Burroughs

DC 1100
DC 1200

Systems

REFERENCE MANUAL

Burroughs

DC 1100/DC 1200 SYSTEMS

REFERENCE MANUAL

Burroughs Corporation
Detroit, Michigan 48232

$4.00

Printed in U.S. America

5-70

1046380

The information contained herein is subject to change without notice. Revisions may be issued to advise of such changes and/or additions.

Correspondence regarding this document should be forwarded using the Remarks Form at the back of the manual, or may be addressed directly to Systems Documentation, Sales Technical Services, Burroughs Corporation, 6071 Second Avenue, Detroit, Michigan 48232.

TABLE OF CONTENTS

TABLE OF CONTENTS (cont)

TABLE OF CONTENTS (cont)

TABLE OF CONTENTS (cont)

TABLE OF CONTENTS (cont)

TABLE OF CONTENTS (cont)

TABLE OF CONTENTS (cont)

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATION (cont)

LIST OF ILLUSTRATION (cont)

LIST OF TABLES

LIST OF TABLES (cont)

LIST OF TABLES (cont)

INTRODUCTION

Data Communications is one of the fastest growing areas in the data processing industry. The need for peripheral devices at remote locations, as well as the ability of a central processor to interface to many terminal devices has created the need for a low-cost remote processor to perform as a peripheral controller and/or a terminal concentrator. The DC 1000 Series represents Burroughs response to this challenge.

Two important characteristics of the DC 1000 Series are internal programmatic control and soft terminal and central systems interface. These features enable the DC 1000 Series to be used for off-line processing, in addition to the normal controller and concentrator functions. The DC 1000 Series is currently available in two models:

The DC 1100 Remote Peripheral Controller provides for directly connected card, paper tape and line printing devices, while maintaining a high speed communications link with a central computer system. This configuration provides the necessary hardware for data collection, remote compilation and other batch-type operations.

The DC 1200 Remote Concentrator/Controller, by the use of core memory and a high speed communications line to the central system, provides for concentration of up to 64 low-speed communications lines. In addition, soft terminal interface provides maximum flexibility in interfacing Burroughs and competitive terminal devices to the DC 1200 System.

This reference manual defines the following:

a. Features, equipment availability and allowable configurations.

b. Functional description of the hardware components that comprise the system.

c. Functional description of the software that is available for the DC 1000 Series.

d. Description of data communications hardware and software.

e. Operating characteristics of the system and its peripheral devices.

f. Instruction and code formats.

SECTION 1
SYSTEMS DESCRIPTION

GENERAL.

The DC 1000 (figure 1-1) is a general purpose digital computer designed for use as the controlling element in data acquisition, process control, communications, and other real-time applications. It features powerful computing ability, easy interfacing, integrated circuit reliability, compact size, and low cost. A versatile register structure offers two complete processing capabilities (not simultaneous). This system allows the servicing of input/output operations without affecting the main program; or allows the programs to timeshare the processor; one in the foreground using I/O and the other in the background.

Figure 1-1. DC 1000

GENERAL CHARACTERISTICS.

Some of the DC 1000 general characteristics are:

a. Variable operand precision of 8, 16, 24, or 32 bits.
b. Two hardware register sets.
c. Direct addressing to 4096 bytes.
d. Indirect and indexed addressing.
e. Comprehensive instruction set.
f. Expandable memory to 32,768 bytes.
g. 1.5 microsecond memory cycle time.
h. Addressing to 8-, 16-, or 32-bit level.
i. High-speed direct I/O channel.

GENERAL SPECIFICATIONS OF THE DC 1000.

Some of the specifications for the units and functions are:

a. Memory.

1) Magnetic core.
2) Eight-bit bytes.
3) 1.5 microsecond full cycle.
4) 500 nanosecond access time.
5) 4096 bytes minumum.
6) Expandable to 32,768 bytes.

b. Arithmetic.

1) Parallel.
2) Binary.
3) Fixed point.
4) Two's complement.

c. Operand word length options.

1) 8 bits.
2) 16 bits.
3) 24 bits.
4) 32 bits.

d. Addressing.

1) Direct addressing to 4096 bytes.
2) Two levels of indirect addressing.
3) Index addressing with hardware registers.

e. Instructions.

1) Over 50 basic one, two, and three-byte instructions.
2) Over 500 useful register-to-register operations.

f. Registers.

1) 14 registers available to the programmer for use as counters, accumulators, and indexes.

2) Four non-program accessible registers.

g. Dual environment.

1) Two sets of hardware registers eliminate the need to save and restore the register contents when transferring between program routines in different environments.

h. Processor.

1) Parallel arithmetic operations.
2) Parallel word transfers.
3) Variable word length.

i. DMA port.

1) Direct access port for external equipment communication handling up to 666,000 bytes per second.

j. Interrupt system.

1) 11 interrupt lines.
2) Programmable priority scheme.

k. Console.

1) Display and data entry switches.
2) Single step control.
3) Three sense switches.

l. Software.

1) An assembler that runs on the DC 1000.
2) An assembler that runs on the B 3500.
3) Debugging aids.
4) Subroutine library.
5) Test programs.

CONFIGURATIONS.

The DC 1000 is designed to integrate with the Burroughs 500 family using the following configurations:

a. Remote Controller.
b. Remote Concentrator.
c. Remote Concentrator/Controller.

DC 1100 REMOTE CONTROLLER.

The Remote Controller configuration provides an inexpensive means for an installation to enter a remote/batch operation. The DC 1000, with its modular concepts, allows minimal hardware for initial operation with built-in expandability for the future. The controller environment provides the necessary hardware for data collection, remote compilation, and other batch-type operations.

The software necessary to operate a Remote Controller is briefly described in section 4.

The DC 1100 configuration consists of a processor and combination of the peripheral units shown in figure 1-2. The specific configurations and designations are defined in the following paragraphs.

Figure 1-2. DC 1100 Remote Controller

DC 1101 CONFIGURATION. The basic DC 1101 Remote Controller (figure 1-3) consists of:

a. DC 1000 CPU.
b. Model 33 ASR.

Figure 1-3. Basic DC 1101 Remote Controller

DC 1102 CONFIGURATION. This configuration is illustrated in figure 1-4 and includes:

a. DC 1000 CPU.
b. Model 33 ASR.
c. 200 cpm Card Reader.

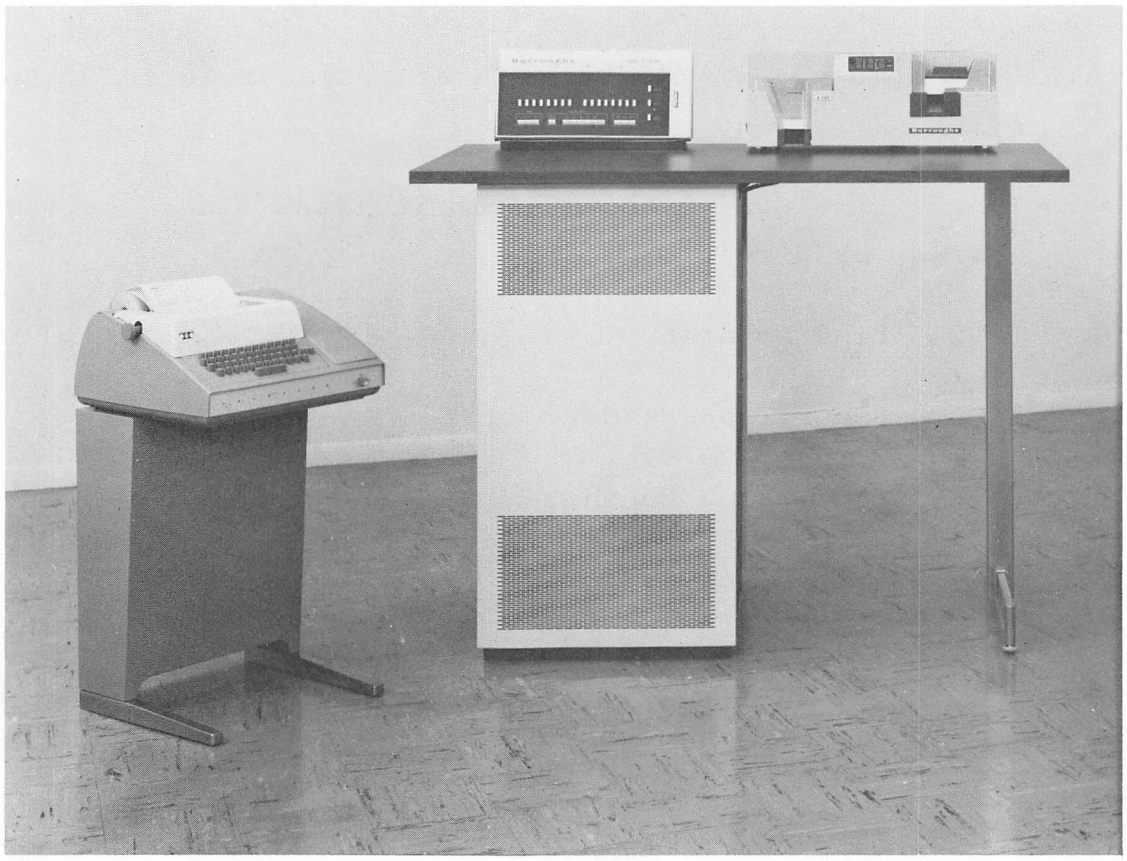


Figure 1-4. DC 1102 Configuration

DC 1103 CONFIGURATION. This configuration is illustrated in figure 1-5 and includes the following components:

a. DC 1000 CPU.
b. Model 33 ASR.
c. 200 cpm Card Reader.
d. 100 cpm Card Punch.

In addition, the following options are avaliable on the DC 1100 series:

a. Core memory: 4096 byte modules, expandable to 32,768 bytes.

b. Single Line Control data communications line: switched, leased, or direct connect.

c. B 9245 Line Printer.

d. B 9120 Paper Tape Reader.

e. B 9220 Paper Tape Punch.

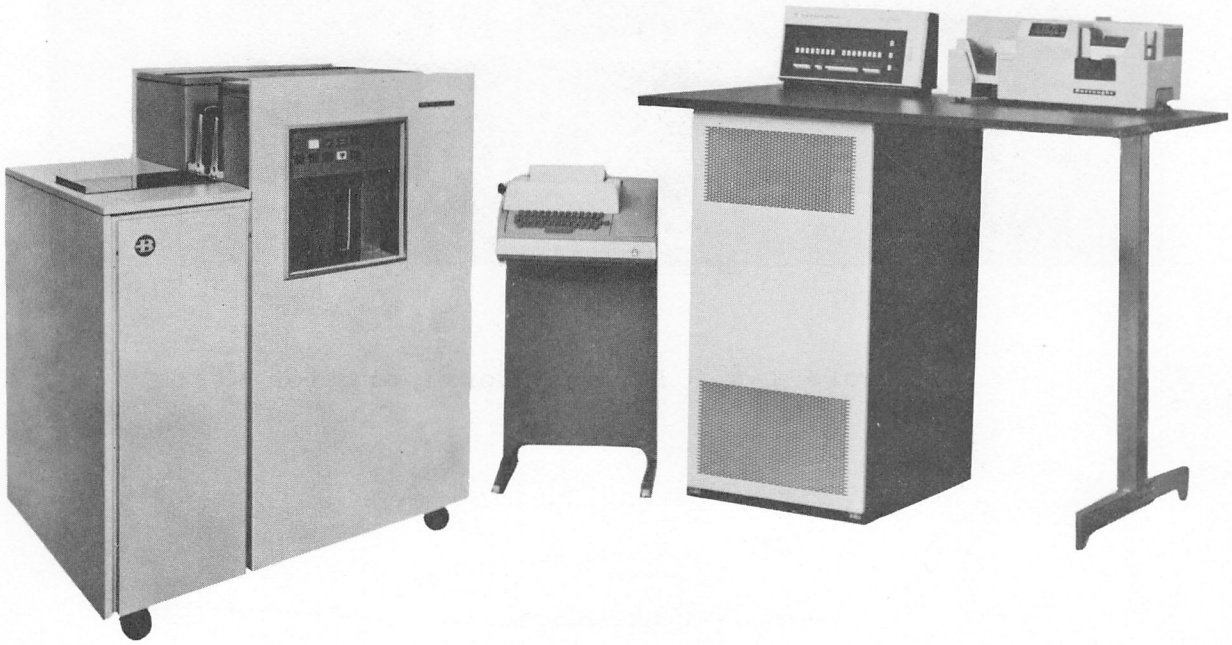


Figure 1-5. DC 1103 Configuration

DC 1200 REMOTE CONCENTRATOR.

The Remote Concentrator provides a data communications environment the tools to relieve the central system of data communications chores such as: polling, selecting, acking, naking, etc. The configuration has the capability of funneling a data-communications network (up to 64 lines) and transmitting the information to the central system over a single line (figure 1-6).

The Remote Concentrator has the capability of handling the following terminal devices:

a. Teletype Models 28, 33 and 35.
b. TC 500/TC 700.
c. B 9351 CRT.
d. B 9352 CRT.

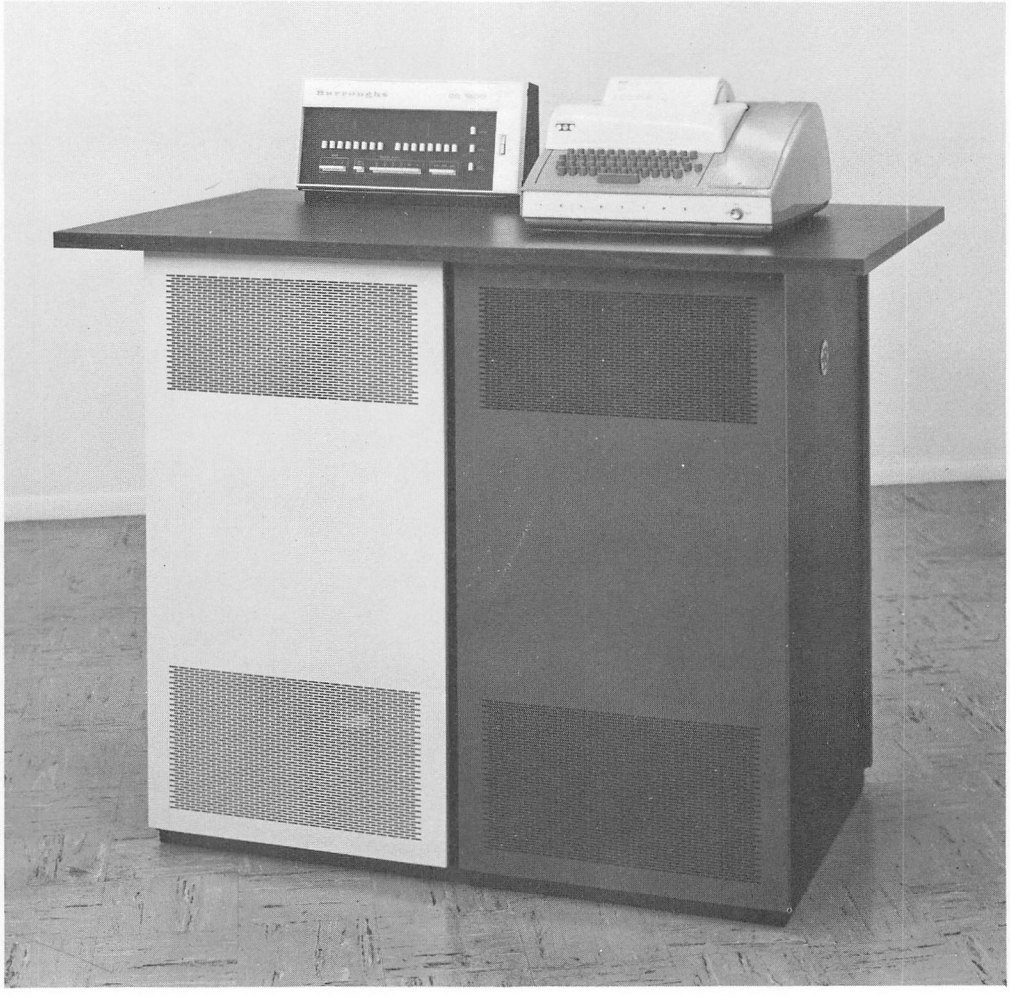


Figure 1-6. DC 1200 Remote Concentrator

NOTE

Terminals will be added to this list as they become available and their interface has been checked.

DC 1201 CONFIGURATION. The basic DC 1201 Remote Concentrator (figure 1-7) consists of the following components:

a. DC 1000 CPU.
b. Basic Multi-Line Control (16 lines).
c. Model 33 ASR.

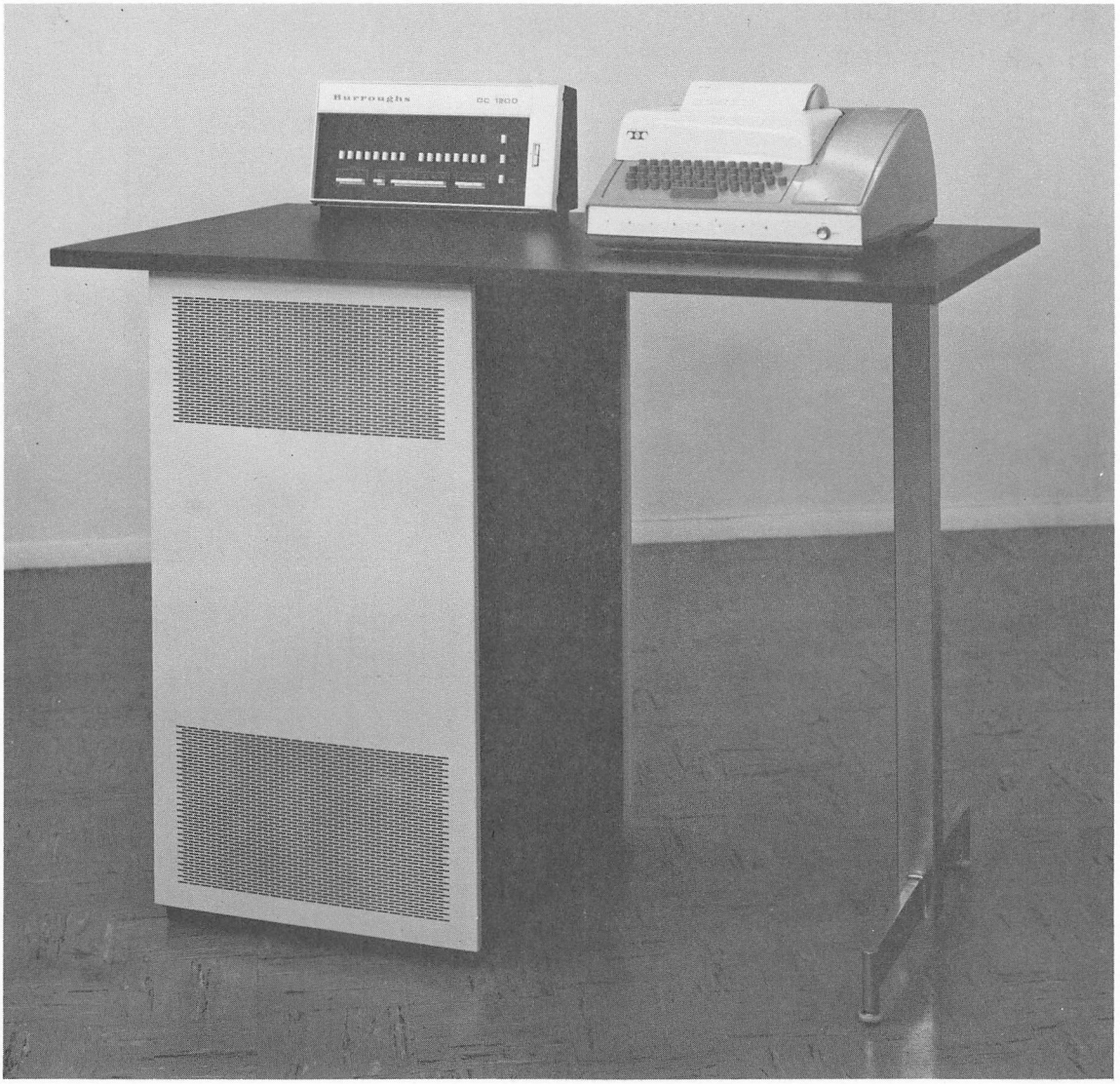


Figure 1-7. DC 1201 Remote Concentrator

DC 1202 REMOTE CONCENTRATOR/CONTROLLER.

This configuration is a combination of the previously mentioned Remote Controller and Remote Concentrator. It is limited to the extent that only two peripheral units (instead of three) and 32 data communications lines (instead of 64) can be handled.

This configuration provides more flexibility than either of the others. For example, it can be used as a concentrator/controller at all times, or it may be used as a controller during certain hours of the day and as a concentrator during other hours. Figure 1-8 illustrates a DC 1202 with a printer option.

Figure 1-8. DC 1202 With Printer Option

DC 1202 CONFIGURATION. The basic DC 1202 Concentrator/Controller (figure 1-9) includes the following:

a. DC 1000 CPU.
b. Basic Multi-Line Control (16 lines).
c. Model 33 ASR.
d. 200 cpm Card Reader.

Figure 1-9. DC 1202 Basic Configuration

DC 1203 CONFIGURATION. This configuration is shown in figure 1-10 and includes the following:

a. DC 1000 CPU.
b. Basic Multi-Line Control (16 lines).
c. Model 33 ASR.
d. 200 cpm Card Reader.
e. 100 cpm Card Punch.

In addition, the following options are available on the DC 1200 series:

a. Core memory: 4096 byte modules, expandable to 32,768 bytes.

b. Four extensions to the basic Multi-Line Control (16 additional lines per extension).

c. Line Adapters in groups of four or eight depending on type.

d. B 9245 Line Printer.

