Computer Instruction: Output from A	Computer Instruction: Input to A
0090	Write	Read
0091	Director Function (1)	Director Status (2)

DIRECTOR FUNCTION (1):

Bit (A Register)	Function	Operation
00	Clear Controller	Clear all interrupt requests. Clear busy, interrupt, data, alarm, and manual interrupt conditions. Select read mode. Connect printer. Any interrupt request bit will take precedence over this function. Clear Controller is also used in conjunction with bits 11, 12, 14, and 15.
01	Clear Interrupt	Clear all interrupt requests and the manual interrupt. Any interrupt request bit will take precedence over this function.
02	Data Interrupt Request	Send an interrupt signal whenever a data status is active.
03	End-of-Operation Interrupt Request	Send an interrupt signal when the controller is not busy. In the EOP state the controller will accept a mode change.
04	Alarm Interrupt Request	Send an interrupt signal when the Lost Data Status is active.
05	Not used	
06	ADT Mode	Auto-data transfer operation
07	Not used	
08	Select Write Mode	An output operation; does not clear the alarm status.
09	Select Read Mode	An input operation
10	Connect Printer	Select a mode of operation in which the printer (with the paper tape punch, when used) and the tape reader (when used) are both connected to the controller. Data read from the paper tape in this mode will also be printed (and punched).
11	Not used	
12	Not used	
13	Disconnect Printer	Select a mode of operation in which the printer (and paper tape punch when used) is disconnected from the controller. Data read from paper tape is not printed (or punched). This mode allows non-ASCII codes and binary information to be transmitted to the computer.
14	Not used	
15	Not used	

All nonconflicting functions may be performed simultaneously. Select write mode and select read mode are rejected when the controller is busy. Other functions are always performed. When several functions are issued simultaneously and some of them can be performed, the output from the A instruction exists normally (Reply), but those functions that should be rejected are not performed. When none of the functions can be performed, the output from the A instruction is rejected.

DIRECTOR STATUS (2):

Bit (A Register)	Status	Description
00	Ready	Unit is ready.
01	Busy	Read mode — The controller is in the process of receiving a character from the TTY/CDT, or the holding register contains data for transfer to the computer. The busy status will drop upon completion of the data transfer. Write mode — The data register contains data and is in the process of transferring it to the TTY/CDT. The busy status will drop when the transfer is completed.
02	Interrupt	An interrupt condition exists in the controller.
03	Data	Read mode — The holding register contains data for transfer to the computer. The data status will drop when the transfer is completed. Write mode — The controller is ready to accept another character from the computer.
04		Always the inverse of busy (bit 01)
05	Alarm	Parity error or lost data or field error (no stop bit when expected) occurred.
06	Lost Data	The holding register contained data for transfer to the computer, and the TTY/CDT began to send a new sequence.
07	Parity Error	A parity error occurred
08	Release	Release reserve interrupt; this interrupt is generated when the CRT or TTY has been reserved for the panel interface and is returned.
09	Read Mode	The controller is conditioned for input operation.
10	Reserved	Indicates if the TTY or CRT is currently assigned to the panel interface and is unavailable to the TTY controller.
11	Manual Interrupt	A manual interrupt has occurred.
12	Not used	Always 0
13	Not used	Always 0
14	Not used	Always 0
15	Not used	Always 0

AUTO-DATA TRANSFER

Auto-data transfer (ADT) provides for pseudo direct memory transfers of data blocks to or from a device. At the macro level, the transfer appears as a direct memory access (DMA) transfer. At the micro level, the 1700 emulator processes each data interrupt and inputs or outputs the next data in a singular fashion. Thus, ADT takes less time than input/output via the INP, OUT, or SIO instructions, but more time than a true DMA transfer.

To accomplish an ADT for a particular device, perform the following:

1. The device and its controller must adhere to the auto-data transfer specifications in the Micro-programmable Computer I/O Specification.

2. The macro programmer must execute a DMI instruction. This command specifies where the block of data is, how long it is, the direction (input/output), and the device's address.

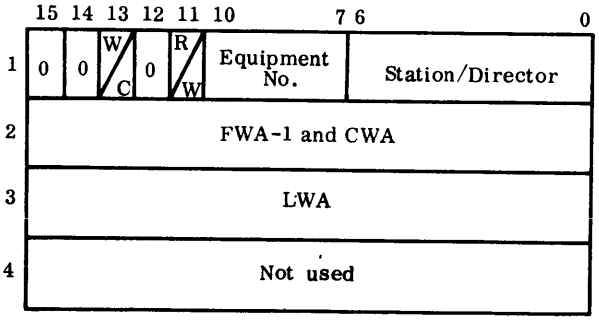
3. The ADT operation is then initiated by an INP, OUT, or SIO instruction as specified by the particular device.

While the ADT operation is in progress, the emulator is executing instructions. After each instruction is executed, interrupts are checked. When the particular ADT micro interrupt becomes the highest active interrupt, the next data is input or output. After the interruption, the next instruction is executed, except when another interrupt is active.

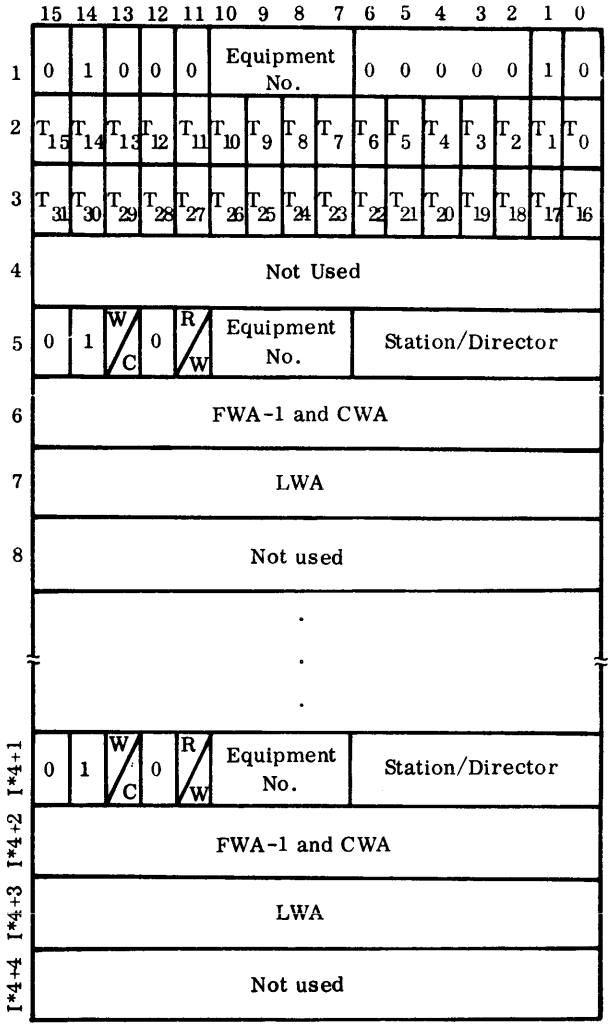
When the ADT operation is completed (or if there is an error), a macro interrupt is generated. The macro programmer may then disable the ADT micro interrupt or initiate another ADT operation to or from the device. For MOS devices, an SPS instruction must be performed to clear the macro interrupt.

The following are the four types of ADT tables specified by DMI instructions.