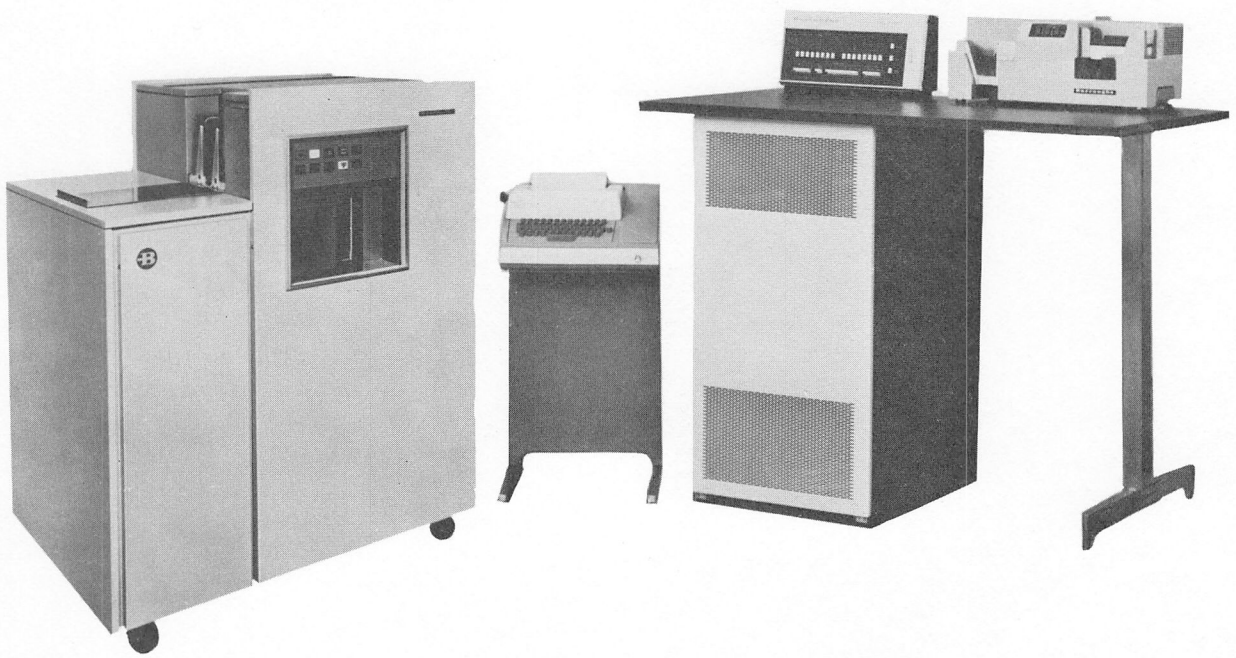


Figure 1-10. DC 1203 Configuration

PERPHERAL UNITS.

The peripheral units provide the input/output facilities for the DC 1000 systems. Three of the following units may be used in any configuration with the remote controller; two of the following units may be used with the controller/concentrator:

a. 200 cpm Card Reader.
b. B 9120 Paper Tape Reader (500-1000 cps).
c. 100 cpm Card Punch.
d. B 9220 Paper Tape Punch (100 cps).
e. B 9245 Line Printer (300 lpm).

The functional charactertistics of these peripheral units are discussed in section 3.

SECTION 2
HARDWARE FUNCTIONAL DESCRIPTION

GENERAL.

The following section of this manual contains descriptions of the hardware components that comprise the DC 1000 system. Functional descriptions are given for each of the following:

a. Central processor.
b. Memory.
c. Direct memory access port.
d. Input/output channel.
e. Control panel.
f. Interrupt logic.
g. Registers.
h. Peripheral controllers.

All references to the DC 1000 are meant to include all of the above. References to a specific component of the system is made in that manner, i.e., central processor unit.

The data communications hardware is covered in section 4.

CENTRAL PROCESSOR.

The DC 1000 Central Processor employs parallel arithmetic and parallel word transfers for maximum speed. Accumulator operations such as Add and Store are performed with variable word lengths from eight bits to 32 bits. This allows the programmer to utilize the precision required by the data rather than fitting the data to the precision of the machine.

The DC 1000 central processing unit block diagram is shown in figure 2-1.

The central processor unit can operate in two environments:

a. Environment 1 is interruptable and can perform the following functions:

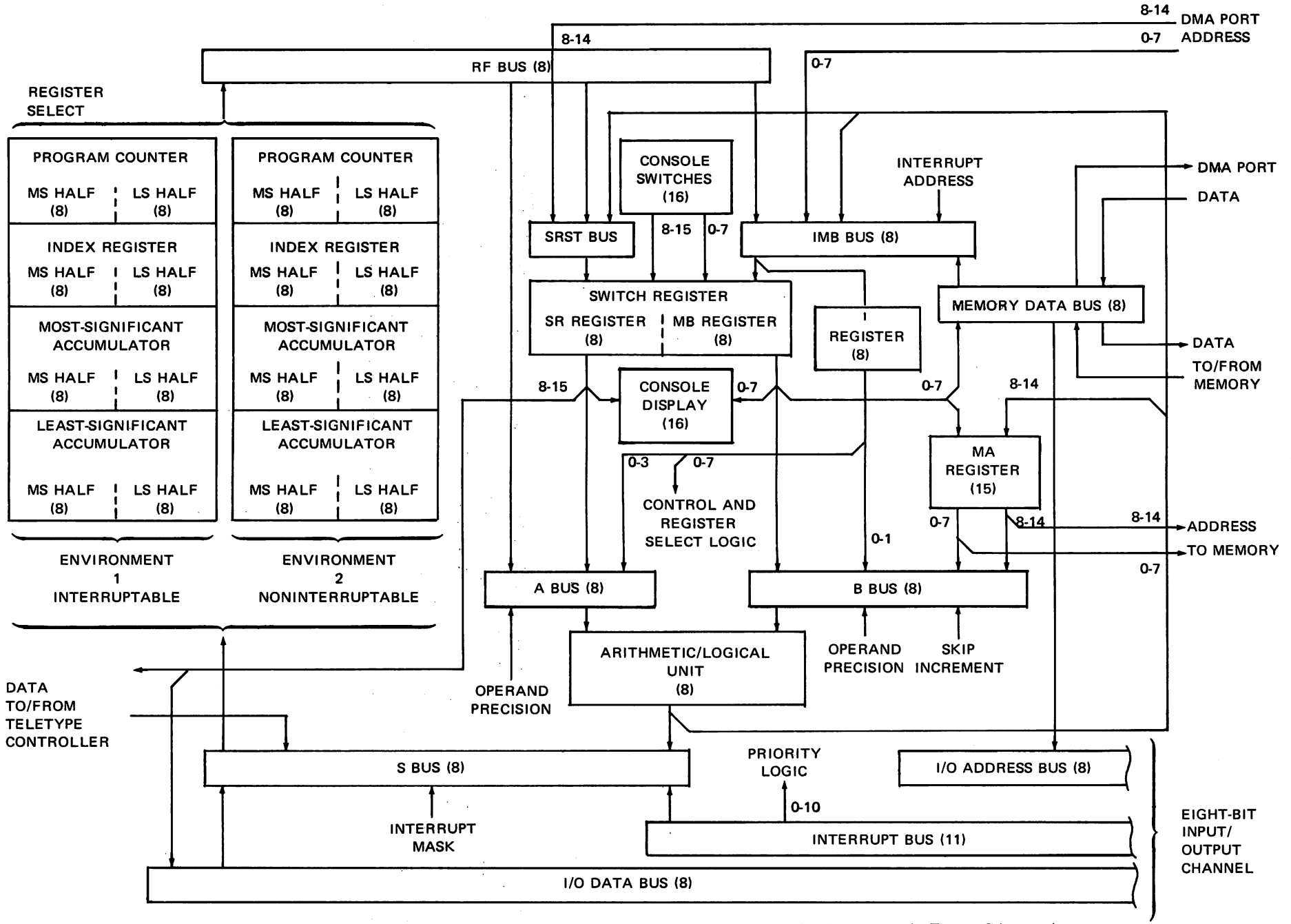


Figure 2-1. DC 1000 Central Processor Register and Bus Structure

1) Line control.

2) Message editing and formatting.

3) Interpretation of heading information.

4) Preparation of buffer areas for data receipt or transmission.

b. Environment 2 is non-interruptable and can perform the following functions:

1) Input/output transfers.

2) Storage and retrieval of buffer areas.

3) Checks for control characters.

Each of the environments may be either current or non-current with changes between environments requiring only 1.5 microseconds. The flexibility of environment is due to the duplication of registers. Each of the environments has its own:

a. Accumulator.
b. Program Address register.
c. Index register.

These registers are detailed later in this section.

MEMORY.

The DC 1000 incorporates a high speed, random access memory of modular construction. It is a magnetic core memory with coincident current read and write control. Each memory module has a capacity of 4096, eight or nine-bit words. The nine-bit memory uses the ninth bit for parity. Up to two memory modules can be accommodated in the DC 1000 main frame. The memory may be expanded to a maximum of 32,768 bytes by using an expansion chassis. Memory allocation is shown in table 2-1.

DIRECT MEMORY ACCESS PORT.

The direct memory access port (DMA) of the DC 1000 allows peripheral units to interrupt the central processor without disturbing the operational registers. The interrupted program continues at the conclusion of the DMA data transfer.

The DMA port permits data transfers to and from the peripheral units at rates up to 666,000 bytes per second. This operation is ideally suited for the high-speed data transfers by magnetic tape or disk units, or from sampling units.

Characteristics of the DMA port are:

a. The peripheral unit using this facility can operate asynchronously with feedback control signals.

b. The peripheral unit provides memory addresses.

c. The peripheral unit can use consecutive memory cycles for a high data-burst rate.

Table 2-1
Memory Sector Allocation

Location	Sector	Comment
0000 – 1023	0	Standard memory module accessible by direct address from anywhere in memory.
1024 – 2047	1	
2048 – 3071	2	

Table 2-1 (cont)
Memory Sector Allocation

Location	Sector	Comment
3072 – 4095	3	Standard memory module accessible by direct address from anywhere in memory.
4096 – 5119	4	Optional memory modules (4096 bytes each).
5120 – 32,767	5-31	

OPERATION.

A peripheral unit using the DMA port must provide address data and control signals compatible with the DC 1000 logic.

The DMA port can operate in one of two modes depending on the data transfer rate of the peripheral units. For peripheral units operating at less than 93,000 bytes per second, a DMA request sent to the DC 1000 is acknowledged at the completion of the instruction process. The peripheral unit then provides the address and data words (if required). The external data word is then stored or an accessed word is sent to the peripheral unit. The maximum access rate is determined by the instruction with the longest execution time. For most programs, the worst-case instruction execution time is about nine microseconds (table 2-2).

Table 2-2
Instruction Access Rates

Number of Peripheral Units	Bytes per Second	
	Low Speed	High Speed
1	93,000	660,000
2	81,500	333,000
3	73,000	222,000
4	65,000	167,000

For peripheral units operating at speeds greater than 93,000 bytes per second, an I/O command from the DC 1000 is sent to the requesting peripheral unit before the data transfer is started. The I/O command sustains the DMA request signal throughout the data transfer. At the completion of the transfer, the DMA request signal is made inactive allowing the interrupted program to continue. In this way, every memory cycle during the data transfer is reserved for the DMA (table 2-2).

The following priorities exist when a DMA request and an interrupt require a response from the DC 1000 at the completion of the same instruction:

a. A power-fail interrupt has priority over a DMA request. In addition, a DMA data transfer is always interrupted by a power-fail interrupt at the end of the data access in process.

b. A DMA request has priority over all interrupts other than power fail. A DMA data transfer in process cannot be

interrupted by interrupts other than power fail. If the DC 1000 includes a parity option, parity error detection is inhibited during a DMA operation. Parity is formed when writing into memory, but is not checked when accessing memory.

INPUT/OUTPUT (I/O).

EIGHT-BIT INPUT/OUTPUT CHANNEL.

The eight-bit I/O channel provides for efficient and flexible communications with peripheral units. Numerous peripheral units can be controlled with a minimum of control logic.

The eight-bit I/O channel communicates with the peripheral units via a data bus and an address bus. The data bus comprises eight bidirectional data lines. The address bus comprises three control lines that specify one of eight control functions or sense inquiries, and five device address lines that specify one of 31 possible devices to communicate with the central processor, (device address 0 is reserved for the central processor).

EIGHT-BIT PROGRAM CONTROLLED INPUT/OUTPUT.

The basic machine provides five types of I/O operations:

a. Sense. The status of a selected peripheral controller is interrogated by the DC 1000 under program control.

b. Function control. A control code is transferred under program control to a peripheral controller.

c. Byte transfer in. A single byte of data is transferred under program control from a peripheral controller to the least significant eight bits of the current accumulator.

d. Byte transfer out. A single byte of data is transferred under program control to a peripheral controller from the least significant eight bits of the current accumulator.

e. Interrupt. A peripheral controller transmits an interrupt to the DC 1000 to initiate special program subroutines.

All of the I/O instructions have an associated order code and device address that are transmitted on an eight-bit address bus to the peripheral units.

FORMAT.

Bits	7	6	5	4	3	2	1	0
	ORDER			DEVICE				

A unique device address for each peripheral unit is stored in bits 0 to 4. Bits 5 to 7 identify the function to be performed or in some other way qualify the I/O command.

INTERFACE.
The standard I/O bus consists of a data bus, address bus, interrupt bus, and six control lines.

DATA BUS. The data bus is an eight-bit, parallel, bidirectional I/O channel. It is used to transmit data from the computer to peripheral devices. In turn, the bus is used by these devices to transmit data to the computer.

ADDRESS BUS. The address bus is an eight-bit output bus. It is used to transmit device address and order code to the peripheral device. Five bits determine the device address and three bits specify up to eight control functions or sense inquiries.

INTERRUPT BUS. The interrupt bus consists of 11 interrupt lines. Highest priority is assigned by hardware to three lines. The eight remaining lines are assigned priorities by the software.

Control signals FUNS-, SENS-, BTIS-, and BTOS-, are generated during computer-initiated I/O functions. They are mutually exclusive and determine which of four operations is in process. Internal gating provides a nominal signal duration of 500 nano-

seconds. The four signals are defined as follows:

a. FUNS- Functional control.
b. SENS- Sense external device.
c. BTIS- Byte transfer in.
d. BTOS- Byte transfer out.

FUNS-. The FUN instruction sends control signals to the peripheral units. The least significant part of the address portion of the instruction determines which of three peripheral units is to receive the control signal. One of eight control functions can be selected by coding the most significant part of the address portion of the instruction.

SENS-. The SEN instruction interrogates the status of a peripheral unit. The least significant part of the address portion of the instruction determines which of the 31 units is to be interrogated. One of the eight sense functions is selected by coding the most significant part of the address portion of the instruction.

BTIS-. The BTI instruction transfers data from a peripheral unit to the computer. The operation is recognized as an input transfer by the presence of the BTIS- signal. The address portion of the instruction determines which peripheral unit will send the data. The three most significant bits of the address portion of the instruction may be used to distinguish several types of input bytes from a single unit.

BTOS-. The BTO instruction transfers data from the computer to the peripheral units. The operation is recognized as an output transfer by the presence of the BTOS- signal. The address portion of the instruction determines the peripheral unit which is to receive the data. The three most significant bits of the address portion of the instruction may be used to distinguish several types of output bytes to a single unit.

CONTROL PANEL.

The DC 1000 main frame includes a functional control panel (figure 2-2). The control panel houses all of the switches and indicators required for the operation of the system.

DATA DISPLAY INDICATORS.

The 16 binary indicators that consitute the data display are used by several sources. The REGISTER SELECT switches permit the following to be displayed:

a. Memory contents.
b. Accumulators.
c. Index registers.
d. Program counters.

These indicators also function with their corresponding data entry switch when manually inserting data into the DC 1000.

DATA ENTRY SWITCHES.

These 16 binary switches correspond to the 16 display indicators. Their function is to manually load a register for data entry into memory or one of the operational registers. Like the data display indicators, their function depends on the setting of the REGISTER SELECT switches.

RESET DISPLAY SWITCH.

Activating this switch clears the data display indicators to all zeros.

OVERFLOW INDICATORS 01 AND 02.

These indicators reflect the status of the arithmetic overflow flip-flops in each environment (interruptable or non-interruptable).

RUN SWITCH.

Activating this switch initiates instruction execution in the DC 1000 starting with the "current" program counter location.

RUN INDICATOR.
The RUN indicator illuminates when the machine is actively executing an instruction.

STEP SWITCH.
This switch causes the DC 1000 to leave the run condition. Each switch activation causes the execution of the next instruction.

This switch is normally used to single-step instructions for hardware or software debugging.

STEP INDICATOR.
This indicator illuminates when the machine is not in a run condition. It also indicates that the control panel switches and indicators are active and may be used for display and entry.

RESET SWITCH.
Momentary activation of this switch initializes all of the control flip-flops in the machine. The environmental status is returned to interruptable (environment 1) and the operand precisions are set to eight bits.

REGISTER SELECT SWITCHES.
These seven switches determine which register is being actively displayed or which register will accept manual entry through the data ENTRY switches.

Table 2-3
Register Select Switches

Switch	Description
EVN-C/EVN-N	Environmental control switch (current, non-current).
A/B	Most significant accumulator.

Table 2-3 (cont)
Register Select Switches

Switch	Description
C/D	Least significant accumulator.
P/Q	Program counter.
X/Y	Index register.
MA	Memory address register (enter only).
M	Memory data register.

When a register switch is down, the contents are displayed by activating the DISPLAY switch. The contents may be altered by clearing the display through the use of the RESET DISPLAY switch; keying in the new data through the data entry switches, and pressing ENTER which routes the data to the selected register. A relationship of the environmental control switch (ENV-C/ENV-N) and the other register select switches is shown in table 2-4.

ENTER SWITCH
The activation of this switch transfers data from the Data Display indicators to the selected register.

SENSE SWITCHES 1, 2, AND 3.
These switches allow program branches and decisions to be made as a result of operator action. The condition of each switch can be interrogated and decisions made in the program based upon their settings.

POWER ON/OFF SWITCH.
This switch controls the application of power to the power supplies

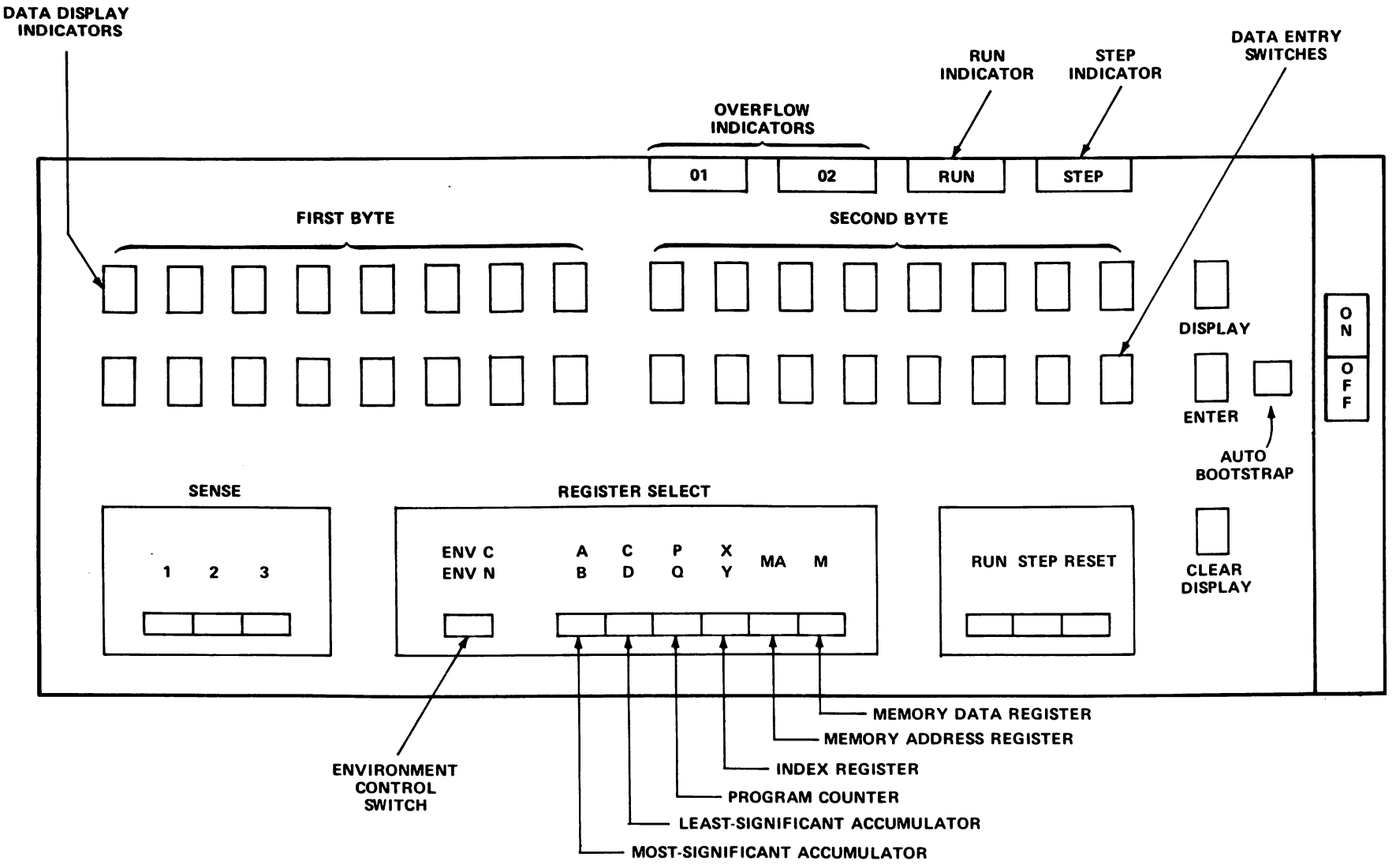


Figure 2-2. DC 1000 Control Panel

within the DC 1000. Activating the ON switch also initializes the DC 1000.

Table 2-4
ENV-C/ENV-N Control of the Register Select Switches

Switch Pressed	ENV-C/ENV-N Switch	
	ENV-C Position	ENV-N Position
A/B	Value of the most significant accumulator in the current environment (16 bits).	Value of the most significant accumulator in the noncurrent environment (16 bits).
C/D	Value of the least significant accumulator in the current environment (16 bits).	Value of the least significant accumulator in the noncurrent environment (16 bits).
P/Q	Value of the program counter in the current environment (16 bits).	Value of the program counter in the noncurrent environment (16 bits).
X/Y	Value of the Index register in the current environment (16 bits).	Value of the Index register in the noncurrent environment (16 bits).
MA	Memory Address register (cannot be displayed).	Memory Address register (cannot be displayed).
M	Memory Data register.	Memory Data register.

NOTE

All switches, with the exception of the environment control switch (ENV-C/ENV-N), are mutually

exclusive. That is, activating a switch resets a previously set switch.

INTERRUPTS.

PRIORITY INTERRUPTS.

The priority interrupt system of the DC 1000 permits rapid response to a variety of external and internal stimuli.

System devices can use the interrupt system to signal alarm conditions, to mark time of day or to alert the central processor for data transfers. In this way, central processor time is used more efficiently than if a program is used to sense device readiness.

Internal interrupts are provided for the power fail/restart and memory parity options.

PRIORITY LEVELS.

There are six priority levels as shown in table 2-5. Internal hardware functions have the two highest priorities followed by the external interrupts.

There are eleven interrupt lines in the I/O cable as shown in figure 2-3. Line 0 has the highest priority.

Lines 3 to 10 all have the same hardware priority, and comprise external interrupt number 3. The programmer assigns priorities within the group by copying the eight interrupt lines into the accumulator (by a copy-interrupt status command) and determining the interrupting device.

Table 2-5

Central Processor Interrupt Addresses

Interrupt	Address (Hexadecimal)
Internal power-fail interrupt (optional).	0000
Internal parity-error interrupt (optional).	0004

Table 2-5 (cont)

Central Processor Interrupt Addresses

Interrupt	Address (Hexadecimal)
External interrupt line 0.	0008
External interrupt line 1.	000C
External interrupt line 2.	0010
External interrupt lines 3 to 10.	0014

OPERATION.

DC latches are used to store the status of the interrupt lines. The interrupt lines are sampled at a time corresponding to the beginning of the fetching of the program counter value for use as the address of the next instruction. See figure 2-3.

The DC latches are cleared once during each timing cycle (1.5 microseconds). This clears the latches of an interrupt that has been acknowledged by the computer and made inactive by the interrupting device.

Following the reset of the latches, they are set to the value of the corresponding interrupt lines. The latches are set in time for the priority logic and address generator to settle before being sampled by the computer.

An interrupt mask is used to enable or disable individual interrupt lines. Bits of the mask are selectively enabled or disabled by a modify interrupt instruction. The mask allows corresponding interrupts to be applied to the priority logic.

Priority logic establishes the relative priority of the interrupt lines. If two or more interrupts occur at once, the line with the highest priority is given precedence. An interrupt line generates an interrupt only if it has not been masked and a higher priority

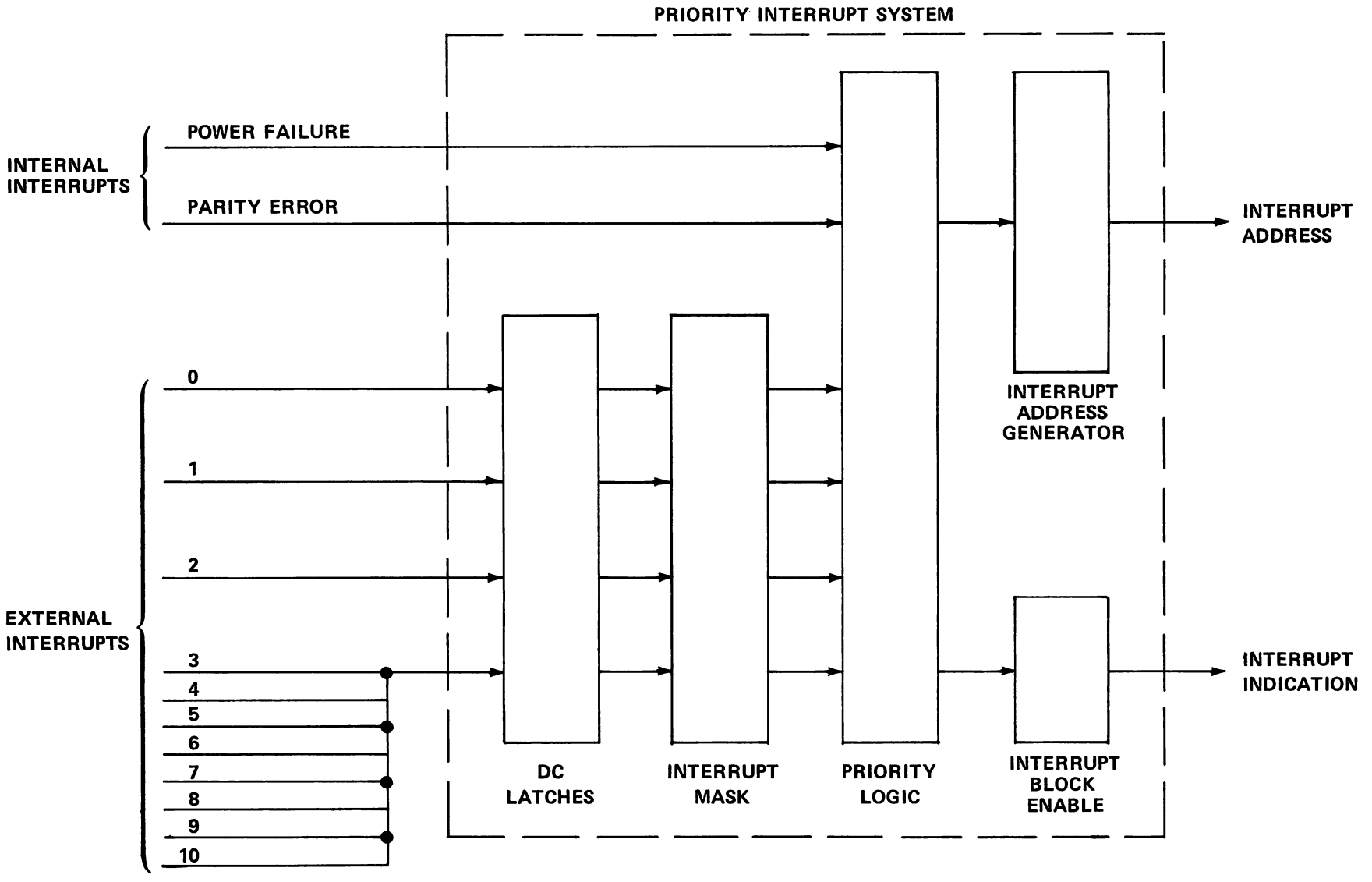


Figure 2-3. Interrupt System Priorities

line is not active.

An interrupt address generator provides the memory address of the interrupt line requesting the interrupt. The address corresponding to each interrupt line is given in table 2-5.

The interrupt system is enabled as a block only after an Interrupt Block Enable instruction (IBE) is executed. The interrupt system is disabled as a block under the following conditions:

a. When power is first applied to the DC 1000.

b. When the front panel RESET switch is activated.

c. When an Interrupt-Block-Disable instruction (IBD) is executed.

d. When an interrupt indication is generated.

With the interrupt system disabled, all interrupts except power failure are inhibited. The power failure interrupt always interrupts the computer regardless of the block enable status or the operating environment.

Individual interrupts are enabled or disabled by the execution of a Modify-Interrupt instruction (MIN).

PROGRAMMING REQUIREMENTS.
The instruction found at the interrupt address must be a Branch and Mark (BRM), Change Status (CST), Change Status and Load Q (CSQ) or Branch Unconditionally (BRU).

Power failure and parity interrupts must be acted upon immediately and are therefore not sensitive to the machine environment. They are recognized even in environment 2. For this reason, they must be serviced only with BRM or BRU instruction at the interrupt address.

INTERRUPT OPERATIONS.

Eleven interrupts are available on the DC 1000 I/O bus. Three of these (E100, E101, and E102) are assigned priority by hardware with E100 having the highest priority. The eight remaining interrupts (E103 to E110) have been assigned equal priority by hardware but with a priority status lower than E102. Priority among the eight low priority interrupts is assigned by software.

Figure 2-4 shows typical uses of the interrupts. Interrupt A is the usual connection with the interrupt driver connected directly to the interrupt source.

The NAND gate used at the input to the driver provides an opportunity to logically OR two interrupt sources as shown at B. Interrupt C is the logical AND of the two interrupt sources. The interrupt shown at D is a logical OR of two interrupt sources from two separate controllers.

Interrupts may be raised at any time but must persist until reset by a response from the computer in the form of a FUN, BTO, or BTI instruction.

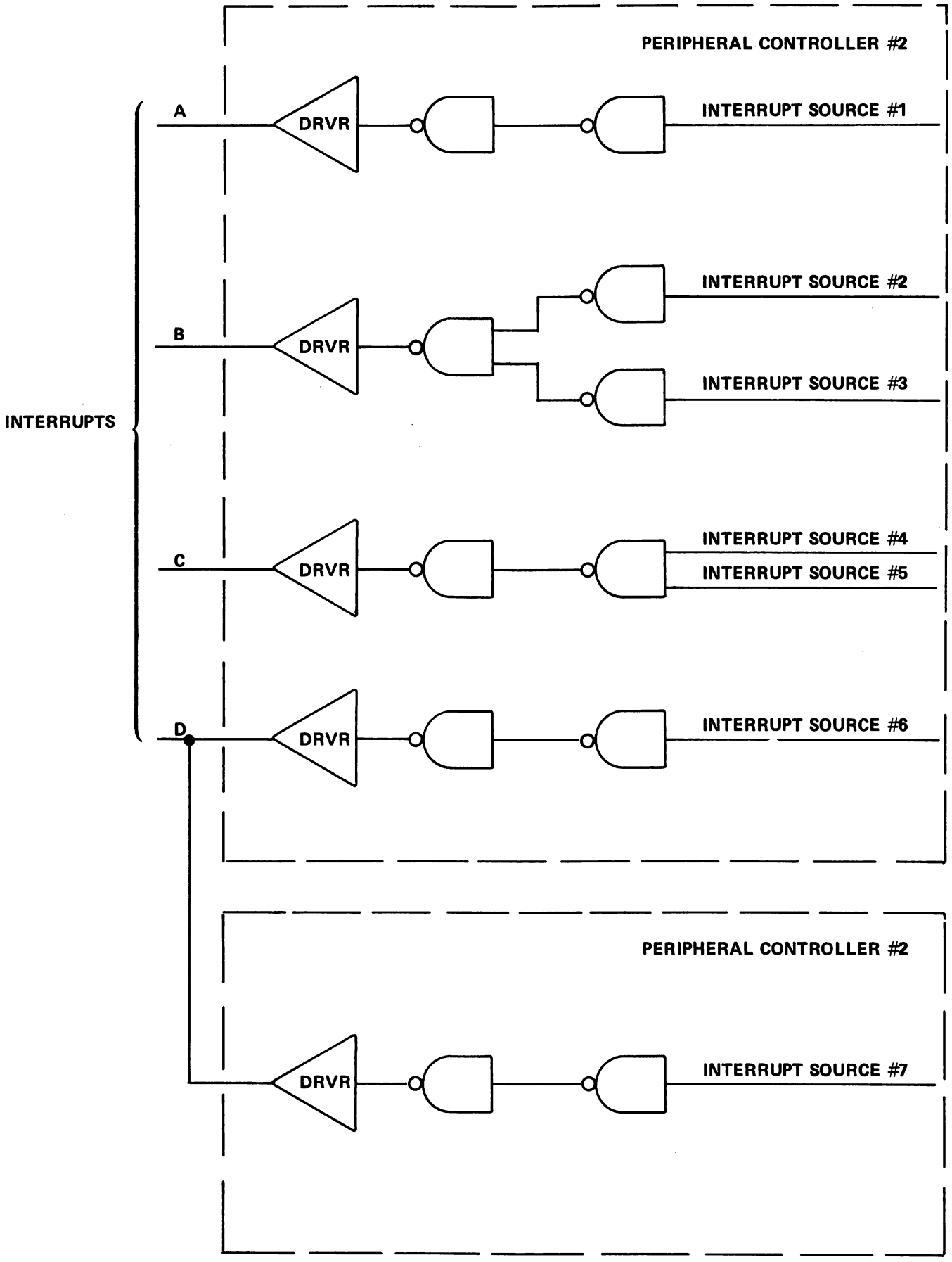


Figure 2-4 Typical Interrupt Application

REGISTERS.

The following registers are accessible to the programmer (table 2-6):

a. Two program counters; 16 bits each.
b. Two index registers; 16 bits each.
c. Two accumulators; 32 bits each.
d. Two overflow indicators; 1 bit each.
e. Two operand-precision registers; 2 bits each.

The registers are divided into two functional environments. Environment 1 is interruptable; environment 2 is non-interruptable. Either environment may be current or noncurrent.

When environment 1 is interrupted, the program counter, index register and accumulator contents are left unchanged while the processor services the demand of a priority using the hardware of environment 2. A single instruction (1.5 microsecond execution time) returns the processor to environment 1 to resume the interrupted program. In this way, an I/O subroutine, with separate computation, can be altered with a background program. If the registers of environment 1 are used for general purpose processing, the programmer has full use of two, 32-bit accumulators, two index registers and two program counters.

Additional registers not available to the programmer are listed in table 2-7.

Table 2-6
Program Accessible Registers

Mnemonic	Description	Register Length
P	Program counter in current environment.	16 bits
X	Index register in current environment.	16 bits
A	Most significant accumulator in current environment.	16 bits

Table 2-6 (cont)
Program Accessible Registers

Mnemonic	Description	Register Length
C	Least significant accumulator in current environment.	16 bits
OPC	Operand precision (word length) in current environment.	2 bits
OVC	Overflow register in current environment.	1 bit
Q	Program counter in non-current environment.	16 bits
Y	Index register in non-current environment.	16 bits
B	Most significant accumulator in non-current environment.	16 bits
D	Least significant accumulator in non-current environment.	16 bits
OPN	Operand precision (word length) in non-current environment.	2 bits
OVN	Overflow register in non-current environment.	1 bit
Z	Pseudo-register containing all zeros.	Precision dependent
F	I/O channel register in peripheral controller.	

Table 2-7

Non-Programmable Registers

Mnemonic	Description	Register Length	Function
I	Instruction register	8 bits	Holds current instruction and address modification.
MB	Memory buffer	8 bits	Temporary storage for accessing memory and for control panel display entry.
S	Temporary operand storage	8 bits	Presents data to the arithmetic unit and for control panel display and entry.
MA	Memory address register	16 bits	Holds the current operand address.

PERIPHERAL CONTROLLERS.

The peripheral controllers provide the logic required to interface the DC 1000 with the following devices:

a. Line printer.
b. Card reader.
c. Card punch.
d. 201 Modem (single line control).
e. Paper tape reader.
f. Paper tape punch.

The controllers are physically located in the DC 1000 main frame as

illustrated in figure 2-5.

The DC 1000 communicates with the peripheral controllers via the standard eight-bit I/O interface. The following paragraphs contain functional descriptions of each of the peripheral controllers.

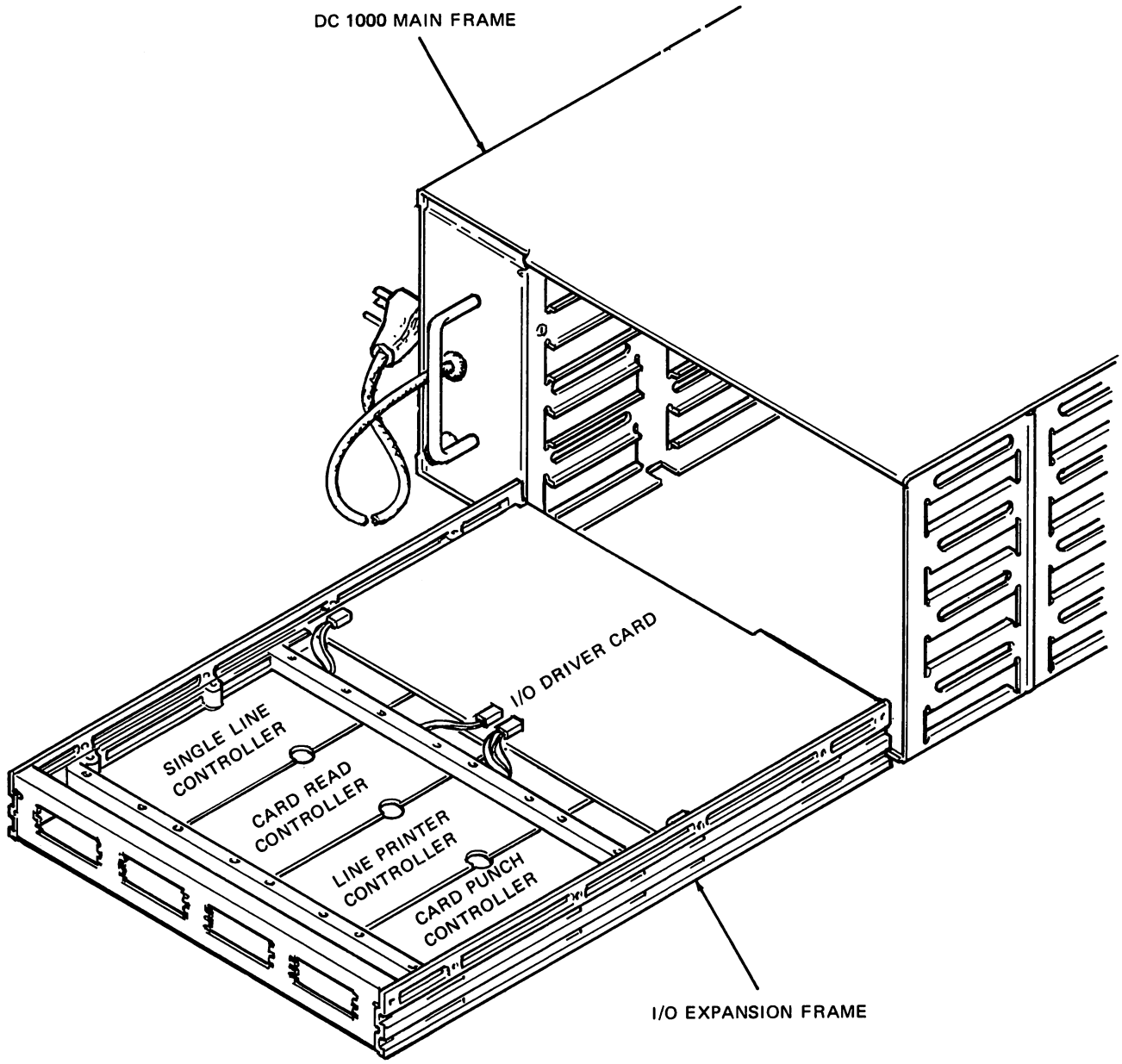


Figure 2-5. DC 1000 Main Frame Controller Installation

LINE PRINTER CONTROLLER.

The Line Printer Controller provides the necessary hardware data buffering and control signals to interface the DC 1000 computer with a fully buffered B 9245 Line Printer. The controller provides data buffering, control and transfer timing, and voltage level shifters. A block diagram of the controller is shown in figure 2-6. The DC 1000, by executing control and data transfer commands to the controller, can fill the line printer data buffer, set up line spacing and format, and activate the print cycle.

FUNCTIONAL DESCRIPTION. Operationally, the computer provides data or character information to the controller by a BTO command as the information is requested. The request is determined by sensing (SEN) controller not busy and is followed by the the character transfer out. The DC 1000, however, is required to keep a count of the character data transfers and when the printer line buffer is filled the CPU must issue the format word to cause the printer to print and space the line.

The printer command is derived from the transfer (BTO) of the format word from the DC 1000 to the controller. This transfer is always required both for format data and the printer command. This is the basic requirement for printing a line and subsequently the execution of paper motion. For paper motion, the printer is instructed to advance one line, two lines, go to top of form, bottom of form, not advance or skip variable.

If desired, any of the above line spacing of paper may be independently accomplished by transferring (BTO) the format word to the controller. However, this mode is operational only if there have been no character transfers to the printer data buffer.

If enabled, interrupts will always occur at the end of the printer activity cycle. The interrupt is always reset by any sense (SEN) command or by transferring the format word.

The line printer controller block diagram is shown in figure 2-6.

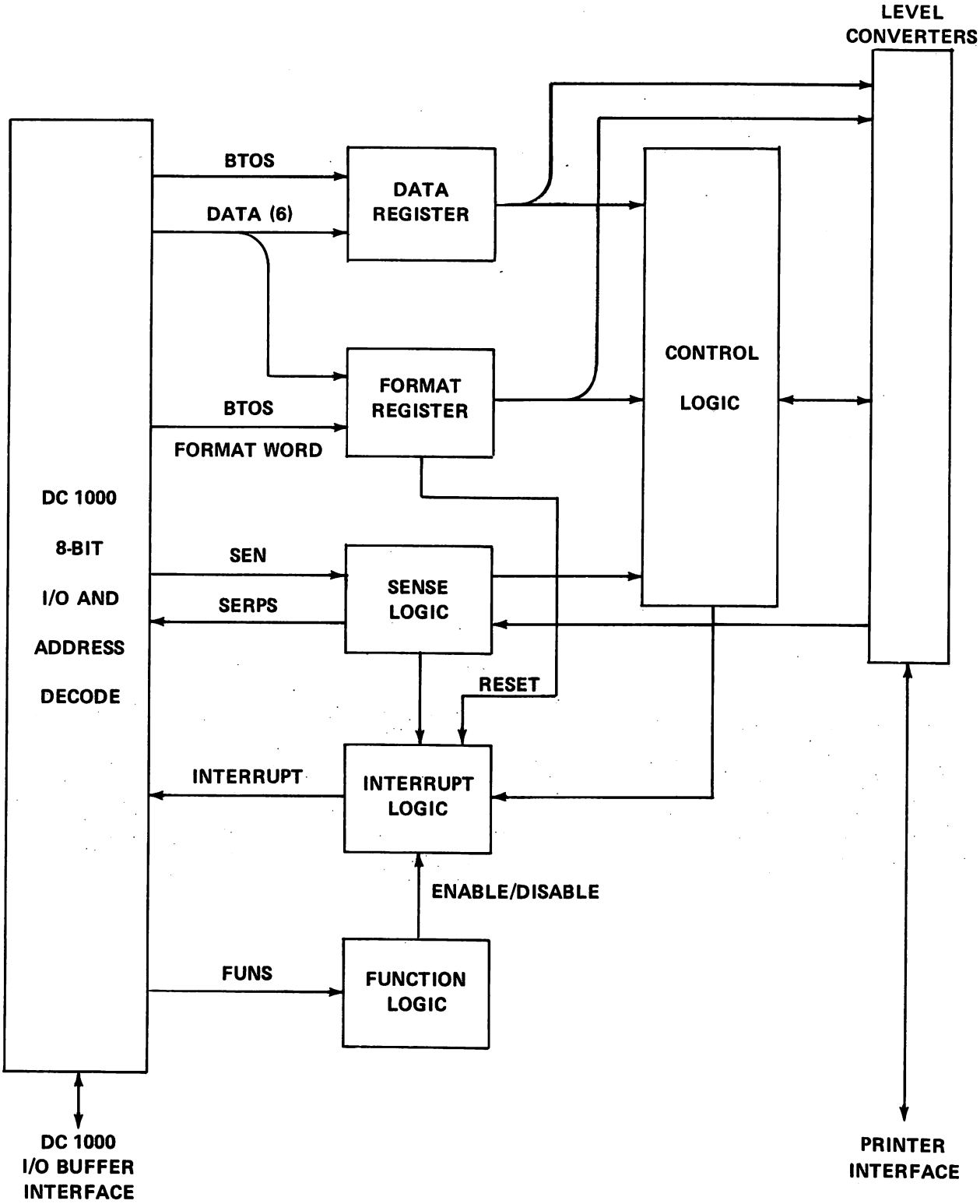


Figure 2-6. Line Printer Controller Block Diagram

Sense logic is included within the controller to allow inspection of the controller/printer status. Function logic (FUN) is also mechanized to enable or disable the interrupt line to the DC 1000.

DC 1000 I/O. The DC 1000 eight-bit I/O signal decoding logic decodes the device address and function/sense codes. The decoded outputs are gated with the DC 1000 I/O control signals to initiate various controller/line printer operations. These are further defined in the instruction set in the following paragraphs.

DATA BUFFER. The data buffer provides temporary storage for a six-bit character being transferred from the DC 1000 to the line printer data buffer. Data is transferred from the computer to the Line Printer controller with a BTO instruction and will be presented to the printer within 10 to 20 microseconds after the transfer. This delay is required for synchronization with the printer clock. The maximum transfer rate is at 33 kHz; however, the printer will accept any lesser data transfer rates.

Even parity is generated within the controller and represents bit six in the data word.

The data word format is shown in figure 2-7. The mnemonics PIlL through PIBL represent respectively the least significant bit through the most significant bit of the data word. The parity bit is generated by the controller.

Data may be transferred to the line printer during any quiescent period during a paper motion cycle, but not when the printer is in a print cycle.