1. ADT Table for a Single A/Q Device:

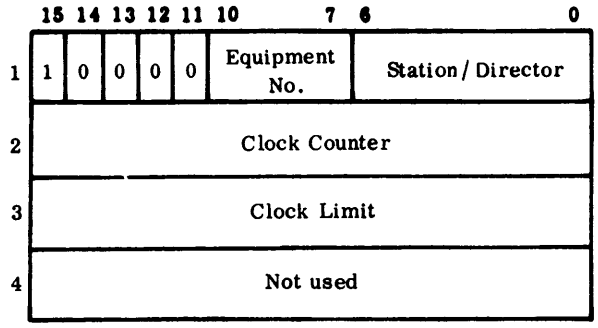


2. ADT Table for Multiple A/Q Devices:



This type of ADT table consists of I*4+4 words, where I is the number of multiple A/Q devices (up to 32) on one micro interrupt.

3. ADT Table for the Clock.



MODEL DESCRIPTIONS

Following are brief descriptions of the standard units. Descriptions of optional peripherals are included in section 6, Standard Peripherals.

CYBER 18-20 PROCESSOR

The CYBER 18-20 is a micro-programmable processor unit that accommodates from 32,768 to 262,144 eight-bit bytes (one protect and one parity bit is provided for each two bytes) of MOS main memory with an effective read/write cycle time of 750 nanoseconds. The 18-20 emulates the basic as well as the enhanced 1700 instruction set. Basic features of the processor are hardware multiply/divide, 16 levels each of micro and macro interrupts, auto-data transfer, a real-time clock, deadstart facility from peripherals, an A/Q I/O port that supports up to nine peripheral controllers, and a DMA I/O port that supports up to four DMA-type devices. Feature options are 512- through 4096-instruction read/write micro memory with minimum instruction execution time of 168 nanoseconds for user micro programs, a programmable micro/macro breakpoint controller, and a breakpoint panel. The 18-20 includes a cabinet operators panel, power supplies, and an I/O controller to support the 1810-1 Communications Console. No main memory or read/write micro memory is included.

CYBER 18-30 DUAL TIMESHARING SYSTEM

The CYBER 18-30 consists of dual micro-programmable processor units, each accommodating from 32,768 to 262,144 eight-bit bytes (one protect and one parity bit for each two bytes) of MOS main memory with an effective read/write cycle time of 750 nanoseconds. Each processor can address the memory of the other for a total capability of 524,288 eight-bit bytes of main memory. The processors emulate the basic as well as the enhanced 1700 instruction set. Basic features of each processor are hardware multiply/divide, 16 levels each of micro and macro interrupts, auto-data transfer, a real-time clock, deadstart facility from peripherals, an A/Q I/O port that supports up to nine peripheral controllers, and a DMA I/O port that supports up to four DMA devices.

The CYBER 18-30 Timesharing System provides a basic configuration that is sized for 16 concurrent users. The basic CYBER 18-30 Timesharing System provides one 64K byte processor, one 128K byte processor, one 1811-11 Conversational Display Terminal, two cassette tape units, one 25 million byte storage module drive with controller, a 300-line-per-minute printer, a 300-card-per-minute card reader, one nine-track, 25 ips, NRZI magnetic tape transport, and a communications multiplexer. Feature options of the CYBER 18-30 Timesharing System include additional main memory, synchronous and asynchronous communications line adapters, additional magnetic tape transports, user terminals (CRT or TTY), and additional storage module drives.

1870-1 512 INSTRUCTION MICRO MEMORY

The 1870-1 micro memory provides storage for up to 512 32-bit micro control instructions for the processor. The memory is a read or write RAM (random-access memory) and can be loaded externally or under control of the micro program. It is designed for micro program storage in those applications that require that the processor be programmed at the micro level.

1870-2 2048 INSTRUCTION MICRO MEMORY

The 1870-2 micro memory provides storage for up to 2048 32-bit micro control instructions for the processor. The memory is a read or write RAM and can be loaded externally or under control of the micro program. It is designed for micro program storage in those applications that require the processor to be programmed at the micro level.

1882-16 MOS MAIN MEMORY STORAGE

The 1882-16 provides 32,768 eight-bit bytes of read/write MOS memory. One protect bit and one parity bit are provided with each two bytes. The effective read/write cycle time is 750 nanoseconds. It occupies one memory position within the processor unit.

1882-32 MOS MAIN MEMORY STORAGE

The 1882-32 provides 65,536 eight-bit bytes of read/write MOS memory. One protect bit and one parity bit are provided with each two bytes. The effective read/write cycle time is 750 nanoseconds. It occupies one memory position within the processor unit.

1874-1 ECC MOS ARRAY, 192K BYTES

The 1874-1 provides storage for the five-bit error correction code (ECC) for up to 192K eight-bit bytes of 1882-16/1882-32 read/write MOS memory. The ECC facility corrects single bit errors and detects double bit errors. All interface control to ECC module is performed by the MOS memory interface module. The ECC module occupies one memory position within the processor unit. It limits the maximum memory size of the 18-20/18-30 processors to 192K eight-bit bytes.

MAIN MEMORY

Main memory for the CYBER 18-20 and 18-30 consists of 32K- or 64K-byte MOS memory stacks and two interface cards. The interface cards provide the control and interfacing required for CPU/memory function and peripheral (DMA) equipment/memory functions. Parity and program protect bits are generated and tested in the interface cards. The two interface cards handle up to four stacks (256K bytes) in the main CYBER 18-20 or CYBER 18-30 chassis. The interface cards also provide the dual-CPU access capability used in the CYBER 18-30 dual processor system.

CENTRAL PROCESSORS

The description of the CYBER 18-10 central processor applies to the CYBER 18-20 and 18-30, with the additions of a memory page file. The memory page file provides 64 nine-bit words that specify the most significant seven bits of the 18-bit memory address (512K-byte memory addressing). The total memory is divided into 128 4K-byte pages.

MACRO INSTRUCTION SET

The following macro level instructions are provided with the CYBER 18-20 and 18-30, in addition to those provided with CYBER 18-10:

- APM ($0B0B_{16}$) — Absolute page mode is specified. Only the first 128K bytes of memory can be addressed.
- PMO ($0B0C_{16}$) — Page mode 0 is specified. Segments of 2048 words (totaling 128K bytes can be addressed via the 32 page mode 0 registers.
- PM1 ($0B0D_{16}$) — Page mode 1 is specified. Segments of 2048 words (totaling 128K bytes can be addressed via the 32 page mode 1 registers.
- WPR R ($0B23_{16}$, $0B43_{16}$, . . . $0BE3_{16}$) — Write the contents of register R (bits 0-8) into the page register specified by bits 10 through 15 of register R. Bit 10 is 0 or 1 for a page mode 0 or 2 register, respectively. Bits 11 through 15 specify one of the 32 page registers. Bit 9 is unused and should be 0.
- RPR R ($0B24_{16}$, $0B44_{16}$, . . . $0BE4_{16}$) — Read the contents of the page register, specified by bits 10 through 15 of register R, into the A register. Bit 10 of register R is 0 or 1 for a page mode 0 or 1 register, respectively. Bits 11 through 15 specify one of the 32 page registers. Bits 0 through 9 are unused and should be zero.
- ECC R ($0B25_{16}$, $0B45_{16}$, . . . $0BE5_{16}$) — Input the ECC status of the memory word, specified by the address in register R, into the A register.

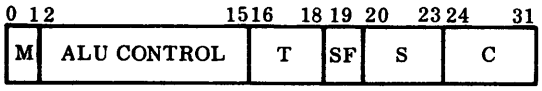
MICRO INSTRUCTION FORMATS

Each micro memory address specifies the location of two micro instructions. Each 32-bit micro instruction is divided into five main sections and is numbered from left to right as bits 0 to 31.

Bits	Function
0,1	Mode (M) field specifies format of S and C field and sequencing mode to obtain next micro instruction pair.

Bits	Function
2-15	ALU control field specifies ALU operation, sources of operands, and destination of result of operation.
16-18	Test (T) field specifies method of selecting which micro instruction of next micro instruction pair to execute.
19-23	Special (S) field specifies subformat selection and special operation.
24-31	Constant (C) or suboperation field specifies constants or other codes.