All characters of a line, including blanks, must be sent across the interface. This is a program restriction imposed on the system. Also, the first character presented to the interface will not be printed. It is required, by the printer, to initialize internal data strobing. The following characters will be printed from right to left. As an example, a 132 column printer will print the first character on the right and the 132nd character on the left.

Any character may be used in the initializing position.

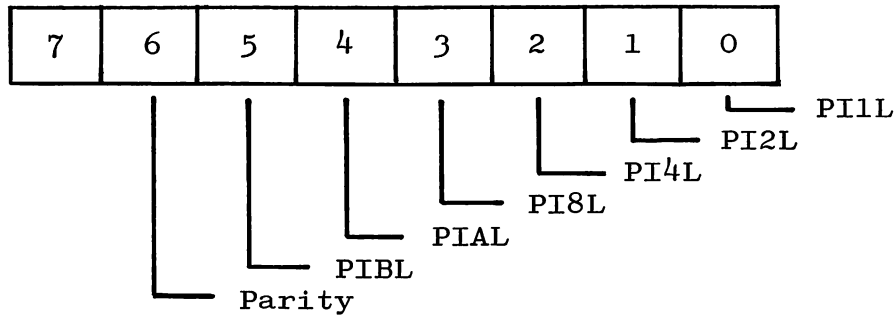


Figure 2-7. Data Word

FORMAT BUFFER. The format buffer provides temporary storage for a six-bit character: four of the bits are used to specify paper spacing; two bits are used to specify single or double line spacing.

NOTE

If the format character specifies spacing, the skip bits must be zero. If the format character specifies skipping, the space bits must be zero.

Space suppression is specified when all of the bits in the format character are off. The bottom of form is designated when bits two and three are on and bits zero and one are off. Top of form is designated when bit zero is on and all other function bits are off. The other possible variations specify a variable skip.

The printer translates these signals to initiate the required paper advance action. When the printer is spacing (PSSL or PDSL) and a hole is detected in channel 2 of the format tape, an end of page signal is sent to the controller. The end of page signal is not sent if the printer is executing a skip instruction. If data transfers have not been initiated, single or double spacing may be accomplished by transferring the format word as many times as line spacing is required. The format word is transferred to the con-

troller by means of a BTO instruction. Whenever the transfer occurs, the interrupt line to the CPU is reset.

The print and/or spacing cycle is initiated within 20 or 30 microseconds after the format word has been transferred from the CPU to the controller. Data buffer ready should be sensed before outputting format.

The format word is shown in figure 2-8. FC1L through FC8L comprise the four elements required for format control. FC1L is the least significant bit. PSSL and PDSL specify single and double spacing.

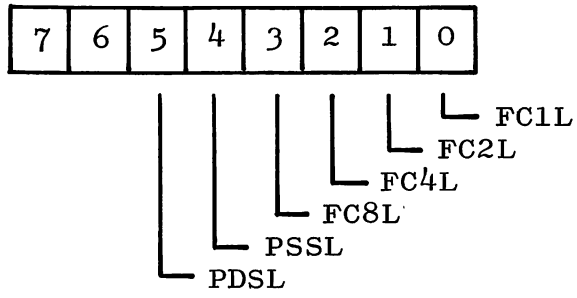


Figure 2-8. Format Word

SENSE LOGIC. The sense logic allows the status conditions of the line printer and the controller to be interrogated by the CPU (table 2-8).

Table 2-8
Line Printer and Controller Conditions

Condition	Description
Parity error	A parity error exists.
Sync error	A printer counter sync error exists.
End of page	An end of page has been reached.
Paper motion	Paper motion due to formatting or line skipping is completed.

Table 2-8 (cont)
Line Printer and Controller Conditions

Condition	Description
Print cycle	The print cycle initiated by the print command is completed.
Printer ready	The following conditions exist: a. Power on. b. Paper loaded. c. No paper motion alarm. d. Start button has been activated. e. Printer is in remote status.
Data buffers ready response	Controller ready for data.

NOTE

If a data buffer not ready exceeds 30 microseconds, a failure mode exists.

Certain sensed conditions require additional controller actions as defined below:

a. When a parity error is sensed, the controller issues a command to reset the error. The reset requires 10 to 20 microseconds.

b. When an end of page is sensed, the controller issues a command to reset. The reset requires 10 to 20 microseconds.

The above conditions should be inspected after a line transfer and its subsequent print cycle followed by the interrupt.

INTERRUPT LOGIC. One interrupt line is provided. The interrupt occurs when:

a. Space suppression is specified and the print cycle is completed.

b. The paper motion specified by the format word or independent single or double line spacing is completed.

The interrupt is reset when:

a. Any sense (SEN) is issued.
b. Format data is transferred to the controller (BTO).

The interrupt is enabled or disabled by separate FUN commands.

CARD PUNCH CONTROLLER.

The Card Punch Controller provides the necessary logical circuits to interface the DC 1000 with a 100 cpm Card Punch. The computer communicates with the card punch via the standard eight-bit input/output interface. A block diagram of the card punch controller is shown in figure 2-9. The DC 1000, by issuing control and data transfer commands to the controller, can cause any combination of holes to be punched in a standard tab card.

There are no operator controls on the card punch controller. All operations are controlled by program execution of instructions from the computer.

FUNCTIONAL DESCRIPTION. When power is first applied to the controller, a system reset from the console or an initialize-controller function (FUN) command is required. This sets the controller to a non-interruptable mode with the interrupt reset and the main card stacker selected. The card punch should be idling ready to receive punch data.

The computer may issue a main-stacker-select or an auxiliary-stacker-select FUN command before processing a card.

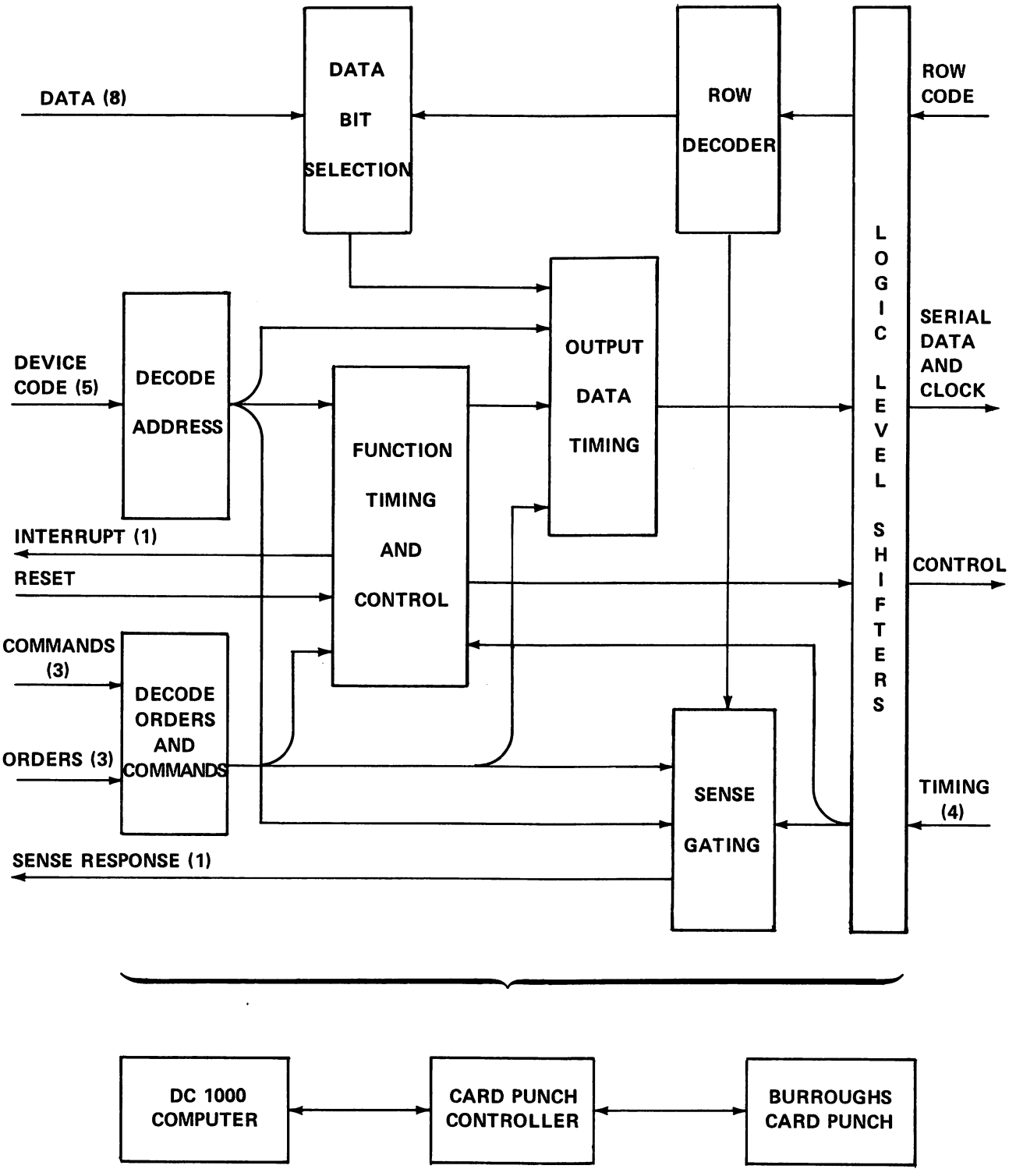


Figure 2-9. Card Punch Controller Block Diagram

DATA FORMAT. Cards are punched one row at a time, requiring 80 serial bits of data from the controller. The controller provides the data by selecting a particular bit from each of the 80 bytes of data transferred from the computer. The bit selected from each byte is determined by the row being punched. Figure 2-10 shows the data format for bytes transferred from the computer.

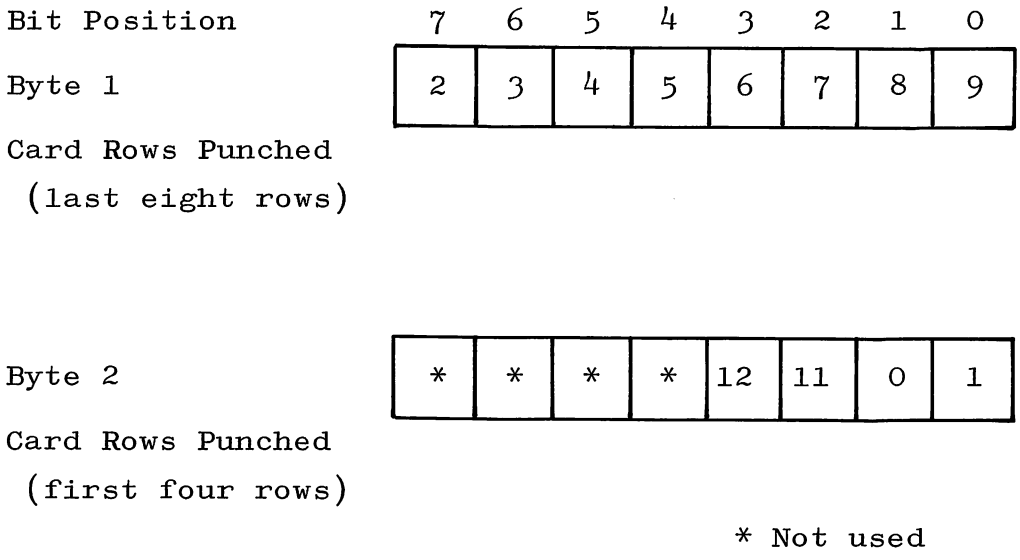


Figure 2-10. Card Punch Controller Data Word Format

The computer stores two bytes of data for each column of the card to be punched. The first byte includes eight data bits for rows 5 to 12. The second byte contains four data bits for rows 1 to 4 of the card. The 80 even-numbered bytes must be transferred to the controller four times to punch four rows. The 80 odd numbered bytes must be transferred eight times to complete the punching of the card. Figure 2-11 shows the correlation between the byte transfers and card positions.

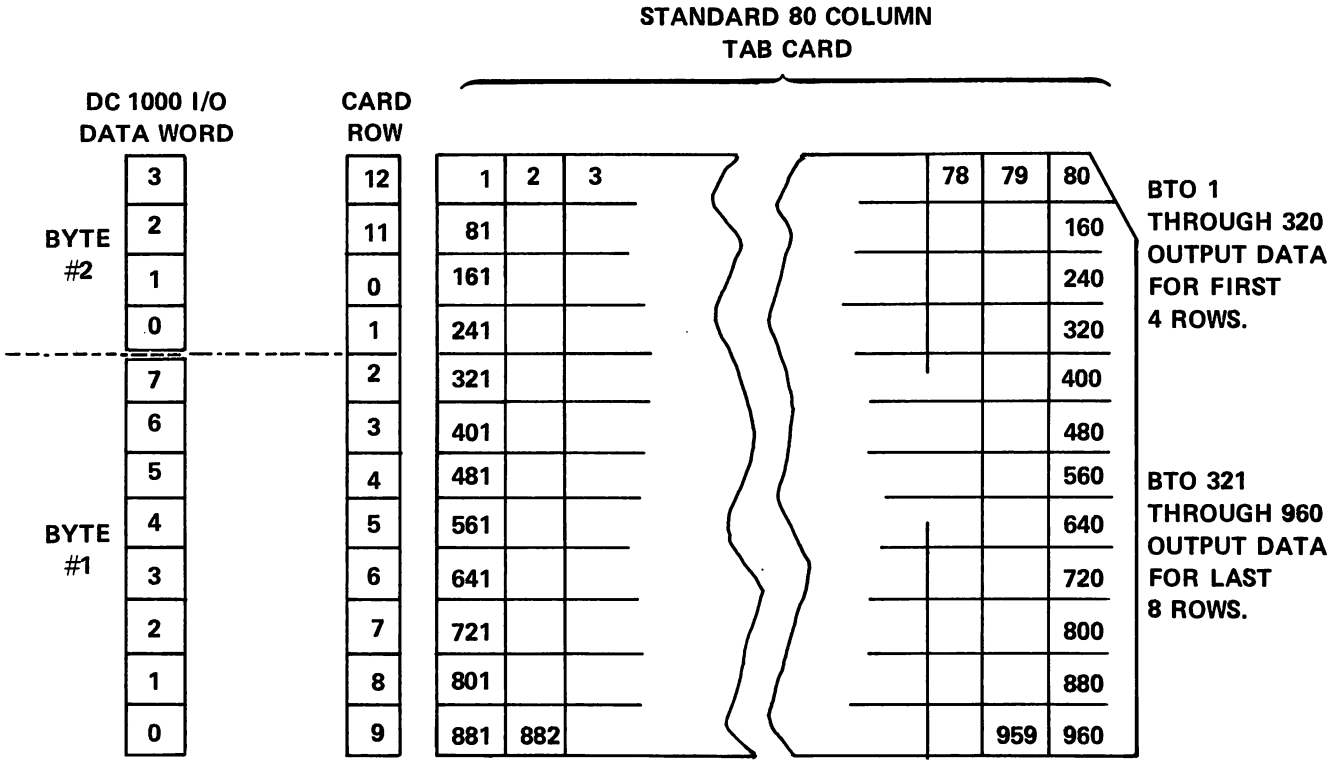


Figure 2-11. Card Punch Controller,
BTO Data Output Sequence

DATA TRANSFER. Punching of a card is initiated by the computer. The computer first enables the controller to operate in either the interrupt or the sense-loop input/output (I/O) mode. The computer executes the Interrupt-Enable FUN command when changing to the interrupt I/O mode. The computer executes an Interrupt-Disable FUN command when changing back to the Sense-Loop I/O mode.

The processing of a card is basically the same for both I/O modes. The computer first executes an Initiate-Cycle FUN command to the controller. The controller transmits a signal to the punch indicating that a card is to be punched. When the punch is ready for data, the controller receives signals indicating that the punch information is needed. The controller sends an interrupt to the computer if in the interrupt mode, or waits for the computer to execute a Data-Buffer-Ready Sense (SEN) command if in the sense-loop mode.

If a SEN command is executed, an affirmative response is sent to the computer, signifying a data-ready condition.

When a data-ready condition is established, the computer transfers (BTO) a burst of 80 bytes to the controller within a period of 5.2 milliseconds. The first BTO resets the controller interrupt. The eightieth BTO resets the data-buffer-ready signal in the punch.

The computer first transfers even-numbered data bytes of the 160 bytes required to punch a complete card. Each of the first four card rows are punched by the even-numbered bytes. The last eight rows are punched by the odd-numbered bytes. Eighty BTO commands are required to punch a row; 960 BTO commands are required to punch a full card.

Card punch errors may be sensed by the computer after the processing of each card. If a punch error occurs, the punch generates a punch-error signal. The controller then produces an affirmative sense response to the computer on receipt of a Punch-Error SEN command.

The computer does not terminate the punching of an erroneous card, but instead, completes the punching of the entire card. The bad cards are automatically rejected to the error stacker by the punch. the punch-error signal is reset when the computer initiates another punch cycle.

CARD READER CONTROLLER.

The Card Reader Controller provides the necessary data buffering and control signals to interface the DC 1000 with a Burroughs 200 cpm Card Reader. A block diagram of the Card Reader Controller is shown in figure 2-12. The DC 1000 communicates with the card reader via the standard eight-bit I/O interface. The DC 1000, by issuing control and data transfer commands to the controller, can read twelve bits of column data, column by column, from a standard 80 column card.

There are no operator controls on the card reader controller. All operations are controlled by program execution of instructions within the DC 1000.

FUNCTIONAL DESCRIPTION. The following paragraphs describe the functional operation of the card reader with reference to the block diagram in figure 2-12.

INITIAL OPERATION. When power is first applied to the controller, initial conditions must be established by a system reset from the console. This resets the controller interrupt line and causes the controller to transmit a general reset signal to the card reader.

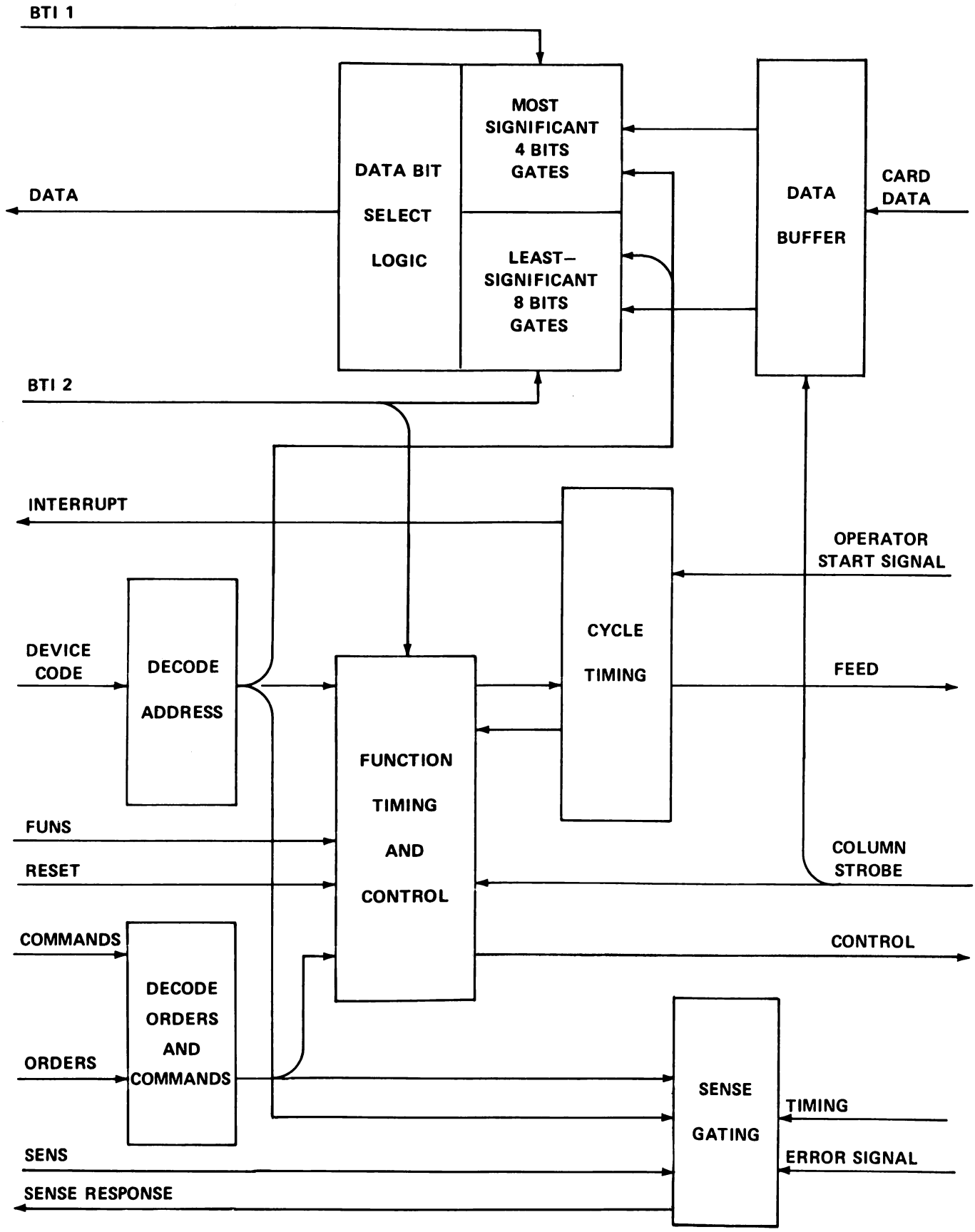


Figure 2-12. Card Reader Controller Block Diagram

DATA FORMAT. Cards are read a column at a time, providing twelve bits of data to the controller. The bits are transferred to the computer (BTI) in two bytes as shown in figure 2-13.

DATA TRANSFER. The computer initiates the reading of a card. The computer enables the controller to operate in either the interrupt or sense-loop input/output (I/O) mode. The computer executes an Interrupt-Enable Function (FUN) command when changing to an interrupt I/O mode. The computer executes an Interrupt-Disable (FUN) command when changing back to the sense-loop I/O mode.

Data Bits	7	6	5	4	3	2	1	0
Card Rows Read Byte 1	*	*	*	*	12	11	0	1
Card Rows Read Byte 2	2	3	4	5	6	7	8	9

* Not used

Figure 2-13. Card Reader Controller Data Word Format

The computer issues an initiate-cycle command after sensing that the card reader is ready to accept a command. This causes the controller to initiate a card-read cycle by sending a feed signal to the card reader.

Shortly after the initiate cycle command, the card reader reads the first column (12 bits) of data from the card. The information is transferred to the controller along with a data strobe signal. The data is stored in data-buffer flip-flops and a data-ready indicator is set. If the controller is in the interrupt mode, the data-ready condition generates an interrupt to the computer. If the controller is in the sense-loop mode, the computer must interrogate the controller with a data-ready sense (SEN) command. In either case, the

computer executes two byte transfer (BTI) commands to transfer a column of data from the controller to the computer. The first BTI command transfers the most significant four bits of the card column to the computer. The second BTI command transfers the least significant eight bits of a card column. The computer must complete the transfer of a card column within 1.7 milliseconds. When the second BTI command is executed, the data-ready indicator and the interrupt are reset.