The total instruction appears as follows:



The M field specifies one of three addressing modes to be used to obtain the next micro instruction pair from micro memory and specifies the format to be used in interpreting bits 19 to 31 of the micro instruction as follows:

M	Addressing Mode	Format for Bits 19 to 31
00	Return	Format 1
01	Sequential	Format 1
10	Jump	Format 2
11	Sequential	Format 3

Figure 5-1 shows all the micro instruction formats. Note that format 1 and format 2 micro instructions have two subformats, which are selected by the value of bit 19 (SF).

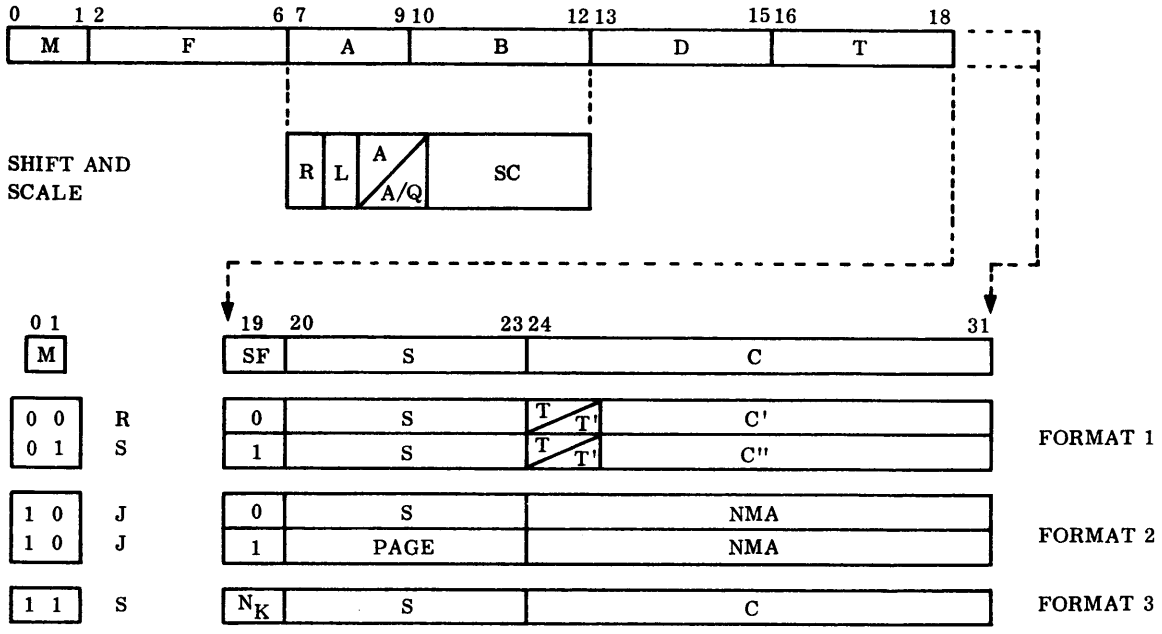
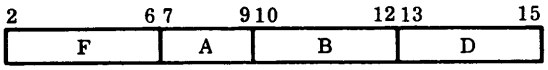
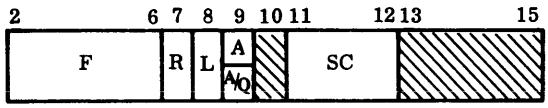


Figure 5-1. Micro Instruction Formats

The ALU control fields specify the sources of two operands on which an arithmetic, logical, shift, or scale operation is to be performed and specify the destination of the result of the operation. For arithmetic and logical operations, the ALU control fields consist of the ALU function (F), A source (A), B source (B), and destination (D) fields, shown as follows:



For shift and scale operations, the A and B fields are interpreted as follows:

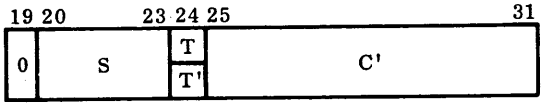


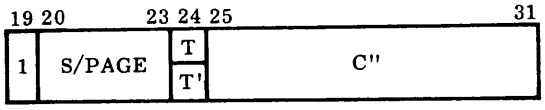
The F field specifies shift or scale operation. Bits 7 and 8 specify right or left shifting. Bit 9 specifies whether the A register alone or the A and Q registers together are to be shifted or scaled. Bit 10 is not used, and bits 11 and 12 specify the shift control code. The D field contains a no-operation code for shift and scale operations.

The T field is the conditional branch of the micro instruction and specifies which micro instruction, upper or lower, of the next micro instruction pair to execute. The test can be based on the result of the ALU operation of the current micro instruction or on some other condition.

The codings in the S and C fields depend upon the contents of the M field. The S and C fields are coded in three formats. Format 1 is specified when the M field contains 00 (return mode) or 01 (sequential mode) as follows:

Format 1

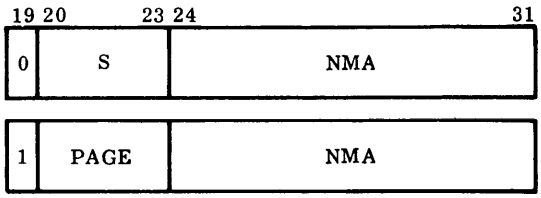




The S field specifies operations, such as main memory read or write operations, or alternate codings to be used in the A, B, and D fields. The T/T' bit specifies that the code in the T field is to be interpreted as the normal T code (T/T'=0) or as the alternate T' code (T/T'=1). The subformat select bit, bit 19, determines whether bits 25 through 31 are to be interpreted as C' codes or as C" codes. The C' code can contain a constant for driving the bit generator, additional information to control the main memory read or write, or other operations. The C" codes are associated with transforms.

Format 2 is specified when the M field contains 10 (jump mode) as follows:

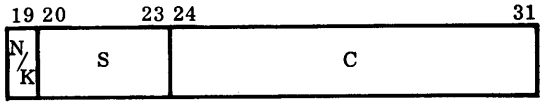
Format 2



If a jump is specified to a micro instruction pair within the same micro memory page, the subformat select bit is 0; bits 20 to 23 contain a special operation code as in format 1. Bits 24 through 31 contain the micro memory address of the next micro instruction pair. The subformat select bit is 1 when a jump is specified to a different micro memory page; bits 20 through 31 contain the complete micro memory address of the next micro instruction pair.

Format 3 is specified when the M field contains 11 (sequential mode), as follows:

Format 3



This format allows one special operation to be performed as specified by the S code and also causes the eight bits of the C field to be transferred to the N register (bit 19=1) or to the K register (bit 19=0).

The following paragraphs describe standard peripherals for the CYBER 18 product line. The availability of a given peripheral with respect to the CYBER 18-10, 18-20, and 18-30 products should be determined by referencing the configuration section of this manual.

1811-1 CONVERSATIONAL DISPLAY TERMINAL

The 1811-1 is a self-contained, single station CRT keyboard display terminal. It provides a 1920-character (24 lines of 80 characters) display unit with character-by-character, line-at-a-time, or page-at-a-time data transmission in either half or full duplex modes. The data rate is selectable from 110 to 9600 bits per second. The interface meets RS232-C, CCITT recommendations V24 as applied to asynchronous data communication. The character repertoire includes 128 symbols displayed within a 9 by 7 dot matrix.

1827-30/31 LINE PRINTER

The 1827-30/31 is a 300 line-per-minute drum printer in a quietized cabinet. It provides full line buffer facility, a 136-column print line, 12 VFU channels, a 64-character print drum, and six or eight line-per-inch print line spacing. The 1827-30/31 includes 20 feet of interface cable. The 1827-30 operates from 120 vac, 60 Hz source power. The 1827-31 operates from 220 vac, 50 Hz source power.

65119-1 LINE PRINTER

The 65119-1 is a 600 line-per-minute drum printer in a quietized cabinet. It provides full line buffer facility, a 136-column print line, 12 VFU channels, a 64-character print drum, and six or eight line-per-inch print line spacing. The 65119-1 includes 20 feet of interface cable. The 65119-1 operates from 120 vac, 60 Hz source power.

1828-1 CARD READER/LINE PRINTER CONTROLLER

The 1828-1 provides two independent controllers for connection of one card reader and/or one line printer to the processor unit; it occupies one A/Q card position within the processor unit. Card reader controller features are data/control interface between the processor and one card reader; it accepts card reader input data in form of Hollerith code, binary code, or any other desired format; it provides facility for deadstart operation of the processor unit; it performs Hollerith-to-ASCII code conversion during deadstart operation; and it performs normal data transfers under program control. The line printer controller features are data/control interface between the processor and one line printer; it performs data transfers under program control; and it has data buffer facility. An additional feature is the test mode capability of closed-loop operation under software control for diagnostic purposes. Cables are included as part of the card reader or line printer.

1829-30 CARD READER

The 1829-30 is a self-contained desk top unit provided with interface control logic and an operators control/indicator panel. Functional characteristics of the unit are 300 card-per-minute read speed, 1000 card hopper/stacker capacity, an 80 column punch card input medium, and a photoelectric read station with light/dark read checking facility. One 7-foot interface cable is supplied. The 1829-30 operates from 120 vac, 60 or 50 Hz source power. Option 1888-1 is available for 220 vac operation.

1829-60 CARD READER

The 1829-60 is a self-contained desk top unit provided with interface control logic and an operator control/indicator panel. Functional characteristics of the unit are 600 card-per-minute read speed, 1000 card/hopper stacker capacity, an 80-column punch card input medium, and a photoelectric read station with light/dark read checking facility. One 7-foot interface cable

is supplied. The 1829-60 operates from 120 vac, 60 or 50 Hz source power. Option 1888-1 is available for 220 vac operation.

1832-4 MAGNETIC TAPE CONTROLLER

The 1832-4 provides interface and control for up to four drives in any mix of seven or nine track. Both seven and nine-track operation is 800 bits per inch. NRZI mode at 25 inches per second. The controller features functional capabilities of read record, write record, write filemark, backspace, erase, rewind to loadpoint, rewind unload, recovery read, controlled backspace, and single track error correction for nine-track units. It occupies one position within the processor unit. It includes a 30-foot interface cable and a tape translator board.

1860-72 MAGNETIC TAPE TRANSPORT

The 1860-72 is a seven-track magnetic tape transport that operates in 800 bit-per-inch NRZI mode, 25 inches per second, and 20K six-bit characters per second. It rewinds at 150 inches per second. The transport does not include skins and must be housed in an 1887-4 cabinet. It requires an installation kit (1860-200 for upper cabinet installation or 1860-201 for lower cabinet installation). The 1860-72 operates from 120 vac, 50 or 60 Hz source power. Option 1888-1 is available for 220 vac operation.

1860-92 MAGNETIC TAPE TRANSPORT

The 1860-92 is a nine-track magnetic tape transport that operates in 800 bit-per-inch NRZI mode at 20K eight-bit characters per second. It rewinds at 150 inches per second. The transport does not include skins and must be housed in an 1887-4 cabinet. It requires an installation kit (1860-200 for upper cabinet installation or 1860-201 for lower cabinet installation). The 1860-92 operates from 120 vac, 50 or 60 Hz source power. Option 1888-1 is available for 220 vac operation.

1833-1 STORAGE MODULE DRIVE INTERFACE

The 1833-1 provides a single CPU A/Q-DMA channel interface to the 1833-3 Storage Module Control Unit. The interface handles all control and status operations via the A/Q channel and all data transfers via the DMA channel. The interface supports the control unit connection of up to eight 1867-xx drives in any mix. Connection to the 1833-3 control unit is via two 25-foot (7.62m) cable assemblies. The interface occupies one A/Q-DMA position within the processor unit.

1833-2 STORAGE MODULE DRIVE INTERFACE — DUAL CPU

The 1833-2 provides the A/Q-DMA channel interface for the second CPU in a dual-CPU SMD subsystem. The interface handles all control and status operations via the A/Q channel and all data transfers via the DMA channel. The interface supports the 1833-3 control unit connections of up to eight 1867-xx drives in any mix. Connection to the first computer's 1833-1 is via two 25-foot (7.62m) cable assemblies. The interface occupies one A/Q-DMA position within the processor unit.

1833-3 STORAGE MODULE DRIVE CONTROL UNIT

The 1833-3 is the control unit for the 1867-10/11/20/21 Storage Module Drives. It provides control for up to eight drives in any mix of 25 million 8-bit bytes and 50 million 8-bit bytes of formatted data capacity. The control unit handles all SMD data transfers, formatting, and error recovery. It provides for either single or dual CPU connection via the 1833-1 and 1833-2 SMD interface. The control unit is physically housed in the base cabinet of the first SMD in the subsystem. Input power may be either 50 or 60 Hz, 120 or 220 vac, single phase.

1867-10 STORAGE MODULE DRIVE

The storage module drive is a random-access device, using removable disk packs as the storage medium. It has a formatted data capacity of 25 million bytes. The maximum data transfer rate is 1.2 million bytes per second. The average access time is 30 milliseconds. The drive includes a base cabinet, one 10-foot A cable (daisy chain) and one 20-foot B cable (star). The 876 disk pack is not included. The 1867-10 operates from 120 vac, 60 Hz source power. The 1867-11 is available, which operates from 220 vac, 50 Hz source power.

1867-20 STORAGE MODULE DRIVE

The storage module drive is a random-access device using removable disk packs for the storage medium. It has a formatted data capacity of 50 million bytes. The maximum data transfer rate is 1.2 million bytes per second. The average access time is 30 milliseconds. The drive includes the base cabinet, one 10-foot A cable (daisy chain) and one 20-foot B cable (star). The 877 disk pack is not included. The 1867-20 operates from 120 vac, 60 Hz source power. The 1867-21 is available, which operates from 220 vac, 50 Hz source power.

1833-5 FLEXIBLE DISK DRIVE CONTROLLER

The 1833-5 provides single CPU A/Q-DMA channel interface and control for one or two 1865-1/2 Flexible Disk Drives. The controller is capable of handling all data, control, and status operations via the A/Q channel only or buffered data transfers via the DMA channel. The controller is capable of reading and writing in either the IBM 3740 format (128 bytes/sector) or the CDC 1700 rotating mass storage format (192 bytes/sector). It occupies one A/Q or A/Q-DMA position within the processor.

1865 FLEXIBLE DISK DRIVE

The flexible disk drive is a random-access device using removable diskettes for the storage medium. It has a formatted data capacity of 256K bytes when using the IBM format (128 bytes/sector) or 280K bytes using the CDC 1700 format (192 bytes/sector). The data transfer rate is 31.2K bytes per second. The average access time is 343 milliseconds. The 1865-1 is the first drive (unit 0) within an FDD subsystem and connects to the 1833-5 via a 10-foot unit 0 cable. The 1865-2 Flexible Disk Drive is the second unit in a dual flexible disk drive subsystem. The 1865-3 and 1865-4 Flexible Disk Drives operate from 220 vac, 50 Hz source power and are unit 0 and unit 1, respectively, in a FDD subsystem.