The card reader resets a busy indicator after all 80 columns of the card have been read. The computer senses this condition with a Card-Reader-Busy SEN command. The next card read cycle can then be initiated.

SINGLE LINE CONTROLLER.

The single line controller interfaces with the DC 1000 Computer to control data input and output through the Bell System Data sets 201A and 201B or equivalent synchronous data sets. The controller may be used for half or full duplex data transfers under control of the DC 1000 software and the data set supplied clock of 2000, 2400 or 4800 bits per second.

The basic function of the single line controller is to match the serial data set interface to the parallel I/O bus of the DC 1000 Computer. Message format and line control procedures are completely under control of the software so that any eight-level character code and error detection schemes may be used.

Data transfer across the modem interface is synchronous with respect to the data set clocks. There are no start or stop bits framing the individual eight-bit characters. Normally, several predetermined sync characters precede the data characters of an input or output message so that the receiving computer can get into character phase with the message data. The controller hardware itself is transparent to all message and synchronizing characters although it does determine receive character parity.

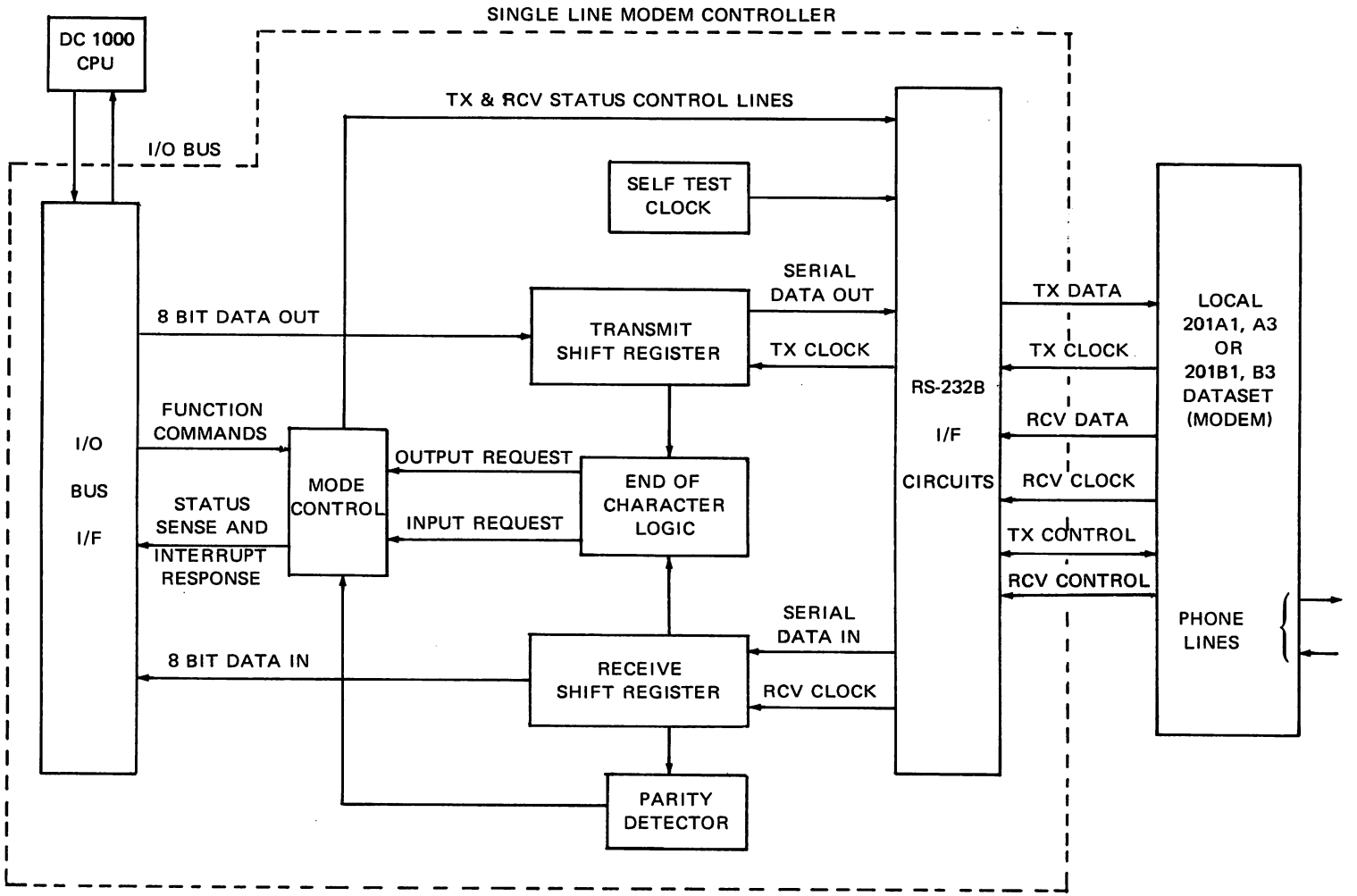


Figure 2-14. Single Line Controller Block Diagram

TRANSMIT. When the transmit section of the single line controller (figure 2-14) requests a character for transmission, the CPU outputs eight bits in parallel to the transmit shift register. The transmit data clock causes the register to shift this character one bit at a time through the modem interface circuits to the data set for bit-serial transmission over the switched voice frequency telephone circuit or private line.

Controller logic detects the eighth shift pulse and interrupts the CPU with a request for the next character to be loaded into the shift register for transmission.

RECEIVE. 'The receive section of the controller operates in the bit mode to achieve character phase with the incoming data and in the word mode to transfer message characters to the CPU. Serial data from the data set is shifted into the receive shift register under control of the receive data clock.

In the bit mode, controller logic interrupts the CPU after every shift pulse. The CPU inputs an eight bit word in parallel from the shift register. These words are examined by the CPU software at each bit time until the sync character pattern is recognized. The CPU then commands the controller to the word mode.

In the word mode, controller logic detects the receipt of a complete message character in the shift register (8 shift pulses) and interrupts the CPU with a request that the received character be inputted to the CPU.

FULL DUPLEX, 4-WIRE, PRIVATE LINE OPERATION. Full duplex operation, with simultaneous message flow in both directions between a local and remote data terminal, requires a four-"wire" connection between the data sets as shown in figure 2-15. "Wire" is used in the sense of a signal path or a signal return path and could actually consist partially of a microwave link.

The connection between terminals in full duplex operation is almost always a private, unswitched line such that automatic answering or calling units are not used.

Refer to table 2-9 for a definition of mnemonics.

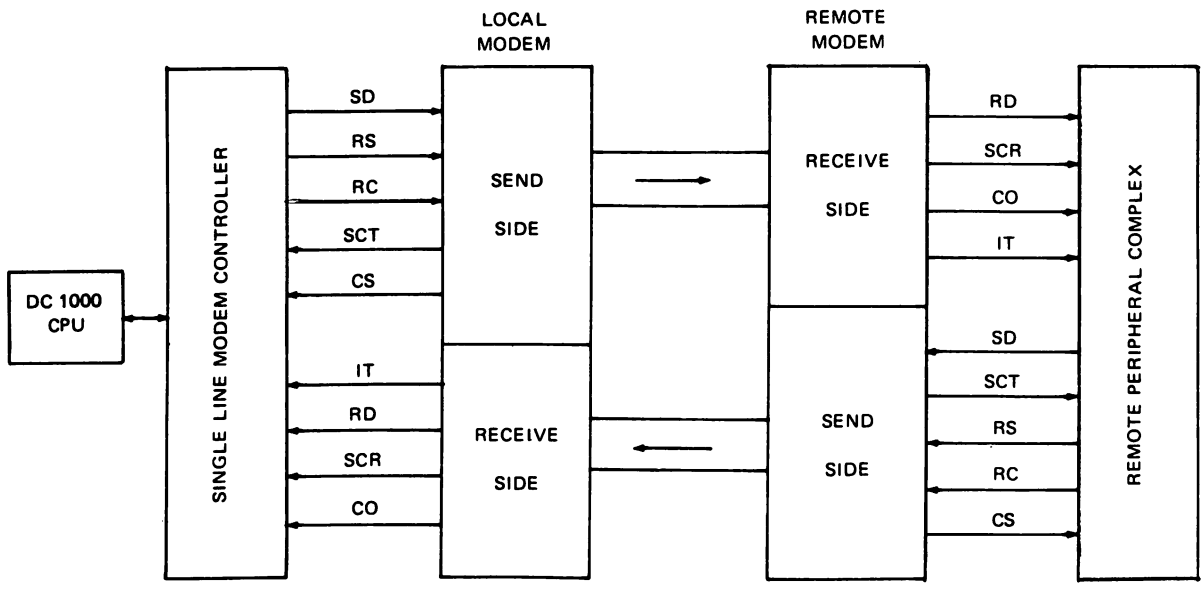


Figure 2-15. Full Duplex, 4-Wire, Private Line

Table 2-9

Mnemonic Definitions

Mnemonic	Signal	Description
CO	Carrier On	The data set receive section is receiving a carrier signal from the remote dataset.
CS	Clear to Send	The local and remote data sets are in sync and ready for transmission.
IT	Interlock	The data set is connected to the phone lines and is ready to transmit or receive data.
RC	Remote Control	Indicates that the data set is to make connection with the phone lines or to maintain the connection if it already exists.
RD	Received Data	Data received from the remote terminal.
RGI	Ring Indicator	Used to inform the controller that a call has been received.
RS	Request to Send	Controller indicates to the local data set that data is ready for transmission to the remote data set.
SCR	Serial Clock Receiver	Synchronizes receive data bits between the local data set and the controller.

Table 2-9 (cont)
Mnemonic Definitions

Mnemonic	Signal	Description
SCT	Serial Clock Transmitter	Synchronizes transmit data bits between the local data set and the controller.
SD	Send Data	Transmit data to the remote terminal.
SG	Signal Ground	Ground reference for all data, clock, and control lines.

Full Duplex Line Turn On.
Assume that the local data terminal consisting of the DC 1000 single line controller and data set is powered up but in an Idle mode, neither transmitting nor receiving messages.

When the terminal is to be placed into operation, the DC 1000 software turns on the Data Terminal Ready line to the data set. When the data set returns the Interlock On signal to the controller, it indicates that the data set is in the data mode and is ready to initiate the transmission or reception of a message.

If the local terminal wishes to send a message, the software raises the Request to Send line. The send side of the local data set then attempts to synchronize the receive section of the remote data set to the local transmit clock. After a delay of about 150 milliseconds, assuming that the remote data set was in the data mode, a Clear to Send signal is returned to the controller from the local data set.

The Clear to Send signal permits the transmit clock to shift the write shift register of the controller and send the serial message data on the Send Data lead.

Typically, in full duplex operation, when the remote data terminal receives the local transmit carrier to synchronize on, it attempts to establish a message linkback to the receive side of the local data set in the same fashion.

When the local data set detects the transmit carrier from the remote data set, it raises the Carrier On lead to the controller. The Carrier On signal enables the receive clock to shift the receive shift register and input serial data from the Receive Data line into the controller.

Once the send and receive start up sequence has been performed, full duplex message transmission is continuously possible without manipulation of the data set control lines. The start and finish of a message are normally identified by certain software generated control characters so that the Request to Send line can be left on all of the time. In periods between messages, the send side of the data set is held to a steady mark condition. In this way, the delay associated with synchronizing the receiving data set between messages is avoided.

Full Duplex Line Turn Off.

If the local data terminal wishes to turn off its transmit section at any time, the software orders the Request to Send Line to the off state. The data set drops the Clear to Send Line to the controller within one millisecond but continues to transmit to the send section of the data set for another two milliseconds to clear data out of the send section of the data set before turning off.

Since the Clear to Send signal enables the write clock of the controller, the software must allow sufficient time for the last character of a message to clear the controller before turning off the Request to Send. The safest way is to include one or more fill characters (such as all 1's) after each message to protect the end of message from turn off timing.

HALF DUPLEX, TWO WIRE, SWITCHED LINE OPERATION. Half duplex operation, with only one terminal operating at a time, generally uses a two wire connection into a switched network (figure 2-16). The operator must manually dial on the telephone handset to establish connection with the remote terminal. If an Automatic Calling Unit (ACU) is used with the data set, the connection with the remote terminal can be made under software control.

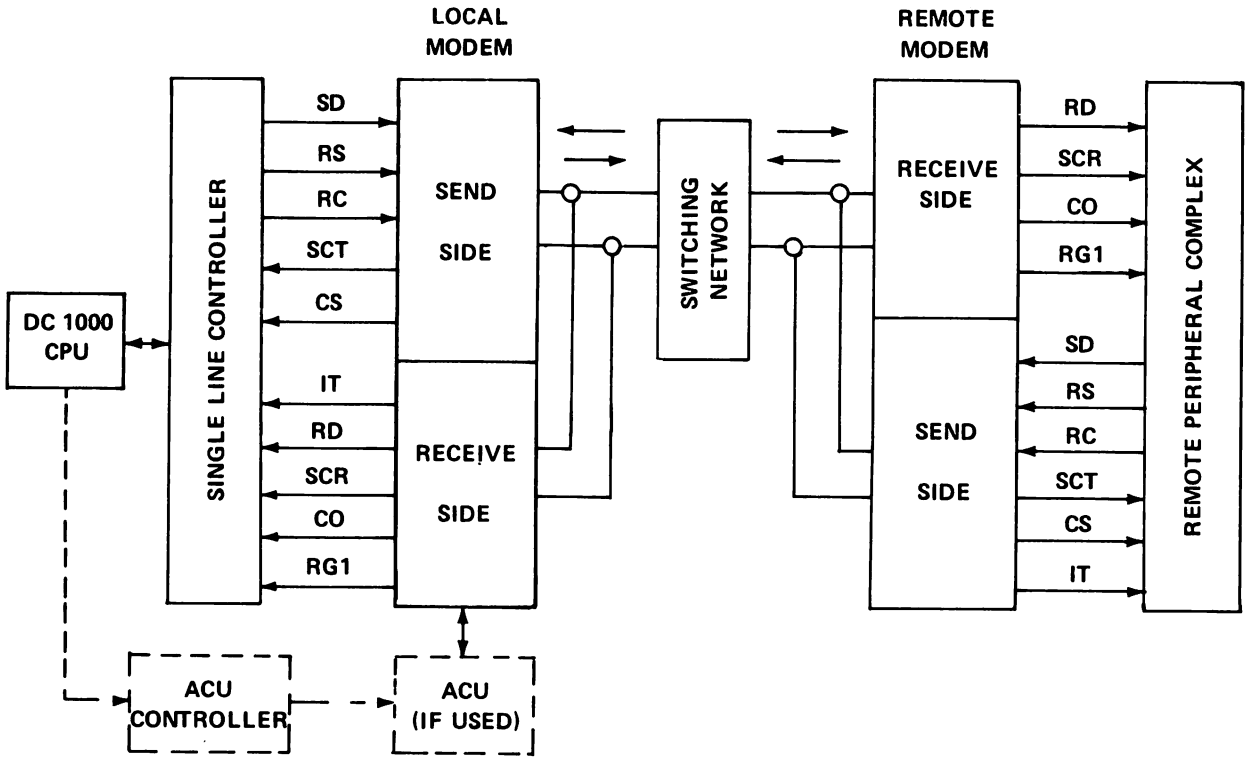


Figure 2-16. Half Duplex, 2-Wire, Switched Line

In a two wire connection, the send side of the data set is strapped to the receive side. Therefore, the receive side of the modem controller receives what the send side is transmitting at all times. The software of the DC 1000 must ignore this wrap around input.

Refer to table 2-9 for a definition of the mnemonics.

Half Duplex Line Turn On.

Assume that the DC 1000 single line controller, data set and ACU (if used) are powered up but in the idle state.

If the data set answering equipment detects an incoming call, it raises the Ring Indicator line to the controller. If the DC 1000 software has decided to accept incoming calls, it will have previously enabled the Ring Indicator interrupt line so that the signal from the data set will interrupt the program.

If the program has not enabled the controller interrupt system, the incoming call is ignored.

The software answers the Ring Indicator by turning the Data Terminal Ready line on to the data set. The data set responds by turning on its interlock line when it is in the data mode.

When the data set detects the presence of the carrier from the remote terminal, it turns on the Carrier On line which enables the receive shift register in the controller. Data is then shifted into the controller. The transmit side of the controller is off and not shifting because Request to Send is off.

If the local terminal wishes to transmit from an idle state, it turns on the Data Terminal Ready line and receives the Interlock On signal in return. When Interlock is received, the program turns on the Request to Send and waits for the Clear to Send response from the data set (150 millisecond delay).

As soon as the data set transmits carrier to the remote terminal, the carrier detector in the receive side of the local data set will cause

the Carrier On line to the controller to go on. This enables the receive shift register so that the data transmitted by the controller to the data set will be reflected back to the receive input of the controller time delayed by the circuit delays in the data set (one or two milliseconds). The program ignores this input when transmitting in half duplex.

Half Duplex Line Turn Off.

If the data terminal is finished with its message transmission and wishes to switch to a receive enabled idle condition (Data Terminal Ready left on), it delays turning off the Request to Send line to allow the send side of the data set to clear itself of message data. The most convenient way to provide this delay is the transmission of one or more non-sync pattern, fill characters at the end of the message.

When the Request to Send is turned off, Clear to Send drops in about one millisecond but the reflected input to the receive side of the controller continues for approximately ten milliseconds so that the Carrier On and Receive Clock will stay on. If the last character was a fill character, the fact that it was not a sync character would protect against the receive side interpreting the reflected input as the initial sync characters from the remote terminal.

Optionally, the controller software may be written to monitor the Carrier On interrupt for a change in state of the Carrier On lead to the off condition before expecting the start of the line turn on sequence initiated by the remote terminal.

If for some reason the local terminal wishes to immediately stop all transmission and reception, the software can turn off the Data Terminal Ready and drop the data set out of the data mode. Any message, of course, would be lost.

INTERRUPT LINES.

The controller notifies the CPU of a need for service via six interrupt lines listed in table 2-10.

Table 2-10
Interrupt Lines

Interrupt Term	Description
INTRA-	Write buffer ready.
INTRB-	Read buffer ready.
INTRC-	Change in state (Off to on or on to off) of data set control line Clear to Send.
INTRD-	Change in state (Off to on or on to off) of data set line Carrier On.
INTRE-	Change in state from off to on of data set control line Ring Indicator.
INTRF-	Change in state (Off to on or on to off) of data set control line Interlock.

These six terms are wired to socket location L2 where an interrupt select block connects them individually or in wired OR combinations to I/O bus interrupt lines E100- through E105- and E107-. Jumper block configuration can be established by the user after consideration of the program timing and service priorities of other peripheral devices connected to the DC 1000 CPU.

Normally the write and read buffer ready interrupts would be assigned to high priority interrupts since they must be serviced within one clock time of the modem clock if message integrity is to be maintained.

PAPER TAPE READER CONTROLLER.

To be specified.

PAPER TAPE PUNCH CONTROLLER.

To be specified.

SECTION 3
SOFTWARE FUNCTIONAL DESCRIPTION

GENERAL.

This section contains descriptions of the software that is available for the DC 1000. Detailed descriptions and proper usage of the software can be found in the DC 1000 Series System Software manual. The software covered in this section includes:

a. Data, instruction, and input/output word formats.
b. Addressing modes.
c. DC 1000 Assembler.
d. B 3500 Assembler.
e. Executive routines.
f. Peripheral Service routines.
g. Mathematical routines.

The data communications software for the DC 1000 are covered in a separate section of this manual.

WORD FORMATS.

There are three types of words used in the DC 1000:

a. Data words.
b. Instructions.
c. Input/output words.

In the following word format descriptions, bit positions are numbered from right to left with the highest-numbered bit being the most significant and bit zero being the least significant.

DATA WORDS.

Data may occupy as few as eight bits or as many as 32 bits in memory and in the accumulator. Precision of the data may be varied in eight-bit bytes as shown in figure 3-1. The most significant bit is off for positive numbers and is set for negative numbers. All negative quantities are represented in the 2's complement form.

The 2's complement of a number may be found in either of two ways:

a. Take the 1's complement of the number (complement each bit) and add one to the result.

Example:

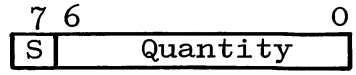
Number	00001001	(+9)
1's complement	11110110	
	+1	
2's complement	11110111	(-9)

b. For an n-bit number (including the sign bit), subtract the number from $2^{(n+1)}$.

Example:

$2^{(n+1)}$	100000000
(+9)	00001001
-9	11110111

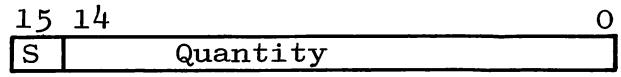
Eight-bit format



Number range: Fractional $= -1 \leq N < +1$

Integer $= -2^{7} \leq N < +2^{7}$

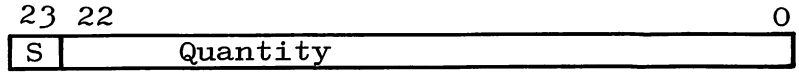
16-bit format



Number range: Fractional $= -1 \leq N < +1$

Integer $= -2^{15} \leq N < +2^{15}$

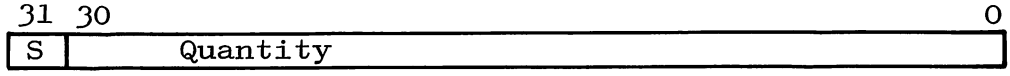
24-bit format



Number range: Fractional $= -1 \leq N < +1$

Integer $= -2^{23} \leq N < +2^{15}$

32-bit format



Number range: Fractional $= -1 \leq N < +1$

Integer $= -2^{31} \leq N < +2^{31}$

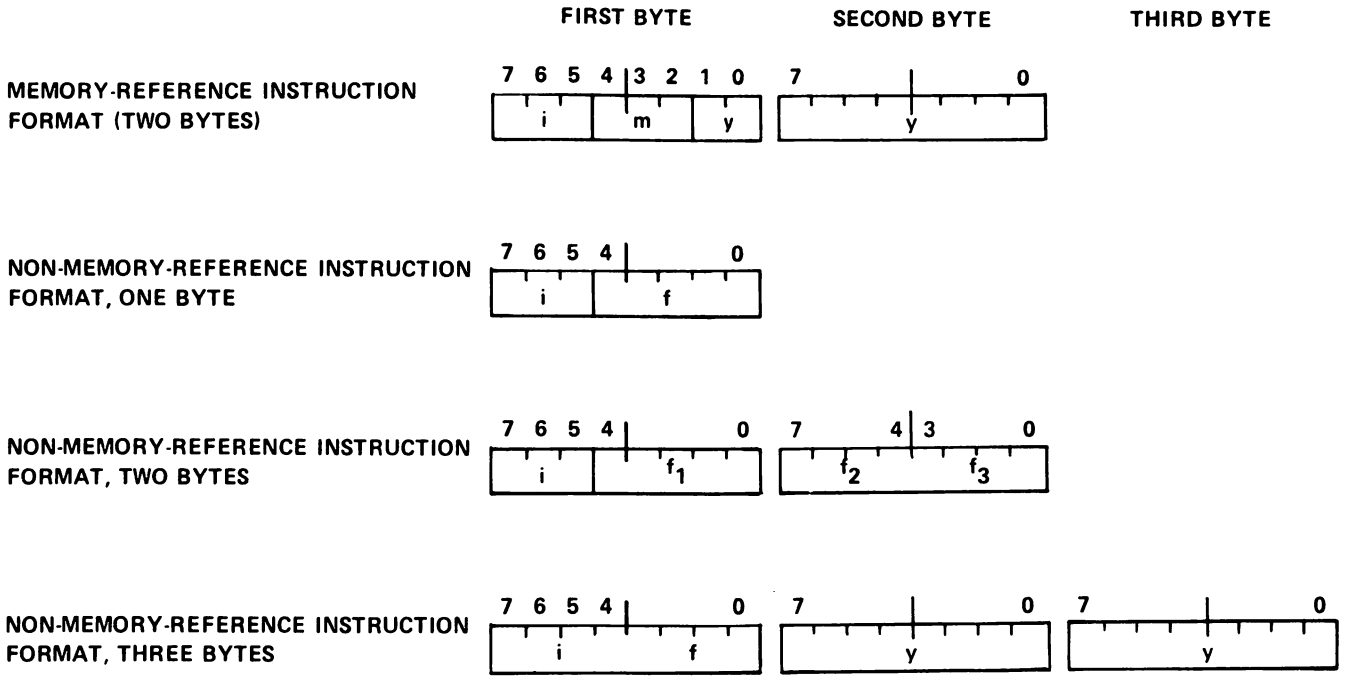
Bit 0 = least significant bit

S = sign bit

Figure 3-1. Data Word Formats

INSTRUCTION WORDS.