1843-1 COMMUNICATIONS LINE ADAPTER

The 1843-1 controller provides multiplexed dual channel interfaces for connection of two synchronous or asynchronous modems that conform to CCITT recommendations V24 or EIA RS232-C interface standards. A selectable baud rate of 110, 150, 300, 600, 1200, 2400, 4800, 9600, and 19,200 in asynchronous mode or 1200, 2400, 4800, and 9600 in the synchronous mode may be selected. The 1843-1 features software selection and control of half duplex or full duplex operation, character code lengths of 5, 6, 7, or 8 bits, stop bit length of 1x, 1.5x, or 2x, odd, even, or no parity bit generation and checking, and variable input/output speeds. An additional feature is an internal cyclic encoder for cyclic checkword generation. The 1843-1 occupies one A/Q position within the processor unit. One 20-foot modem cable is supplied.

1875-1 BREAKPOINT CONTROLLER

The breakpoint controller provides the breakpoint halt register for both micro memory and main memory for program debug. It also provides a hardware interface to the micro processor. This allows the operator to load and display all registers and interface to the function control register, which allows the setting of all control bits. When the 1875-1 is not installed, all functions except breakpoint are emulated by the panel simulator of the 1700 emulator. Operator interface is via the console display or the breakpoint panel. The controller occupies one dedicated card position within the processor.

1875-2 BREAKPOINT PANEL

The 1875-2 Breakpoint Panel provides the operator with an input medium to the breakpoint controller in the absence of the console display. It contains a 16-bit LED display and a limited keyboard. It receives parallel input from the breakpoint controller. The panel does not require a processor card position.

ACCESSORY DEVICES

Table 6-1 provides a list of the accessory devices available for the CYBER 18 product line.

TABLE 6-1. ACCESSORY DEVICES

Product No.	Description	Applies To	Use
1833-950	Storage Module Drive Interface Cable Option - 50 ft	1833-1	50-ft cables to 1867-xx
1843-901	Terminal Cable Adapter	1843-950	Connects display terminal to 1843-1
1843-950	Modem Cable - 50 ft	1843-1	Connects modem to 1843-1
1860-200	Magnetic Tape Drive Installation Kit, Upper	1860-72/92	Installs magnetic tape in upper half of 1887-4
1860-201	Magnetic Tape Drive Installation Kit, Lower	1860-72/92	Installs magnetic tape in lower half of 1887-4
1865-200	Small Peripheral Equipment Cabinet	1865-1/2/3/4	Houses two 1865-x
1865-920	Flexible Disk Drive Cable Option - 20 ft	1865-1/3	Connects 1865-1/3 to 1833-5
1865-921	Flexible Disk Drive Cable Option - 20 ft	1865-2/4	Connects 1865-2/4 to 1833-5
1887-4	Equipment Cabinet	1860-72/92	22.5 in. wide x 29.5 in. deep x 67.88 in. high cabinet for two 1860-xx
1827-950	Line Printer Interface Cable - 50 ft	1827-30/31/ 60/61	Connects 1827-xx to 1828-1
1888-1	Power Conversion Transformer - 220 vac to 120 vac	18-10/18-10A/ 18-20/18-30/ 1811-1/ 1827-30/ 1860-72/ 1860-92/ 1865-1/1865-2/ 1867-10/ 1867-20/ 1877-4	Provides 220 vac, 50 Hz operation of mainframes and/or peripherals
1888-2	Power Conversion Transformer - 120 vac to 220 vac	1811-2/1827-31/ 1865-3/1865-4/ 1867-11/ 1867-21	Provides 120 vac, 50 Hz operation of peripherals

CYBER 18 HARDWARE CONFIGURATOR

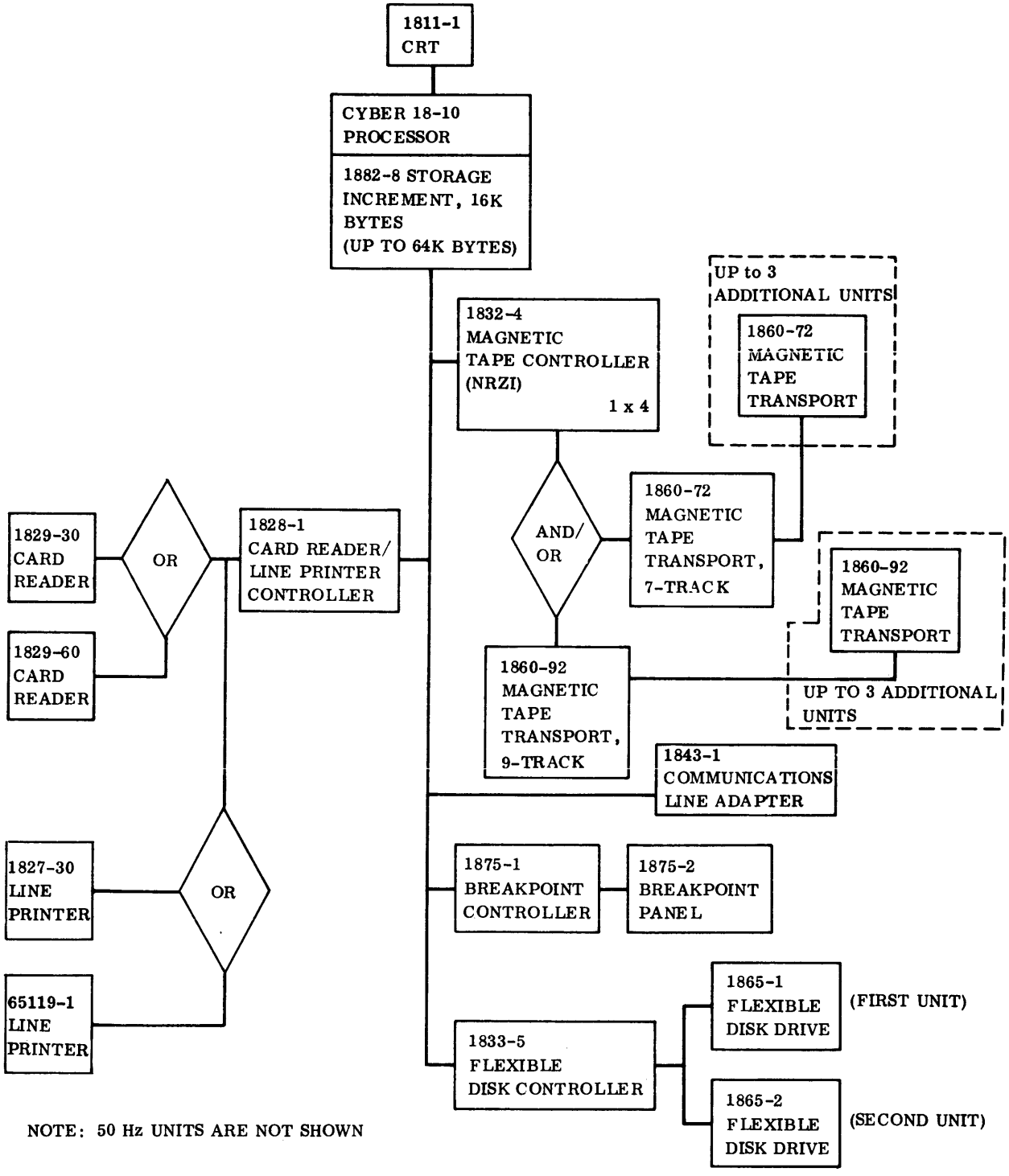


Figure 6-1. CYBER 18-10 Configurator

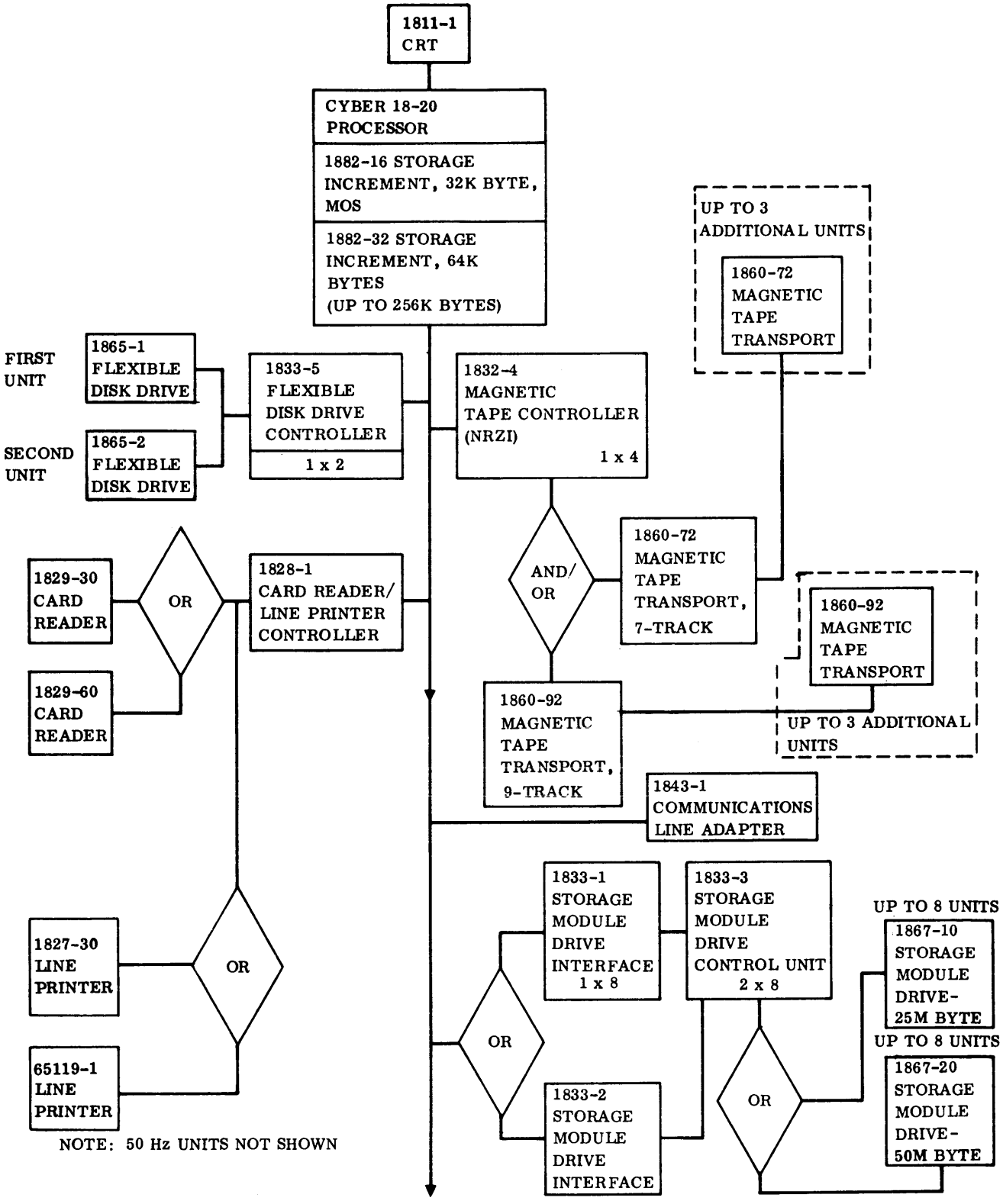


Figure 6-2. CYBER 18-20 Configurator

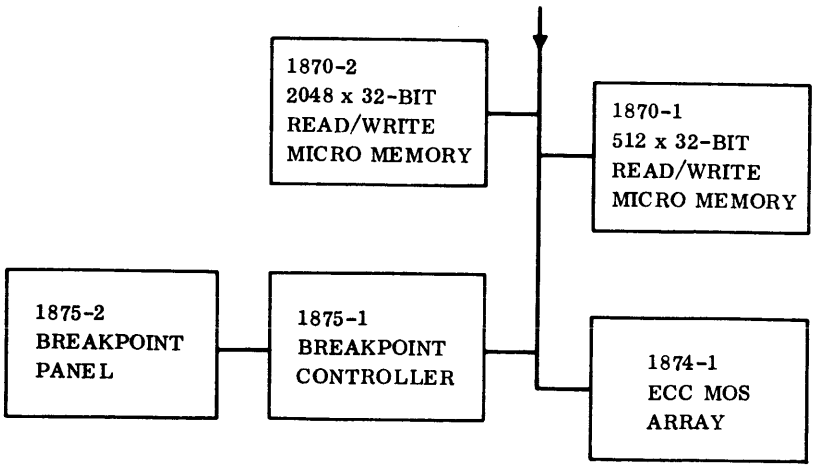


Figure 6-2. CYBER 18-20 Configurator (Continued)