Instructions occupy as few as eight or many as 24 bits in memory. Instructions are either memory-reference or non-memory-reference types (figure 3-2).



0 = LEAST SIGNIFICANT BIT
i = BINARY INSTRUCTION CODE
m = BINARY ADDRESSING MODE (SEE TABLE 3-1)
y = OPERAND ADDRESS (MAY BE MODIFIED TO OBTAIN EFFECTIVE ADDRESS)
f = FUNCTION TO BE PERFORMED
f_1 = FUNCTION CLASS
f_2 = FUNCTION CLASS MODIFIER
f_3 = FUNCTION CLASS MODIFIER

Figure 3-2. DC 1000 Instruction Formats

INPUT/OUTPUT WORDS.

Input/output words communicate information on the eight-bit I/O channel with the formats shown in figure 3-3.

Basic communication with the DC 1000 utilizes an eight-bit data word, an eight-bit address word, and interrupt signals.

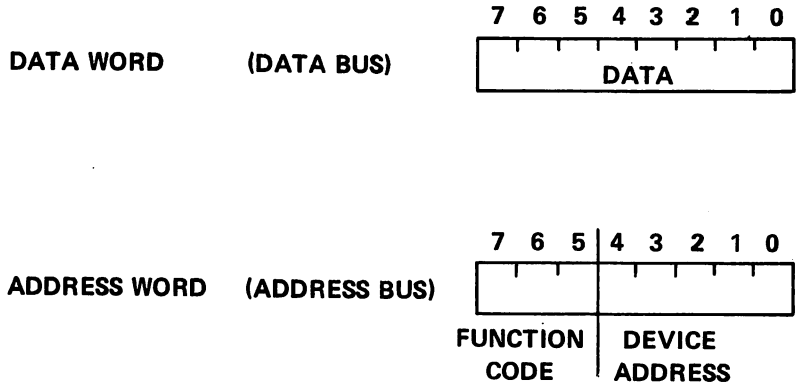


Figure 3-3. DC 1000 Input/Output Word Formats

ADDRESSING MODES.

Addressing the contents of memory is a function of the current operand and accumulator word length. Word length (precision) is program selectable. The memory is addressed at the byte level. If the current data precision is one byte, the operand address is the only address for that eight bits of data. If the current data precision is two, three or four bytes, the operand address refers to the memory location of the most significant byte. The accessing of multiple bytes of memory occurs automatically in the correct order for execution.

Address modification is defined by the m field of the instruction word. These three bits specify direct, indirect or indexed addressing. The bits are decoded into eight address conditions defined in table 3-1.

DIRECT ADDRESSING.

There are two forms of direct addressing used in the DC 1000:

a. Direct to lower four sectors; enables direct addressing to 4096 bytes of memory. For the standard computer this is total direct addressing.

b. Direct to current sector; gives access to 1024 bytes in the sector in which the current instruction is being executed (determined by the most significant bits of the program counter).

INDIRECT ADDRESSING.

Indirect addressing uses the address to obtain another operand address contained as a 16-bit word in memory locations y and y+1. A 16-bit address is always assumed when indirect addressing is specified. The most significant bit specifies further indirect addressing when it is set. Each level of indirect addressing requires an additional two machine cycles of execution time.

The DC 1000 uses two methods of indirect addressing:

a. Indirect addressing to zero. Regardless of current sector location, the addressing mechanism returns to the lowest 1024-byte sector of memory to find the effective address. This allows indirect addresses for commonly used subroutines to be accessible from all sectors.

b. Indirect addressing in current sector. Address y designates the least significant part of the address while the program counter specifies the most significant part. This enables the use of the 16-bit indirect addresses within the current operating sector.

INDEXING.

The 10-bit operand address y may be modified by the addition of the contents of the Index register. This yields a 16-bit address that allows addressing of any location in the maximum 32,768-word memory.

A sector is 1024 bytes. The basic DC 1000 has four sectors, with a maximum of 32 sectors possible through optional memory expansion.

Table 3-1
Address Bit Decoding

Bit 12	Bit 11	Bit 10	Address Condition
0	0	0	Direct to lower four sectors.
0	0	1	Direct to lower four sectors.
0	1	0	Direct to lower four sectors.
0	1	1	Direct to lower four sectors.
1	0	0	Direct to current sector.
1	0	1	Indirect through current sector.
1	1	0	Indexed.
1	1	1	Indirect through zero sector.

DC 1000 ASSEMBLER.

The assembler for the DC 1000 is a three-pass symbolic assembly program; it will assemble source statements written in symbolic machine language, and will produce an assembly listing and an object program in loadable form for future execution on the DC 1000. Source statements input may be via either the teletype paper tape reader or the teletype keyboard. If the keyboard is specified, the input statements are punched into paper tape for input to pass II and III.

The DC 1000 Assembly program will identify errors in the source input statements with appropriate flags which will appear on the

assembly listing. When applicable, NOP's will be assembled into the output object code whenever errors are detected in order to maintain continuity and as a convenience for patching without reassembly. The assembler will reference multiple defined symbols to the location of the first occurrence of the symbol.

The location counter and the machine language translations of the symbolic source statements will appear in hexadecimal notation on the assembly listing. The instruction mnemonics together with their hexadecimal representations are shown in appendix B.

FUNCTIONAL DESCRIPTION.

PASS I. The following functions are executed during pass I:

a. Generate symbol table.

b. Assign absolute values to all symbols.

c. Check for illegal operation codes.

d. Check for undefined and multiple defined labels.

e. Make literal assignments.

f. Make block storage assignments.

g. Signal end of pass.

h. Halt with the program counter set to the beginning of the next pass.

When the RUN switch is activated, the DC 1000 Assembly program checks the console SENSE switches to determine which pass will be executed.

The DC 1000 Assembly program recognizes two modes of input to pass I:

Interactive Keyboard Mode.

The DC 1000 Assembly program provides full pass I processing with

automatic error diagnostics on a statement by statement basis for statements entered through the teletype keyboard. Only error free data is retained. As data is received from the keyboard, it is arranged internally to agree with the standard symbolic coding format (figure 3-4). Upon receiving a space character from the keyboard, the assembler will advance to the first position of the next field of input. The first non-space character from the keyboard marks the beginning of that input field. A carriage return followed by a line feed character terminates the input statement. The assembler examines each statement for erroneous content. If the statement is free of error, its contents through position 58 of the source input buffer will be punched onto paper tape.

Symbolic Paper Tape Mode.

Symbolic source statements on paper tape are received through the teletype paper tape reader. In the Symbolic Paper Tape mode of input, no diagnostics are provided to the user during pass I. The punched source statements produced in the Interactive Keyboard mode will be accepted as input to pass I in the Symbolic Paper Tape mode of input.

Mixed Modes of Input to Pass I.

Through the proper use of the MOR pseudo operation (see appendix B) a user may change, at will, the input mode during pass I from paper tape to keyboard and back. The flexibility of the assembly program input/output in conjunction with the MOR pseudo operation will allow this user option.

PASS II. The following functions are executed during pass II:

a. Formulates the binary code from the source data.

b. Generates indirect address references when the instruction memory references conflict with the hardware core sectorization.

Burroughs DC 1000 CODING FORM

PROGRAM ID		PROGRAMMER		DATE		PAGE		OF	

LABEL		OPERATION		VARIABLE FIELD	REMARKS		SEQ. NO.
01	06	08	14	16 29	30 58	59 72	73 80

Printed in U.S. America

Form 1045689

Figure 3-4. DC 1000 Symbolic Coding Form

C. Punches the binary object code into paper tape for subsequent loading into the DC 1000 Processor (figure 3-5).

d. Upon completion, the processor halts with the program counter set to the beginning of the next pass.

When the RUN switch is activated, the DC 1000 Assembly program will determine from the console SENSE switches which pass will be executed.

PASS III. The following functions are executed during pass III:

a. Print assembly output and associated source statements on the teletypewriter (figure 3-6).

b. Signal the end of the pass.

c. Halt with the program counter set to the starting location of the next pass.

When the RUN switch is activated, the assembly program executes the next selected pass.

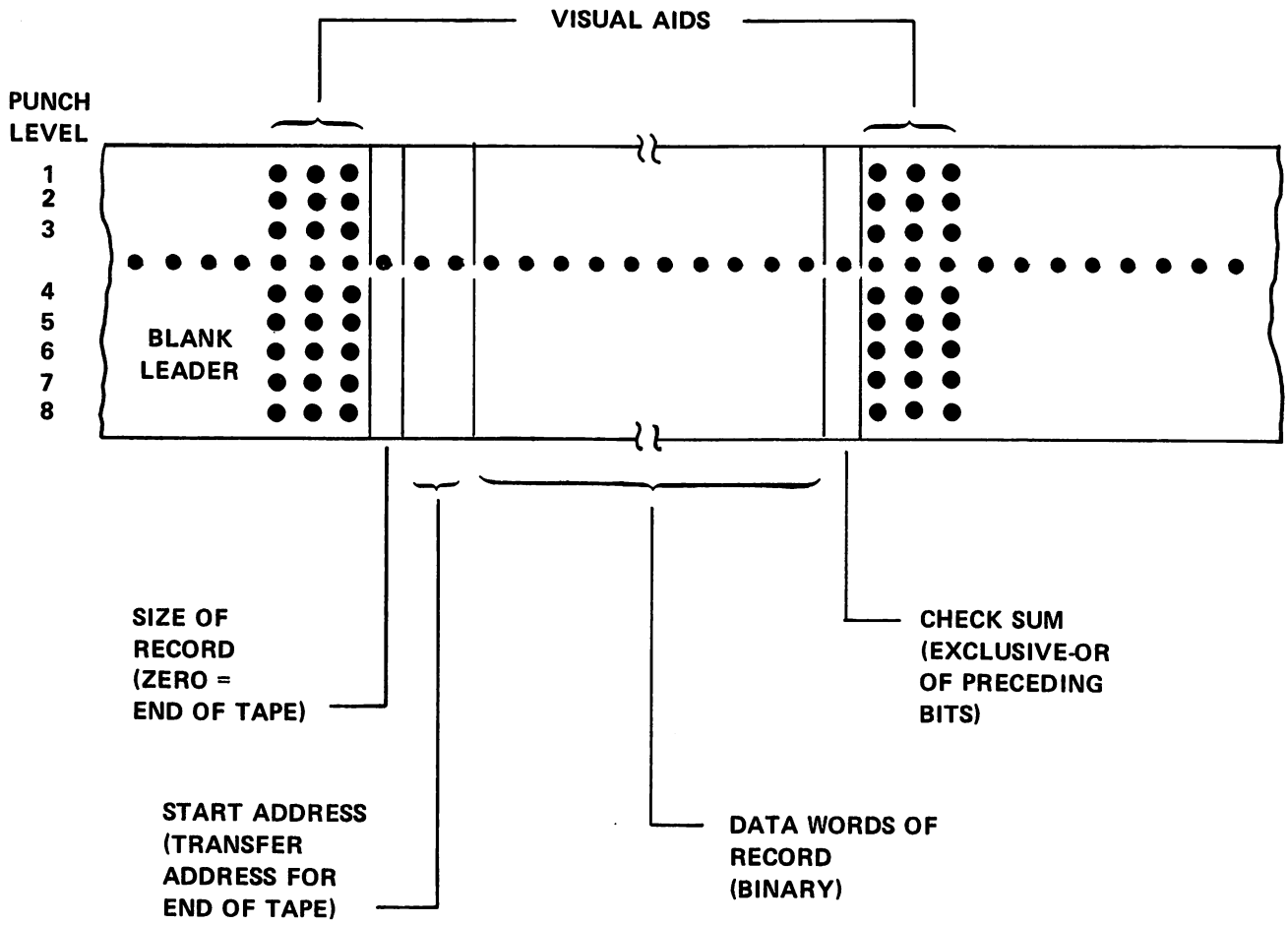


Figure 3-5. Object Tape Format

```
PAGE  1
ADDRESS  CODE      ERROR  LABEL  OP     VARIABLE    COMMENTS                  SEQUENCE
0100                      ORG    $0100                                    00000100
0100                      ENTRE  EQU    *                                 00000200
0100     10               SP1                                             00000300
0101     4502             LOD1   =$8D    CARRIAGE RETURN                  00000400
0103     3221             SEN    $21     OUTPUT BUFFER READY              00000500
0105     F103             BRU    *-2     NO                               00000600
0107     3001             BTO    $01     YES                              00000700
0109     4503             LOD1   =$8A    LINE FEED                        00000800
010B     3221             SEN    $21     OUTPUT BUFFER READY              00000900
010D     F10B             BRU    *-2     NO                               00001000
010F     3001             BTO    $01     YES                              00001100
0111     5002             LOD    $02     RESET POINTER AT $0502           00001200
0113     7122             STO    ADD +1                                   00001300
0115                      INTO   EQU    *                                 00001400
0115     3241             SEN    $41     INPUT BUFFER READY               00001500
0117     F115             BRU    *-2     NO                               00001600
0119     3101             BTI    $01     YES                              00001700
```

Figure 3-6. Sample Listing

INSTRUCTIONS.

The DC 1000 machine language instructions are divided into five basic types listed in appendix B.

a. One-byte non-memory reference (table B-1).
b. Two-byte non-memory reference (table B-5).
c. Two-byte memory reference (table B-7).
d. Three-byte non-memory reference (table B-6).
e. Two-byte input/output (table B-8).

LITERALS.

The DC 1000 Assembly program allows a user up to four literal pools in one source program. Each pool has a capacity of 40 bytes with a maximum of four symbolic literal definitions per pool; thus a single source program may contain as many as 160 bytes of literal definitions with 16 symbolic literal definitions.

DATA DEFINITION.

Different types of data may appear in a single data definition statement. In the case where an alpha constant is contained in a data definition statement, the precision is ignored and is limited in extent only by the size of source input buffer which does not accept data beyond column 58 of the operand field.

The Assembly program will generate zeros to the precision specified for each successive comma contained in the operand field of the data definition statement and for the blank end-of-line character following a comma at the end of a data definition statement.

SOURCE PROGRAM SIZE RESTRICTIONS.

The DC 1000 is organized so that the literal pool and the symbolic table follow the coded portion of the program. With the extension of core, more literal definitions and symbols may be accommodated for a given source program. Because of the statement by statement processing of the source data by the DC 1000 Assembly program, the generated object programs may extend to any size as long as the number of literals and symbols do not exceed the specified limits.

Because of the limited core in a 4K byte processor, a given source program which is assembled on this size machine is restricted to 60 symbols maximum.

OPERATION.

The DC 1000 Assembly program is fully operable within a basic 4K-DC 1000 Computer. An ASR 33 Teletypewriter is required for input/output.

The DC 1000 assembler is designed for up to three different passes of the source program. After the first pass, the source program must be reread for each subsequent pass. Either pass II or pass III may be executed following pass I. Multiple executions of pass II or pass III may be made following a single execution of pass I. The setting of the SENSE switches determines which pass is to be executed (table 3-2).

Table 3-2

SENSE Switch Settings for Assembly Program Passes

	SENSE Switch Settings		
Pass	Switch 1	Switch 2	Switch 3
I	Down	Down	Down
II	Up	Up	Down
III	Up	Down	Up

B 3500 ASSEMBLER.

Essentially, the B 3500 Assembler is a duplicate of the DC 1000 Assembler. The input and output formats are the same as for the DC 1000 Assembler. Features, such as card input, magnetic tape input/output, and additional pseudo instructions make this assembler more desirable from both a coding and operational viewpoint. The

assembler is written in COBOL and generates DC 1000 object code that may be loaded as though it had been assembled on the DC 1000.

PSEUDO INSTRUCTIONS.

One of the extensions to the assembler is the pseudo instruction:

```
INC file-name
```

This pseudo instruction causes a symbolic file on disk to be completely inserted at this point in the program.

ASSEMBLER OPTIONS.

In addition to the extended source input, the B 3500 assembler has the options listed in table 3-3.

Table 3-3
B 3500 Assembler Options

Option	Description
SEQ	Ascending sequence check.
TAPE	Input is from source language tape (DCSOLT) with corrections, inserts and change cards from card reader. CARD is assumed if TAPE is omitted.
PTI	Input is from paper tape.
NEWT	Creates a DCSOLT initially from cards or an updated version using an old DCSOLT plus change cards.
LIST	Creates a full listing of the source language input with error messages as required.

Table 3-3 (cont)
B 3500 Assembler Options

Option	Description
DUMP	Dumps object code to tape upon end of assembly.
PTP	Punches object output on paper tape only.
PDSK	Produces object code output on paper tape and on disk.
+NNNNNNNN	Resequencing increment of source language input onto the output list and a new DCSOLT if applicable.
NNNNNNNN	Resequencing starting number of source language input.
"NON-NUMERIC LITERAL"	Name of object TAPE output file, DCTAPE is used if no other name is supplied.

For detailed information regarding both assemblers, refer to the DC 1000 Series Software manual.

EXECUTIVE ROUTINE.

In general, the Executive routine is the traffic officer of the DC 1000. It may or may not be application dependent and functions to interface the peripheral service routines. Some of the functions

that must be performed by the Executive routine are:

a. Define input and output buffer areas.
b. Specify the service routine function:
    1) Status requests.
    2) Processing requests.
c. Specify language conversions.
d. Supply communication formats.
e. Maintain line disciplines.

The specific information that must be provided by the Executive routine to the peripheral services, is discussed in detail, under the appropriate routine description.

PERIPHERAL SERVICE ROUTINES.

The peripheral service routines provide the necessary software to interface the DC 1000 with Burroughs peripheral devices. Routines to service the following equipment are currently available:

a. Line Printer.
b. Card Reader.
c. Card Punch
d. Single Line Control.
e. Paper Tape Punch and Reader.

In general, it is the function of the peripheral service routine to either transmit or receive data to or from a buffer area. It is the responsibility of the Executive routine to fill and empty these buffer areas prior to invoking the use of a peripheral service routine. The following paragraphs will discuss in detail each of the peripheral service routines.

LINE PRINTER SERVICE ROUTINE.

The Line Printer Service routine provides the necessary software to convert 132 characters in memory from eight-bit USASCII to six-bit BCL (appendix A) and output this code to the line printer controller.

It is the responsibility of the Executive routine to pass the buffer address and the format word (table 3-5) to the Line Printer Service routine.

INSTRUCTIONS.

The set of exclusive instructions used by the Line Printer Service routine are listed in table 3-4:

Table 3-4

Line Printer Service Routine Instructions

Type	Mnemonic	Function
Function	FUN 1D	Enable interrupts.
Function	FUN 3D	Disable interrupts.
Function	FUN 5D	Initialize controller.
Byte transfer	BTO 1D	Load data register.
Byte transfer	BTO 3D	Load format register.
Sense	SEN 1D	Detect parity or sync error.
Sense	SEN 3D	Detect end of page.
Sense	SEN 5D	Detect paper motion complete.
Sense	SEN 7D	Detect print cycle complete.
Sense	SEN BD	Detect printer ready.
Sense	SEN DD	Detect buffer ready.

ROUTINE ENTRY.

Entry to the Line Printer Service routine is made by a Branch and

Mark instruction (BRM) to the symbolic location PL. The contents of the C register determines the function to be performed by this routine. If the C register is positive, a print cycle is executed; if the C register is negative a device status is being requested.

STATUS ENTRY.
When the Line Printer Service routine is entered with a request for device status (C register is negative), one of the following status words is returned (figure 3-7):

a. If a print cycle is currently being executed, the status word is returned to the C register with the least significant bit set (bit 0), indicating that the printer is busy.

b. If a print cycle is not in operation, the status of the printer following the last print cycle is returned to the C register. Any error conditions generated during the last print cycle would be indicated at this time.

Example:

The code to request status of the Line Printer Service routine is:

```
SP2      Set operand precision to 2.
LOD      Load negative value in the C register.
BRM PL   Branch and Mark to symbolic location PL.
```

The status request should be made prior to each print cycle to insure that the previous print cycle was complete and free from errors. Failure to check status before executing a print cycle may result in errors going undetected and therefore unresolved.

Table 3-5
Line Printer Format Words

A Register (HEX Code)	Function Performed
00	Print and no space.
10	Print and single space.
20	Print and double space.
01	Print and skip to top of page.
0C	Print and skip to bottom of page.

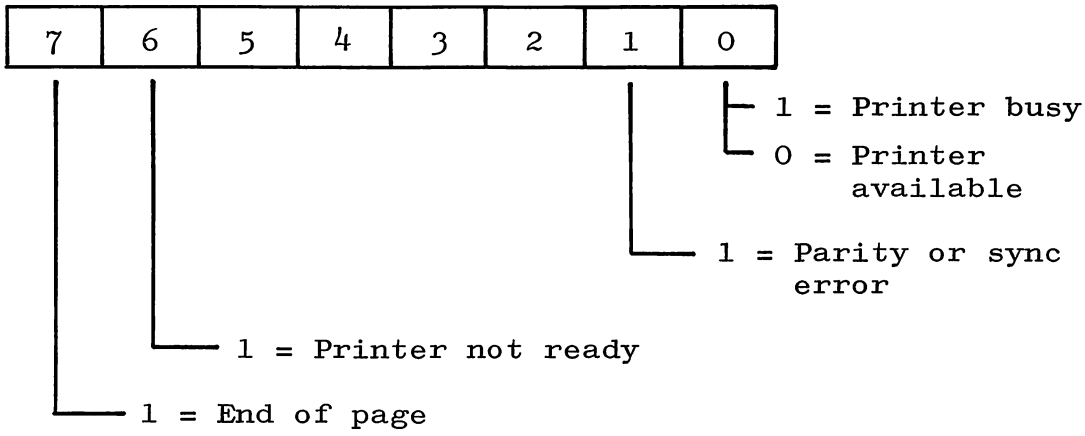


Figure 3-7. Line Printer Status Word

PRINT CYCLE ENTRY.

When the Line Printer Service routine is entered with a request to execute a print cycle (C register is positive), the A register contains the format word (table 3-5) and the C register contains the starting address of the output buffer.

Upon entry, the Line Printer Service routine will save the contents of the Index register. The Index register will be restored prior to returning control to the Executive routine at the completion of a print cycle.

PRINTING.

The Line Printer Service routine will take one USASCII character at a time from the executive buffer area and convert it to the BCL printer code (table 3-4). The executive buffer area must contain 132 characters plus one trailing blank character. Any unused locations will be supplied with blank characters.

When all 133 bytes of information have been transferred to the printer buffer, the format word is then output to the line printer. The transfer of the format word causes the actual printing to occur.

RETURN CONTROL.

After transfer of the format word is initiated, control is returned to the Executive routine. The Index register is restored; overflow is reset; and the status word, in the C register, is set to indicate that the printer is busy.

Example:

The code necessary to have the Line Printer Service routine print one line is:

```
SP1      Set operand precision to 1.
LOD      Load format word to C register.
TCA      Transfer C register to A register.
SP2      Set operand precision to 2.
LOD      Load buffer start address into C register.
BRM PL   Branch and Mark to symbolic location PL.
```

SPACING.

Spacing and skipping may be accomplished independently of printing by transferring a format word to the Line Service routine. However, this mode is operational only when the executive buffer area for the line printer contains all blanks.

CONFIGUREMENTS AND REQUIREMENTS.

HARDWARE.

a. DC 1000 with 4K memory.
   (Routine requires approximately 380 bytes).

b. Line printer controller.

c. Line printer.

SOFTWARE.

a. Line Printer Service routine.
b. Executive routine.

COMPATIBLE SERVICE ROUTINES.

a. Single Line Control.
b. Card Reader.
c. Card Punch.

CHARACTER SET.
Refer to appendix A.

CARD READER SERVICE ROUTINE.
The Card Reader Service routine is designed to provide a programed interface capable of converting Burroughs Common Language (BCL) or EBCDIC code to the DC 1000 internal character representation (USASCII). This routine stores the characters from one card in a buffer area which is accessible by the Executive routine.

The Card Reader Service routine consists of two parts: The first part is an initialization section which processes status requests and/or initializes the card reader interrupts and the feed cycle. The second part inputs the punch code to the CPU, converts it to USASCII, and modifies the status word to make it current.

ROUTINE ENTRY.
Entry to a Card Reader Service routine is made by a Branch and

Mark instruction (BRM) to the symbolic location RC. The contents of the C register determines the function to be performed by this routine.

STATUS ENTRY.

When the Card Reader Service routine is entered with a status request (C register is negative), the following actions occur:

a. The status word is loaded into the least significant eight bits of the accumulator (figure 3-8).

b. The precision bit is set to one byte.

c. Return is made to the Executive routine.

Example:

The code necessary to request status of the Card Reader Service routine is:

```
SP2      Set operand precision to 2.
LOD      Load negative value to C register.
BRM RC   Branch and Mark to symbolic location
         RC.
```

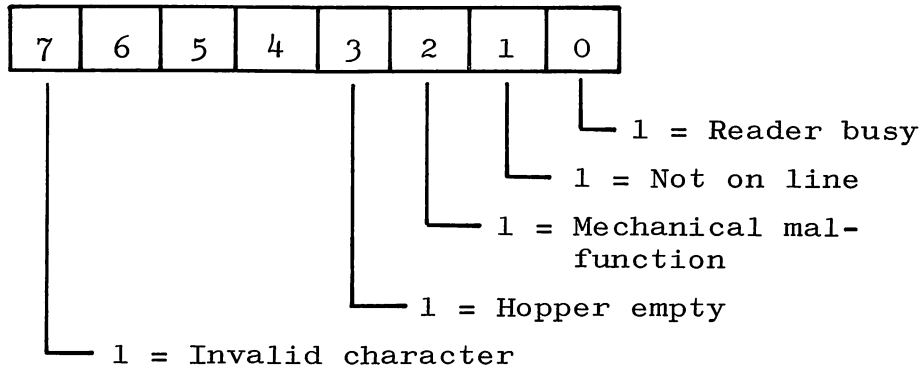


Figure 3-8. Card Reader Service Routine Status Word

INSTRUCTIONS.

The exclusive set of instructions used by the Card Reader Service routine are listed in table 3-6.

Table 3-6
Card Reader Service Routine Instructions

Type	Mnemonic	Function
Function	FUN 05	Enable interrupts.
Function	FUN 25	Initiate cycle.
Function	FUN 45	Disable interrupts.
Function	FUN 65	Initialize controller.
Byte transfer	BTI 05	Input data from card to computer.
Sense	SEN 05	Sense reader on line.
Sense	SEN 25	Sense reader malfunction.
Sense	SEN 45	Sense data ready.
Sense	SEN 65	Sense card reader ready.

READ ENTRY.

If the call to the Card Reader Service routine is an attempt to initiate the reading of a card, the C register should be loaded with the starting address of the data buffer. If the card to be read is EBCDIC, overflow should be set (SOF); otherwise BCL code is assumed. Operand precision should be set to two bytes (SP2), and a Branch and Mark instruction (BRM) to symbolic location RC should be executed.

Control will be returned to the Executive routine with a status word in the least significant byte of the accumulator. This status word will indicate one of the following conditions:

a. If the status word is zeros, the attempt was successful. The card reader is now busy, and will remain busy until an end-of-card interrupt is processed.

b. If the returned word is odd, the attempt was not successful because the card reader was busy at the time that the attempt was made.

Example:

The code necessary to have the Card Reader Service routine read a card is:

```
SP2       Set operand precision to 2.

ROF       Reset overflow (read BCL).
or
SOF       Set overflow (read EBCDIC).

LOD       Load start-of-buffer address into C
          register.

BRM RC    Branch and Mark to symbolic location RC.
```

DATA READY ENTRY.

Not under control of the Executive routine; this part of the Card Reader Service routine is entered when a Data Ready interrupt is generated by the card reader controller. The service routine remains in the interruptable environment and responds to the interrupt by executing two Byte Transfer In commands (BTI). These two bytes of data are converted to USASCII characters (appendix A) and then stored in the buffer area of the Executive routine.

Control is returned to the Executive routine after each two-byte word is read, converted and stored. The registers, the overflow status, and the operand precision of the Executive routine remain unchanged.

END-OF-CARD ENTRY.

Entry to this part of the Card Reader Service routine is not under control of the Executive routine; it is entered when an End-of-Card interrupt is generated by the card reader controller. The service routine remains in the interruptable environment and responds to the interrupt by performing the following functions:

a. Reinitializes the card reader controller which resets the interrupt.

b. Checks for invalid characters and sets the error bit in the status word to reflect the current condition.

c. Resets the busy bit of the status word to indicate that the card reader is no longer busy.

d. Returns controls to the Executive routine with the registers, the overflow status and the operand precision unchanged.

CONFIGURATIONS AND REQUIREMENTS.

HARDWARE.

a. DC 1000 with 4K memory.
(Routine requires approximately 460 bytes)

b. Card Reader Controller.

c. Card Reader.

SOFTWARE.

a. Card Reader Service Routine.

b. Executive Routine.

COMPATIBLE SERVICE ROUTINES.

a. Single Line Control.

b. Card Punch.

c. Line Printer.

CARD PUNCH SERVICE ROUTINE.

The Card Punch Service routine provides a programed interface capable of converting the DC 1000 internal character representation (USASCII) to either Burroughs Common Language (BCL) or EBCDIC punch code. This routine will output the proper sequence of codes from the executive buffer area, to punch one card, a row at a time, across the 80 columns of a standard data card.

This is a two-part service routine: The first part is an initialization section which processes status requests and/or generates the actual card punch code output buffer. The second part outputs the punch codes to the controller and modifies the status words to make it current.

ROUTINE ENTRY.

Entry to the Card Punch Service routine is made by a Branch and Mark instruction (BRM) to the symbolic location PC. The contents of the C register determines the function to be performed by this routine.

STATUS ENTRY.

When the Card Punch Service routine is entered with a status request (C register is negative), the following actions occur:

a. The status word is loaded into the least significant byte of the accumulator (figure 3-9).

b. Precision is set to one byte.

c. Return is made to the Executive routine.

Example:

The code necessary to request status of the Card Punch Service routine is:

SP2	Set operand precision to 2.
LOD	Load negative value to C register.
BRM PC	Branch and Mark to symbolic location PC.

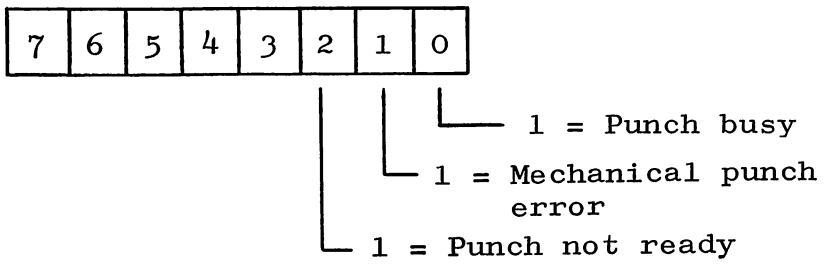


Figure 3-9. Card Punch Service Routine Status Word

INSTRUCTIONS.

The exclusive instructions that are used by the Card Punch Service routine are listed in table 3-7.

Table 3-7
Card Punch Service Routine Instructions

Type	Mnemonic	Function
Function	FUN 07	Enable interrupts.
Function	FUN 27	Disable interrupts.
Function	FUN 47	Select main stacker.
Function	FUN 67	Select auxiliary stacker.
Function	FUN 87	Initiate cycle.
Function	FUN A7	Initiate controller.
Byte transfer	BTO 07	Output data from computer to card.
Sense	SEN 07	Sense punch ready.
Sense	SEN 27	Sense punch busy.
Sense	SEN 47	Sense punch error.

Table 3-7 (cont)
Card Punch Service Routine Instructions

Type	Mnemonic	Function
Sense	SEN 67	Sense first four rows.
Sense	SEN 87	Sense data buffer ready.

PUNCH ENTRY.
If the call to the Card Punch Service routine is an attempt to initiate the punching of a card, the C register should be loaded with the starting address of the data buffer. Any data buffer of less than 80 characters will be filled with trailing blanks. If the card is to be punched in EBCDIC, overflow should be set (SOF); otherwise BCL punch code is assumed. The operand precision should be set to two bytes (SP2), and a Branch and Mark instruction (BRM) to symbolic location PC should be executed.

Control will be returned to the Executive routine with a status word in the least significant byte of the accumulator. This status word will reflect one of the following conditions:

a. If the status word is zeros, the attempt was successful. The card punch is now busy and will remain busy until an End-of-Card interrupt is processed.

b. If the status word is odd, the attempt was not successful because the card punch was busy at the time that the attempt was made.

Example:

The code necessary for the Card Punch Service routine to punch a card is:

SP2	Set operand precision to 2.
ROF	Reset overflow (punch BCL).
or	
SOF	Set overflow (punch EBCDIC).
LOD	Load start-of-buffer address into C register.
BRM PC	Branch and Mark to symbolic location PC.

DATA NEEDED ENTRY.
Entry to this part of the Card Punch Service routine is not under control of the Executive routine; it is caused by a Data Needed interrupt. The service routine remains in the interruptable environment during execution and responds to the interrupt by executing 80 BTO's to the card punch controller. Each set of 80 BTO's, provides the controller with the data necessary to punch one row in the card.

The data is obtained from the service routines buffer which was filled during the punch entry portion of the Card Punch Service routine.

Control is returned to the Executive routine after each row is punched. The registers, the overflow status, and the operand precision of the Executive routine remain unchanged.

END-OF-CARD-ENTRY.
Entry to this part of the Card Punch Service routine is not under control of the Executive routine; it is caused by the End-of-Card interrupt. The service routine remains in the interruptable environment and responds to the interrupts by executing the following functions:

a. Reinitializes the card punch controller which resets the interrupt.

b. Checks the punch errors and sets the error bit in the status word to reflect the current condition.

c. Resets the busy bit of the status word to indicate that the card punch is no longer busy.

d. Returns control to the Executive routine with the registers, the overflow status, and the operand precision unchanged.

CONFIGURATION AND REQUIREMENTS.

HARDWARE.

a. DC 1000 with 4K memory.
(Routine requires approximately 930 bytes)
b. Card Punch Controller.
c. Card Punch.

SOFTWARE.

a. Card Punch Service Routine.
b. Executive Routine.

COMPATIBLE SERVICE ROUTINES.

a. Single Line Control.
b. Card Reader.
c. Line Printer.

SINGLE LINE SERVICE ROUTINE.

The Single Line Service routine will transmit and receive data through the 201 type modem according to Standard Burrough Communication Procedures 12849006. The Exective routine has the responsibility of supplying the communication formats and line disciplines. The Single Line Service routine operates similar to the card punch routine with certain additional functions required for data communication. This routine will recognize 1967 USASCII codes. On input, it will store in the buffer all characters starting

after the last SYNC up to but not including the BCC. The ETX, ETB, or EOT character will signal the end of input. This routine will check for character parity and message parity. Message parity will be checked beginning after the first STX or SOH up to and including the ETX, EOT, or ETB. On output, this routine will generate parity for all characters and generate message parity (BCC) for the entire message starting after the STX or SOH up to and including the ETX, EOT or ETB character. The BCC shall have character parity. The routine will handle timeouts as defined in the Standard Burroughs Procedures.

The Single Line Service routine is in two parts: Interface subroutines and interrupt subroutines. There are three interface subroutines and four interrupt subroutines. The interface subroutines are closed subroutines which are called with parameters by the Executive routine. The interrupt subroutines monitor the status of the hardware and perform the character by character operations during message transmission.

INTERFACE SUBROUTINES.
The three interface subroutines are:

a. Initialization.
b. Input driver.
c. Output driver.

INITIALIZATION.
The Initialization subroutine performs the following services:

a. Initializes the single line controller.

b. Initializes the input and output drivers.

c. Makes preparations for message input.

d. Provides the capability of checking the status of the current input operation.

This routine must be called by the Executive routine before the input or output drivers can be called. It may be called at any other time to re-initialize the system.

INITIALIZATION ENTRY. The initialization routine is called with a Branch and Mark instruction (BRM) to the symbolic location MM. The The C register will contain zero or the address of a five-character field. If the C register contains zero, the previously entered codes will be used. If the C register contains an address, the five-character field at that address will be transferred into a table and the new codes will be used for subsequent messages. The format of the field is as follows:

TABLE LOC	SYN	Character code
LOC+1	ETX	Character code
LOC+2	ACK	Character code
LOC+3	NAK	Character code
LOC+4	EOT	Character code

The sync code (SYN) is the bit combination to be used to detect character sync on input. Four of these characters are sent preceeding each output message. The ETX character is used by both the input and output interrupt service routines to detect the end of message. The ACK, NAK, and EOT are line control characters. The ACK code is the acknowledgement of a message; NAK is a negative acknowledgement code; and EOT indicates end of transmission.

On exit, the eight least significant bits of the C register contains the input status byte. (This byte is discussed in the next section and the operand precision is set to 1.)

The initialization routine enables the single line controller interrupts and also sets the DC 1000 interrupt mask to allow modem interrupts. However, the DC 1000 Interrupt Block Enable (IBE) command is not executed. This command must be executed by the Executive routine before data transmission will begin.

THE INPUT DRIVER.

The input driver causes message input to be initiated and/or provides status on the current operation.

The calling sequence for the routine is a Branch and Mark instruction (BRM) to the symbolic location MI. On entry, the input driver expects the address of an input buffer or a negative number in the C register.

If a buffer address is specified, the current status is examined. If message input can be initiated, it will be, and a zero status byte will be returned.

If status is requested (negative number in the C register) or if message input cannot be initiated, a status byte will be returned. Figure 3-10 illustrates the status byte of the input driver.

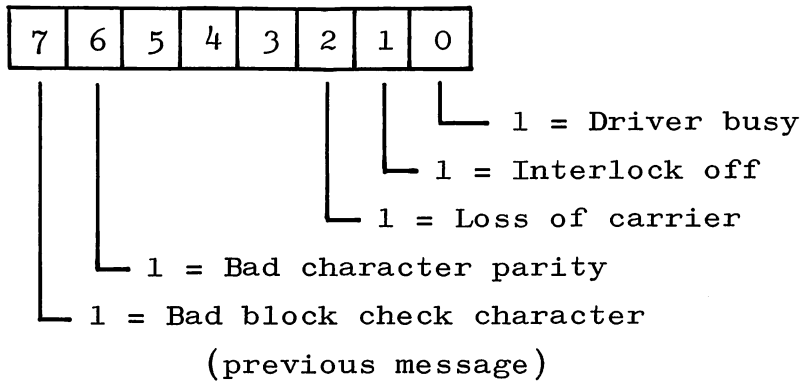


Figure 3-10. SLC Input Driver Status Word

When the driver is no longer busy, only those status bits affecting the last transfer are cleared after the examination. For example, if status is requested and bits two and seven were set and then status was again requested, only bit two would be set. Bit seven is only concerned with the previous message while bit two is concerned with the line status.

The following steps should be taken to initiate message input:

a. Wait for driver to become not busy.

   1) If the status byte is zero, the previous message was received successfully.

   2) If the status byte was non-zero, the message was not received successfully, or a condition exists which prevents message input.

b. Take the appropriate action on the status returned.

c. Request message input initiation.

   1) If the status is zero, the transfer was initiated successfully.

   2) If the status is non-zero, the operation cannot be initiated.

d. Take appropriate action on the status returned.

THE OUTPUT DRIVER.

The output driver initiates message output and/or provides status information.

The calling sequence for the output driver is a Branch and Mark instruction (BRM) to the symbolic location MO. The driver expects the address of a message to be output or a negative number in the C register.

If the address is provided, the current status will be examined, and if possible, message output will be initiated and a zero status will be returned.

If the status was requested (negative number in the C register) or if the message cannot be transmitted, a status byte will be returned. Precision will be set to 1. Figure 3-11 illustrates the status byte of the output driver.

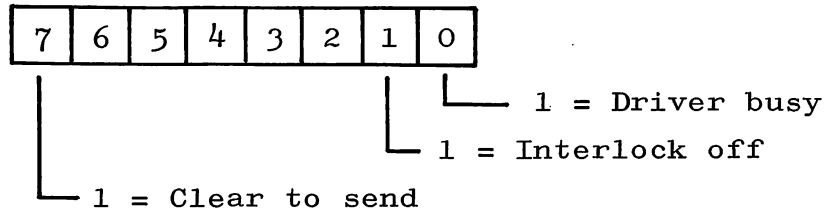


Figure 3-11. SLC Output Driver Status Word

The following sequence can be used to initiate message output:

a. Wait for the driver to become not busy.

b. If the status byte is zero, message output can be initiated.

c. Take action on error status.

d. Initiate message output.

e. If the status byte is zero, the operation was initiated successfully.

INTERRUPT ROUTINES.
The driver contains a common interrupt supervisor routine which decodes the common interrupts (3 through 10) and calls the indicated routine. The routines supplied are:

a. Ring-Interlock interrupt handler.
b. Clear to Send interrupt handler.
c. Carrier interrupt handler.

These routines note the change in state of the indicated signal and cause the corresponding flag to be set in the input and output driver routines.

INTERRUPT VERSUS SENSE MODE. The software may be written to react to I/O interrupts or wait in sense loops for controller logic events to occur. The approach used would depend on the service load of the controlling program with respect to other peripherals in the data terminals.

SENSE MODE. The following sense terms are enabled by the On state of the dataset interlock lead and will always be false until interlock comes on:

SEN	06	Bit Mode
SEN	26	Write Buffer Ready
SEN	46	Read Buffer Ready
SEN	66	Carrier On
SEN	86	Clear to Send
SEN	E6	Odd Parity

The remaining sense terms, SEN C6-Ring Indicator and SEN A6- Interlock, are not interlocked and give valid responses whenever interrogated.

INTERRUPT MODE. There are provisions for six interrupt conditions in the controller logic. They may be grouped as follows:

Data Interrupts:	Write Buffer Ready
	Read Buffer Ready
Status Interrupts:	Clear to Send
	Carrier On
	Ring Indicator
	Interlock

All of these conditions are gated with an enabling term set true by an Enable Interrupt command (FUN 26) and conversly, all can be disabled with a Disable Interrupt command (FUN 46). In addition, Write and Read Buffer Ready, Clear to Send, and Carrier On are further enabled or disabled with the Interlock signal from the dataset.

When used, these conditions cause the interrupt to the program to be latched on until the program responds appropriately. The proper responses are given below.

Write Buffer Ready Interrupt.

This interrupt indicates that the write shift register has shifted a character out and requires another character.

The program must service this request within one bit time (416 microseconds at 2400 baud) to maintain the message flow. If the program does not respond, the controller hardware will force each successive bit to be a mark (1) until the next BTO or initializing command.

The Write Buffer Ready interrupt is unlatched by the execution of a BTO command, a Controller Initialize command (FUN E6), a turn on Request to Send command (FUN 86) or a manual System Reset from the front panel of the DC 1000.

Read Buffer Ready Interrupt.

This interrupt indicates that the receive section has shifted in another bit or character.

The program must service this interrupt with a BTI command within one bit time (416 microseconds at 2400 baud) or that bit or character will be overwritten by the following input.

The Read Buffer Ready interrupt is unlatched by the execution of a BTI command, a Controller Initialize command (FUN E6), or a manual System Reset from the front panel of the DC 1000.

Clear to Send Interrupt.

This interrupt is true whenever the Clear to Send signal from the dataset changes state from the Off to On or On to Off.

The interrupt is cleared by the execution of a SEN 86 command, a Controller Initialize command (FUN E6) or a manual System Reset.

Carrier on Interrupt.

This signal goes true whenever the Carrier On signal from the dataset changes state from Off to On or from On to Off.

The interrupt is cleared by the execution of a SEN 66 command, a Controller Initialize command (FUN E6) or a manual System Reset.

Ring Indicator Interrupt.
This signal goes true whenever the Ring Indicator signal from the dataset changes state from Off to On.

The interrupt is cleared only by the execution of a SEN C6 command.

Interlock Interrupt.
This interrupt is set whenever the Interlock signal from the dataset changes state from Off to On or from On to Off.

The interrupt is cleared by the execution of a SEN A6 command, a Controller Initialize command (FUN E6) or a manual System Reset.

INSTRUCTION SET SUMMARY.
The I/O instructions that are used exclusively by the Single Line Control Service routine are listed in table 3-8.

Table 3-8
Single Line Service Routine Instructions

Instruction	Use
FUN 06	Places the controller in the receive bit mode for acquiring character sync by inputting bit patterns at each receive clock time.
FUN 26	Enables the six interrupt terms to be gated onto the I/O bus. Note that four of the interrupt conditions are also gated by the Interrupt signal from the data set.
FUN 46	Disables all of the six interrupt terms in the hardware.
FUN 66	Turns on the Data Terminal Ready line (RC) to the 201 data set.