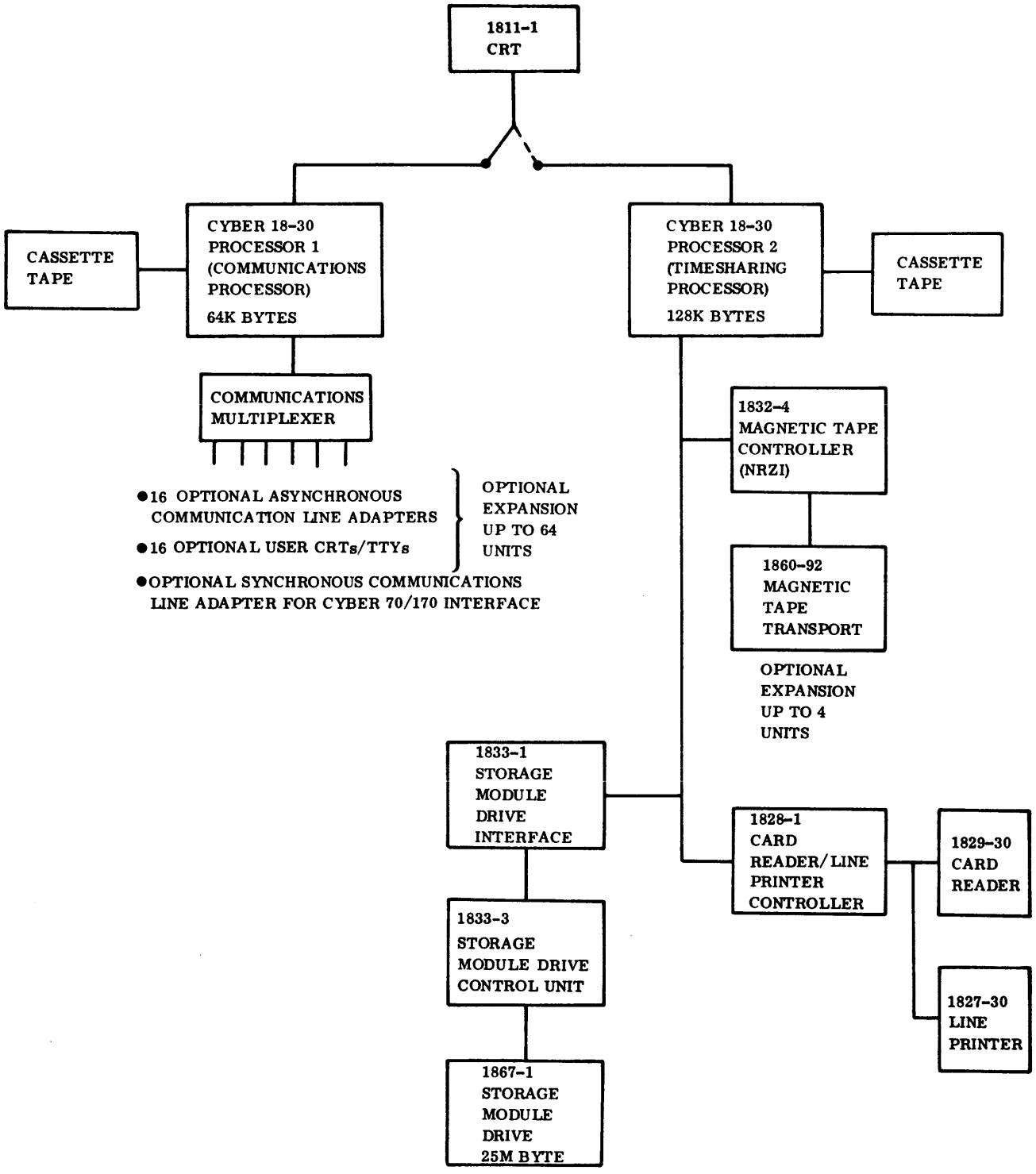


Figure 6-3. CYBER 18-30 Timesharing System

MACRO INSTRUCTION EXECUTION TIMES

When calculating the instruction execution times listed on the following pages, the user must consider the following parameters:

1. For basic storage reference instructions, add the following addressing mode time to the execution time given in this appendix.

F1	Delta	Additional Execution Time	F1	Delta	Additional Execution Time
0	≠ 0	.00	0	= 0	.29
1		.00	1		.51
2		.00	2		.51
3		.29	3		.51
4		.80	4		.80
5		.80	5		.80
6		.80	6		.80
7		.80	7		.80
8		.00	8		.80
9		.29	9		.80
10		.29	10		.80
11		.57	11		1.07
12		.80	12		1.57
13		.80	13		1.57
14		.80	14		1.57
15		.80	15		1.57

2. For an enhanced storage reference field instruction, add the following address mode time to the execution time given in this appendix.

Addressing Mode	Delta	Additional Execution Time
0	≠ 0	.00
1		.67
2		.28
3		.78

Addressing Mode	Delta	Additional Execution Time
0	= 0	.22
1		.72
2		.72
3		1.50

3. Add .72 microseconds for the following instructions:

   SJI

   ARJ

   SBI

   ANI

   LRI

   ORI

Mnemonic	Definition	Execution Times (μsec)	OP Code			
AAB	Transfer Arithmetic Sum A, Q+M	1.74, 1.91, 2.08, 2.25	0	8	3	8—F
AAM	Transfer Arithmetic Sum A, M	1.74, 1.91, 2.08, 2.25	0	8	2	8—F
AAQ	Transfer Arithmetic Sum A, Q	1.10, 1.34, 1.51, 1.54†	0	8	3	0—7
ADD	ADD A	1.76	8	0 to F	Δ	
ADQ	ADD Q	1.76	F	0 to F	Δ	
ALS	A Left Shift	1.62 + .056 * N	0	F	C/D	0—F
AMr	AND Memory	5.68	0	4	0 to F	0 to F
			A	1	Δ	
AND	AND with A	1.62	A	0 to F	Δ	
ANr	AND Register	5.40	0	4	0 to F	0 to F
			A	0	Δ	
ARr	Add Register	5.40	0	4	0 to F	0 to F
			8	0	Δ	
ARS	A Right Shift	1.62 + .056 * N	0	F	4/5	0—F
ASC	Accumulator Scale	2.88 + .056 * N	0	B	0	A
CAB	Transfer Complement Logical Product A, Q+M	1.63, 1.80, 1.96, 2.13†	0	8	F	8—F
CAM	Transfer Complement Logical Product A, M	1.63, 1.80, 1.96, 2.13†	0	8	E	8—F
CAQ	Transfer Complement Logical Product A, Q	1.18, 1.34, 1.34, 1.51†	0	8	F	0—7
CBP	Clear Breakpoint Interrupt	2.19	0	B	0	7
CCE	Compare Character Equal	6.14	0	4	0—F	0—F
			E	2	Δ	

†For inter-register instructions, the first execution time is for A or Q register destinations, the second time is for M, A and M, or Q and M registers, the third time is for A and Q, and the fourth time for A and Q and M.

Mnemonic	Definition	Execution Times (μsec)	OP Code			
CLF	Clear Field	6.64	0 0 to F	5 0 to F	+ Δ	6
CLR	Clear to Zero	1.18, 1.34, 1.34, 1.51†	0	8	4	0 to 7
CPB	Clear Program Protect	1.72	0	7	0	0
CrE	Compare Register Equal	5.23, 5.46††	0 E	4 0	0 to F Δ	0 to F
DMI	Define Micro Interrupt	3.43	0	B	0	6
DrP	Decrement and Repeat	2.22†††	0	6	††††	0 to F
DVI	Divide Integer	10.48	3	0 to F	Δ	
EAB	Transfer Exclusive OR A, Q, M	1.63, 1.80, 1.96, 2.13†	0	8	7	8 to F
EAM	Transfer Exclusive OR A, M	1.63, 1.80, 1.96, 2.13†	0	8	6	8 to F
EAQ	Transfer Exclusive OR A, Q	1.18, 1.34, 1.34, 1.51†	0	8	7	0 to 7
EIN	Enable Interrupt	1.40	0	4	0	0
EMS	Execute Micro Sequence	6.20†††††	0	B	r, c	2
ENA	Enter A	.95	0	A	Δ	
ENQ	Enter Q	.95	0	C	Δ	
EOR	Exclusive OR with A	1.62	B	0 to F	Δ	
EXI	Exit Interrupt State	1.85	0	E	Δ	

†For inter-register instructions, the first execution time is for A or Q register destinations, the second time is for M, A and M, or Q and M registers, the third time is for A and Q, and the fourth time for A and Q and M.

††For compare instructions, the first execution time listed is for unequal conditions, and the second time is for equal conditions.

†††Add .62 microseconds for the DIP instruction.

††††2, 4, 6, 8, A, C, E

†††††Plus micro-sequence time

Mnemonic	Definition	Execution Times (μsec)	OP Code			
GPE	Generate Character Parity Even	3.92	0	B	0	8
GPO	Generate Character Parity Odd	3.92	0	B	0	9
IIN	Inhibit Interrupt	1.40	0	5	0	0
INA	Increase A	.95	0	9		Δ
INP	Input to A	3.77 min. 15.63 max†	0	2		Δ
INQ	Increase Q	.95	0	D		Δ
JMP	Jump	1.17	1	0 to F		Δ
LAB	Transfer Logical Product A, Q+M	1.63, 1.80, 1.96, 2.13††	0	8	B	8 to F
LAM	Transfer Logical Product A, M	1.63, 1.80, 1.96, 2.13††	0	8	A	8 to F
LAQ	Transfer Logical Product A, Q	1.10, 1.34, 1.34, 1.51††	0	8	B	0 to 7
LCA	Load Character to A	5.80	0 C	4 2	0 to F	0 to F Δ
LDA	Load A	1.62	C	0 to F		Δ
LDQ	Load Q	1.62	E	0 to F		Δ
LFA	Load Field	6.19 + .056 * N	0 0 to F	5 0 to F	0 to F	4 or C Δ
LLB	Load Lower Unprotected Bounds	2.14	0	B	r,o	1
LLS	Long Left Shift	2.30 + .056 * N	0	F	E/F	0 to F
LMM	Load Micro Memory	2.42 + 3.5 * N	0	B	0	1
LRG	Load Registers	13.80	0	B	0	2

†Maximum time is for internal reject

††For inter-register instructions, the first execution time is for A or Q register destinations, the second time is for M, A and M, or Q and M registers, the third time is for A and Q, and the fourth time for A and Q and M.

Mnemonic	Definition	Execution Times (μsec)	OP Code			
LRr	Load Register	5.40	0 C	4 0	0 to F Δ	0 to F
LRS	Load Right Shift	2.30 + .056 * N	0	F	6/7	0 to F
LUB	Load Upper Unprotected Bounds	2.14	0	B	r, o	0
MUI	Multiply Integer	5.62 min. 7.49 max.	2	0 to F	Δ	
NOP	No Operation	1.17	0	F	0 to 1	0 to F
OMr	OR Memory	5.68	$\bar{0}$ D	$\bar{4}$ 1	$\bar{0}$ to $\bar{F}$ Δ	$\bar{0}$ to $\bar{F}$
ORr	OR Register	5.40	0 D	4 0	0 to F Δ	0 to F
OUT	Output from A	3.49 min. 15.63 max.†	0	3	Δ	
QLS	Q Left Shift	1.96 + .056 * N	0	F	A or B	0 to F
QRS	Q Right Shift	1.96 + .056 * N	0	F	2 or 3	0 to F
RAO	Replace Add 1 in Storage	2.22	D	0 to F	Δ	
RTJ	Return Jump	1.69	5	0 to F	Δ	
SAM	Skip if A = -	1.23, 1.52††	0	1	3	0 to F
SAN	Skip if A ≠ +0	1.23, 1.52††	0	1	1	0 to F
SAP	Skip if A = +	1.23, 1.52††	0	1	2	0 to F
SAZ	Skip if A = +0	1.23, 1.52††	0	1	0	0 to F
SBr	Subtract Register	5.47	0 9	4 0	0 to F Δ	0 to F
SCA	Store Character from A	6.53	0 C	4 3	0 to F Δ	0 to F

†Maximum time is for internal reject

††For skip instructions, the first execution time is for no skip, and the second is for skip.

Mnemonic	Definition	Execution Times (μsec)	OP Code			
SEF	Set Field	6.64	0 0 to F	5 0 to F	0 to F Δ	7 or F
SET	Set to 1s	1.18, 1.34, 1.34, 1.51†	0	8	8	0 to 7
SFA	Store Field	7.15 * .056N	0 0 to F	5 0 to F	0 to F Δ	5 or D
SFN	Skip if Field Nonzero	5.85, 6.08††	0 0 to F	5 0 to F	0 to F Δ	3 or B
SFZ	Skip if Field Zero	5.85, 6.08††	0 0 to F	5 0 to F	0 to F Δ	2 or A
SIO	Set/Sample Output or Input	3.88	0	B	0	4
SJE	Subroutine Jump Exit	4.50	0 5	4 0	0 to F Δ	0 to F
SJr	Subroutine Jump	4.67	0 5	4 0	0 to F Δ	0 to F
SLS	Select Stop	1.35	0	0	0	0
SNF	Skip on No Program Protect Fault	1.17, 1.46††	0	1	B	0 to F
SNO	Skip on No Overflow	1.17, 1.46††	0	1	F	0 to F
SNP	Skip on No Storage Parity Error	1.35, 1.46††	0	1	D	0 to F
SOV	Skip on Overflow	1.17, 1.46††	0	1	A	0 to F
SPA	Store A, Parity to A	2.18	7	0 to F	Δ	
SPB	Set Program Protect	1.72	0	6	0	ι
SPE	Skip on Storage Parity Error	1.35, 1.46††	0	1	C	0 to F
SPF	Skip on Program Protect Fault	1.17, 1.46††	0	1	E	0 to F

†For inter-register instructions, the first execution time is for A or Q register destinations, the second time is for M, A and M, or Q and M registers, the third time is for A and Q, and the fourth time for A and Q and M.

††For skip instructions, the first execution time is for no skip, and the second is for skip.

Mnemonic	Definition	Execution Times (μsec)	OP Code			
SPS	Sample Position Status	3.94	0	B	0	5
SQM	Skip if Q = -	1.23, 1.52†	0	1	7	0 to F
SQN	Skip if Q ≠ +0	1.23, 1.52†	0	1	5	0 to F
SQP	Skip if Q = +	1.23, 1.52†	0	1	6	0 to F
SQZ	Skip if Q = +0	1.23, 1.52†	0	1	4	0 to F
SRG	Store Registers	12.59	0	B	0	3
SrM	Skip if Register Negative	1.91, 2.02†	0	0	3, 7 B, F	0 to F
SrN	Skip if Register Nonzero	1.91, 2.02†	0	0	1, 5 9, D	0 to F
SrP	Skip if Register Positive	1.91, 2.02†	0	0	2, 6 A, E	0 to F
SRr	Store Register	5.51	0 C	4 1	0 to F	0 to F Δ
SrZ	Skip if Register Zero	1.91, 2.02†	0	0	0, 4 8 C	0 to F
STA	Store A	1.69	6	0 to F		Δ
STQ	Store Q	1.69	4	0 to F		Δ
SUB	Subtract	1.76	9	0 to F		Δ
SWN	Skip if Switch not Set	1.12, 1.40†	0	1	9	0 to F
SWS	Skip if Switch Set	1.12, 1.40†	0	1	8	0 to F
TCA	Transfer Complement A	1.18, 1.34, 1.34, 1.51††	0	8	6	0 to 7
TCB	Transfer Complement Q+M	1.46, 1.62, 1.79, 1.96††	0	8	5	8 to F

† For skip instructions, the first execution time is for no skip, and the second is for skip.

†† For inter-register instructions, the first execution time is for A or Q register destinations, the second time is for M, A and M, or Q and M registers, the third time is for A and Q, and the fourth time for A and Q and M.

Mnemonic	Definition	Execution Times (μsec)	OP Code			
TCM	Transfer Complement M	1.46, 1.62, 1.79, 1.96†	0	8	4	8 to F
TCQ	Transfer Complement Q	1.18, 1.34, 1.34, 1.51†	0	8	5	0 to 7
TRA	Transfer A	1.18, 1.34, 1.34, 1.51†	0	8	A	0 to 7
TRB	Transfer Q+M	1.46, 1.62, 1.79, 1.96†	0	8	9	8 to F
TRM	Transfer M	1.46, 1.62, 1.79, 1.96†	0	8	8	8 to F
TRQ	Transfer Q	1.18, 1.34, 1.34, 1.51†	0	8	9	0 to 7
XFr	Transfer Register	2.47††	0	7	0, 1 to 7	1 to 7

†For inter-register instructions, the first execution time is for A or Q register destinations, the second time is for M, A and M, or Q and M registers, the third time is for A and Q, and the fourth time for A and Q and M.

††Add .67 microseconds for XFI instruction.

COMMENT SHEET

MANUAL TITLE CYBER 18 System Summary

PUBLICATION NO. 96767850 REVISION 02

FROM NAME:

BUSINESS ADDRESS:

COMMENTS: This form is not intended to be used as an order blank. Your evaluation of this manual will be welcomed by Control Data Corporation. Any errors, suggested additions or deletions, or general comments may be made below. Please include page number to which your comment applies.

CUT ALONG LINE

STAPLE

STAPLE

FOLD

FIRST CLASS
PERMIT NO. 333

LA JOLLA CA.

BUSINESS REPLY MAIL

NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

POSTAGE WILL BE PAID BY

CONTROL DATA CORPORATION
PUBLICATIONS AND GRAPHICS DIVISION
4455 EASTGATE MALL
LA JOLLA, CALIFORNIA 92037

CUT ALONG LINE

FOLD

STAPLE

STAPLE