Table 3-8 (cont)
Single Line Service Routine Instructions

Instruction	Use
FUN 86	Turns on the Request to Send line (RS) to the 201 data set.
FUN A6	Turns off the Request to Send line of the 201 data set.
FUN C6	Places the controller in the receive word (character) mode to input eight-bit characters from the receive shift registers.
FUN E6	Initializes the controller to its lowest mode by disabling the interrupt terms, setting the controller to the receive bit mode, turning off Request to Send, turning off Data Terminal Ready, and clearing the write and receive shift registers.
BTO 06	Transfers an eight-bit byte from the DC 1000 to the write shift register.
BTI 06	Transfers the contents of the eight-bit receive shift register to the DC 1000.
SEN 06	A true response to this status sense indicates that the controller is in the receive bit mode.
SEN 26	True response when the write buffer is ready to be loaded.
SEN 46	True response when the read buffer is ready to be read.

Table 3-8 (cont)
Single Line Service Routine Instructions

Instruction	Use
SEN 66	True response when the Carrier On signal from the dataset is on.
SEN 86	True response when the Clear to Send signal from the data set is on.
SEN A6	True response when the Interlock signal from the data set is on.
SEN C6	True response when the Ring Indicator signal from the dataset is on.
SEN E6	True response when the character in the receive register has odd parity.

PAPER TAPE SERVICE ROUTINE.

The Paper Tape Service routine provides a programmed interface capable of converting the DC 1000 internal character code USASCII to Burroughs Common Language (BCL) paper tape punch code (appendix D). The routine will output either converted or binary bytes to the paper tape punch. On input, the service routine will convert, when necessary, from BCL to USASCII and stores the data in an executive designated buffer area.

The Paper Tape Service routine consists of two parts: the first part is an initialization section which processes status requests and starts peripheral processes. The second part processes interrupts from the paper tape peripheral unit, either to input or output

data, and modifies the status needed to make it current.

INSTRUCTIONS.

The set of exclusive instructions that are used by the Paper Tape Reader Service routine are listed in table 3-9.

Table 3-9
Paper Tape Reader Service Routine Instructions

Type	Mnemonic	Function
Function	FUN 02	Start Reader.
Function	FUN 22	Stop Reader.
Function	FUN 42	Select binary mode of input.
Function	FUN 62	Select alphanumeric mode of input.
Function	FUN 82	Enable controller interrupts.
Function	FUN A2	Disable controller interrupts.
Function	FUN C2	Backspace tape.
Function	FUN E2	Rewind tape (reel mode only).
Sense	SEN 02	Sense not parity error.
Sense	SEN 22	Sense data ready.
Sense	SEN 62	Sense not end of tape.
Sense	SEN 82	Sense not end of message (stop code).
Byte transfer	BTI 02	Input one byte from reader to accumulator.

Table 3-9 (cont)

Paper Tape Reader Service Routine Instructions

Type	Mnemonic	Function
Byte transfer	BTO 02	Initialize reader (reset).

The exclusive instructions that are used by the paper tape punch are listed in table 3-10.

Table 3-10

Paper Tape Punch Service Routine Instructions

Type	Mnemonic	Function
Function	FUN 09	Select binary mode.
Function	FUN 29	Select alphanumeric mode.
Function	FUN 49	Enable interrupts.
Function	FUN 69	Disable interrupts.
Function	FUN 89	Punch one character.
Function	FUN A9	Start punch.
Function	FUN C9	Stop punch.
Function	FUN E9	Reset controller.
Sense	SEN 09	Sense motors not stopping.
Sense	SEN 29	Sense not stop code.
Sense	SEN 49	Sense not end of tape.
Sense	SEN 89	Sense punch ready.
Sense	SEN A9	Sense data needed.
Byte Transfer	BTO 09	Output one byte of data to punch controller from accumulator.

ROUTINE ENTRY.

Entry to the Paper Tape Service routine is made by a Branch and Mark instruction (BRM) to the symbolic location PT. The X register determines which device is being accessed; X register = zero for the paper tape reader, X register = non-zero for the paper tape punch. The C register determines the function to be performed by the service routine. If the C register is negative, then the call is for status only. If the C register is positive, then the call is to process tape.

STATUS ENTRY.

When the Paper Tape Service routine is entered with a request for device status (C register is negative), one of the status words is returned to the least significant byte of the accumulator and operand precision is set to one byte (figures 3-12, 3-13).

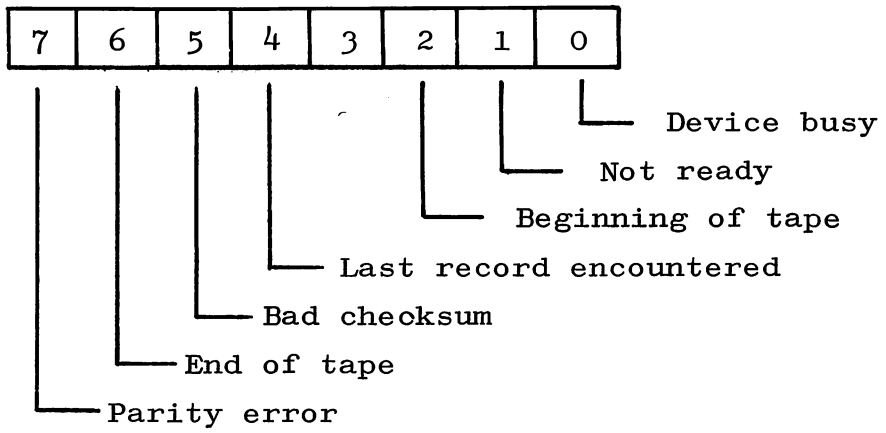


Figure 3-12. Paper Tape Reader Status Word

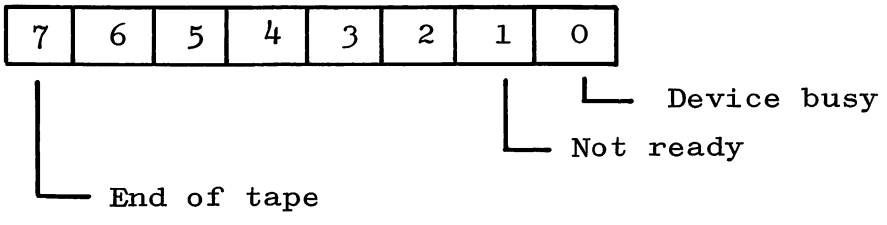


Figure 3-13. Paper Tape Punch Status Word

PROCESS TAPE ENTRY.

If the call to the Paper Tape Service routine is an attempt to initiate the processing of tape, the C register should be loaded with a positive number.

In the case of a call to initiate the reading of a tape in binary mode, the C register contains one of two positive numbers:

a. If the number is two, the tape is assumed to be in absolute addressing format.

b. If the number is six, the tape is assumed to be in relocatable addressing format.

The four bytes of information immediately following the BRM instruction are used by the service routine. The format and description of these bytes are given in table 3-11.

Table 3-11

Parameter Byte Format

Byte Number	Buffer	Paper Tape Reader	Paper Tape Punch
1 and 2		Input buffer starting address.	Output buffer starting address.
3	Negative	Backspace one record. This number must be the one's complement of the number of characters in the record.	Record/character count designates the number of characters to be punched.
	Zero	Rewind the reel to the reflective strip.	

Table 3-11 (cont)
Parameter Byte Format

Byte Number	Buffer	Paper Tape Reader	Paper Tape Punch
3 (cont)	Positive (non-zero)	Read forward (non-zero) one record.	Record/character count designates the number of characters to be punched.
4		Interrupt line used by the device = 1.	Interrupt line used by the device = 2.

The mode of punching or reading is given by the overflow status. If overflow is set, the mode is alphanumeric. If overflow is reset, the mode is binary. In the alphanumeric mode, characters are converted and stop codes are recognized. In the binary mode, neither conversion nor stop code recognition is in effect.

Control is returned to the Executive routine at the fifth byte following the BRM instruction. The appropriate status word is loaded into the least significant byte of the accumulator. If the returned status word is binary zeros, the initiation attempt was successful. The paper tape peripheral unit is now busy and will remain so until end-of-data is processed. If the returned status word is non-zero, the attempt was not successful as the device was not available at the time of the initiation attempt.

PUNCH INTERRUPT ENTRY.
This part of the service routine is entered when a data interrupt is generated by the paper tape punch controller. The service routine changes environments through the use of a CSQ instruction and checks for the following conditions:

SUFFICIENT TAPE LEFT. If false, the end of tape bit of the status word is set, and if data is yet to be output, a branch is taken to the data output part of the routine. If an end of data has occurred, control is returned to the Executive routine. If true, no action is taken.

If the above test is true, the interrupt is assumed to be for data. The byte to be output is loaded, and if alphanumeric mode, converted to tape punch code. The byte is output, and tested to determine if that was the last data byte; if so, a flag is set. On the next data needed interrupt, if this flag is set, control is returned to the Executive routine with all registers, the precision and the overflow status unchanged. The status word is set to not busy to indicate that the punch can be used again.

READER INTERRUPT ENTRY.

This part of the service routine is entered when an interrupt is generated by the paper tape reader controller. The service routine changes environments through the use of a CSQ instruction, and checks for the following conditions:

BEGINNING OR END OF TAPE. If true, the Beginning of Tape bit of the status word is set on. Control is returned to the Executive routine with all registers, precision, and overflow status unchanged. If false, no action is taken.

PARITY ERROR (ALPHA MODE). If true, the tape is stopped, backspaced to the last record mark and restarted reading forward. This process is repeated for a maximum of three times for each parity error encountered. If the parity error still occurs on the third try, the tape continues forward, and the Parity Error Encountered bit of the status word is set. A byte of binary zeros is stored in the data

buffer at the position corresponding to the character in error. Control is returned to the Executive routine.

If false, no action is taken.

If none of the above error conditions exist, the interrupt is assumed to be for correct data. The data byte is inputted, converted (if in Alpha mode) to USASCII, and stored. If the byte is the last of the record, the status word is set to indicate not busy. Control is returned to the Executive routine.

On return to the Executive routine, only the status word is changed. The registers, operand precision, and overflow status remain unchanged.

TAPE FORMAT.

In alphanumeric mode (for both punch and reader), all records are 127 characters or less in length. The end of a record is denoted by a BCL carriage return and two line feeds (hexadecimal $6F5A5A).

In binary mode, the record length is variable, and is given by the first byte of each record (maximum 127 bytes). The last byte of the record is the checksum (Exclusive OR) of all previous bytes of the record (including length, starting address, etc.). All records are preceded with three rub outs, and the last record is also followed by three rub outs and at least one frame of leader (blank or hex $00).

It should be noted that the Paper Tape Punch routine does no tape formatting whatever. Any leader, visual aids, checksum, etc., must be output as a part of the data given to the punch routine.

MISCELLANEOUS SOFTWARE.

MATHEMATICAL ROUTINES.

The DC 1000 mathematical routines are grouped into three major subsections: fixed point arithmetic, floating point arithmetic, and floating point functions. The software is in the form of subroutines, which are called from the other programs. The fixed

point subroutines are: Multiply, Divide, Decimal to Binary Conversion, and Binary to Decimal Conversion. The floating point arithmetic routines include: Add, Subtract, Multiply, Divide, Integer to Floating Point Conversion, and Floating Point to Integer Conversion. There are also several floating point utility subroutines included in this section. The floating point function subroutines are: Integer Part, Fractional Part, Square Root, Exponential Function, Logarithmic Function, Exponentiation, Decimal to Binary Floating Point Conversion, Binary to Floating Point Conversion, Sine, Cosine, and Arctangent.

The mathematical software subroutines are intended to fulfill the mathematical requirements of a large class of computer users. The fixed point section is intended for those users who require only limited arithmetic capability. The fixed point subroutines together with the SUM and SUB machine instructions allow the user to perform integer arithmetic operations and conversions. The floating point arithmetic is intended for those users who require more accuracy, more flexibility, and greater number ranges. The floating point functions are also intended for those users who, in addition to arithmetic operations, need more powerful mathematical tools.

The DC 1000 mathematical software subroutines have been designed to work together as a system. Where possible, each subroutine calling sequence has been made compatible with the other subroutine calling sequences and exit formats. This allows a user to execute a string of Branch and Mark instructions (BRM) to the desired math routines to evaluate an expression without a series of formatting instructions between each call. In this way, the math subroutines act as an extended instruction set.

The math subroutines are programed to utilize only one environment. This allows the user to do simultaneous computing, and input/output processing utilizing to dual environment powers of the DC 1000.

The number formats and number ranges of the math subroutines have

been chosen to provide the most efficient utilization of the DC 1000 computer features and a wide area of application. Binary integers may be up to 32 bits long. Floating point numbers have an exponent range from 2^{-64} to 2^{63} (about 10^{-19} to 10^{18}). The floating point mantissa is 24 bits long which provides approximately seven significant digits. The size of the math routines are given in appendix D.

The routines will check for error conditions such as underflow, overflow, incorrect sign, and other conditions which are unacceptable for specific routines. If error conditions are found, the routine will return with an indication of the error found. In the case of overflow or underflow, the largest or smallest number is returned to allow continuation of calculations when the user's accuracy requirements will allow this.

The Fixed Point Multiply and Divide routines have been designed to have exact accuracy and fast execution. The floating point routines have been designed to maintain accuracy to the least significant bit of the mantissa, with a minimum of core required. The floating point design also optimizes speed, whenever possible, without incurring a corresponding size penalty. The results of these design criteria are very tightly coded subroutines which combine to form an integrated system of math routines.

GENERAL. All of the math subroutines are designed in a general format to faciliate using the routines as a system. Arguments and parameters are passed to and from the subroutines in the A, C, and X registers. A minimum amount of this information is altered by the operation of the subroutine. If a floating point subroutine has only one argument, it will be passed in the four-byte accumulator. If the subroutine has two arguments, one will be in the four-byte accumulator and the other in memory at an address specified in the X register. Results will be returned in the four-byte accumulator. All subroutines operate in one environment and use or

alter only the registers in that environment.

LABEL FORMAT. The math routines have been written using a label format which allows all of the routines to be assembled together, and at the same time allows the user to differentiate between possible math routine labels and all other labels. Each math routine has a unique symbolic name. This name is used as the operand of the BRM instruction which calls the routine. Each routine has associated with it a unique letter which is used as the first character of all labels used in the routine. The second character of any label in a math routine is a digit. Therefore, there is no possibility of label conflict as long as the user does not use the symbolic name of any math routine assembled with the main program, or use a letter followed by a digit. The format leaves the user with more than 675 unused combinations.

In each math routine, the digit part of the label generally implies the use of the label. The digit zero identifies the starting address of the temporary storage block when temporary storage is required. Nine identifies a two-byte area used as temporary storage for the X register. These two bytes are imbedded in the routine itself but may still be used as temporary storage when the routine is not active. Digits one and greater are used for branch points within the routine. Digits eight and less are used for additional imbedded temporary storage or for constants.

OPERATING CONFIGURATION. The math routines are designed to operate in the minimum configuration of a 4K memory DC 1000 with a Model 33 Teletype. There are no limitations on using the routines with any other configuration of the DC 1000 Computer.

The math routines are designed to operate in one environment only of the DC 1000 without altering the conditions of the other environment. The one environment may be either interruptable or non-interruptable. The SENSE switches are not used with the math routines.

Several of the math routines operate independently. Most of the routines call other math routines to perform their functions. A matrix which defines the subroutines that are called by each routine, and the size of each routine is given in appendix C, table C2.

Table C1 in appendix C represents the minimum, maximum, and average time of the execution for each routine including the time consumed by each subroutine called by a particular math routine.

Table 3-12
DC 1000 Fixed Point Mathematical Routines

Mnemonic	Function	Description
MU	Multiply	Forms the product of two, signed, 16-bit binary integers. The returned product is a 32-bit binary integer with correct algebraic sign.
DI	Divide	Form the quotient of a signed, 32-bit binary dividend divided by a signed, 16-bit integer divisor. The returned quotient is a signed, 16-bit binary integer. A 16-bit positive remainder is also returned. The nonrestoring method is used.
DB	Decimal to Binary Conversion	Convert a decimal integer to a four-byte binary number. The number to be converted may be from one to 10 digits long in USASCII or eight-bit BCD code.
BD	Binary to Decimal Conversion	Converts a four-byte binary number in the accumulator into USASCII coded decimal digits in an area specified in memory.

Table 3-13

DC 1000 Floating Point Mathematical Routines

Mnemonic	Function	Description
RX	Load X Immediate	Utility routine used by other math routines to load the X register with the two bytes following the call instruction without altering any other conditions.
SE	Separate Exponent and Mantissa	Separate the exponent from the mantissa of a floating point number, returns the signed mantissa in the four-byte accumulator, and returns the signed exponent in the two-byte X register.
FC	Construct Floating Point Number	Constructs a floating point number given the mantissa in the four-byte accumulator and the exponent in the X register. This routine is the inverse of the Separate Exponent and Mantissa routine.
FN	Floating Point Normalize	Normalizes a floating point number by left shifting the mantissa and decrementing the exponent until the most significant bit of the mantissa is one.
IF	Fixed Point to Floating Point Conversion	Utility routine to convert a four-byte (32-bit) signed binary integer to floating point format.

Table 3-13 (cont)

DC 1000 Floating Point Mathematical Routines

Mnemonic	Function	Description
FI	Floating Point to Fixed Point Conversion	Converts a floating point number 32-bit, signed, binary integer. Any fractional part of the number is lost. This routine is the inverse of the Fixed Point to Floating Point conversion routine.
FA	Addition	Adds the floating point numbers by adding the number addressed by the X register to the number in the accumulator. This routine calls subroutines: SE, FN, and RX.
FS	Subtract	Subtracts a floating point number stored in memory at the address in the Index register from the floating point number contained in the four-byte accumulator. This routine calls FA which in turn calls SE, FN, and RX.
FM	Multiply	Multiplies the floating point number in the accumulator by the floating point number in memory at the address contained in the X register.
FD	Divide	Divides the floating point dividend in the accumulator by the divisor in memory at the address contained in the X register.

Table 3-14

DC 1000 Floating Point Functions

Mnemonic	Function	Description
IP	Integer Part	Extracts the integer part of a floating point number. The fractional part of the number is lost. This routine calls SE, FC, FN, and RX subroutines.
FP	Fractional Part	Extracts the fractional part of a floating point number. The integer part of the number is lost. This routine also calls SE, FC, FN, and RX subroutines.
SQ	Square Root	Calculates the square root of a floating point number using the Newton's iterative method. The number must be positive. This routine also calls the SE, RX, FD, and FA subroutines. FN is called indirectly through the FA subroutine.
EX	Exponential Function	Computes the exponential value of the floating point number X entered (e^X). This routine also calls FI, FA, FS, FM, FD, and FP. RX, SE, FC, and FN are called indirectly.
LN	Natural Logarithm	Computes the natural logarithm of the floating point number entered (in X). This routine also calls

Table 3-14 (cont)
DC 1000 Floating Point Functions

Mnemonic	Function	Description
LN (cont)		the SE, IF, FM, FA, FS, FD, and RX subroutines. FN is called indirectly.
FE	Exponentiation	Computes A**B where A and B are floating point numbers. This routine also calls the LN, FM, and EX subroutines. RX, SE, FC, FN, IF, FI, FA, FS, FD, and FP are called indirectly.
QF	Decimal to Floating Point Conversion	Converts a decimal fraction and exponent in memory into a four-byte, floating point, binary number. The mantissa may be from one to 10 signed decimal digits, and the exponent may be one or two signed decimal digits. This routine calls the RX, SE, IF, FI, FC, FM, EX, and DB subroutines. FN, FA, FS, FD, and FP are called indirectly.
FQ	Binary to Decimal Conversion	Converts four-byte, binary, floating point number into a decimal fraction and exponent in memory. The decimal number will be seven, signed decimal digits followed by two, signed decimal digits representing the exponent. This routine calls the RX, SE, FC, IF, FM, FP, and EX subroutines. FN, FA, FS and FD are called indirectly.

Table 3-14 (cont)
DC 1000 Floating Point Functions

Mnemonic	Function	Description
SI	Sine	Computes the sine of the floating point number X entered. The function is approximated by a truncated Taylor's series. This routine calls the RX, FA, FS, FM, FD, and FP subroutines. SE, FC, and FN are called indirectly.
CO	Cosine	Compute the cosine of the floating point number X entered. The function is approximated by a truncated Taylor's series. This routine calls the SI and FS subroutines. RX, SE, FC, FN, FA, FM, FD, and FP are called indirectly through SI.
AT	Arctangent	Computes the arctangent of a floating point number. This routine calls the FA, FS, FM, and FD subroutines. RX, SE, and FN are called indirectly by FA.

BINARY LOADER.

The Basic Bootstrap is a program designed to load the Binary Loader into core and transfer it to the loader. Since the bootstrap must be manually entered into the computer via the console, it is designed to be as small as possible. The bootstrap is entered near the top of any memory sector and reads in the loader until it is over-

layed, at which time it transfers to the loader and halts.

The Binary Loader loads formatted object tape, as generated by the assembler or by a paper tape memory dump, into the computer memory. The checksum is computed and checked, and control is transferred as directed, or a halt is performed upon completion of the load.

AUTOMATIC BOOTSTRAP LOADER.

a. Place the binary Loader into the teletype reader.

b. Actuate the RESET switch.

c. Activate the AUTO BOOTSTRAP button.

d. The Binary Loader will read into the system.

e. The system is now set to load subsequent programs.

RELOCATE AND LINKING LOADER.

This loader routine will relocate and link together the binary object load modules produced on punched paper tape by the DC 1000 assembly program for relocatable object code. The loader routine will allow programs to be relocated at load time from their originally assigned routine area to any other suitable area in the DC 1000 memory. The loader will effect linkages between programs which are assembled independently but which are loaded together.

The loader routine will permit a user to combine separately assembled segments of a single program or to integrate independently assembled programs at load time into whatever memory configuration is suitable for a given computer run. The loader will establish the linkages which are necessary to effect the transfer of control and the referencing of data between programs during execution.

SOURCE TAPE CORRECTION.

The Source Tape Correction program (STC) provides the ability to edit a DC 1000 assembly language source tape using the teletype as

the interface device. STC provides the user with the ability to ADD, DELETE, CHANGE, LIST, and PUNCH out any number of source lines up to the entire contents of the 127 line, 3000 character buffer.

The editor may be used when generating a source tape the first time to permit corrections before producing a clean source tape. When used in this manner, a line feed is generated automatically with each carriage return. The STC keeps continuous track of the total number of lines in the buffer as well as the number of the present line being processed, and allows access to any line or lines indexed by these numbers. The numbers are consecutive decimal count calling the first line in the buffer one. The numbers are continuously updated as each change is made to the program. Whenever the user requests a listing of the current form of his program, each line of the listing is preceded by its line number.

The STC program eliminates the tedious task of correcting the symbolic source tapes off-line. An appreciable amount of time and labor is saved and the ease of operation of the STC allows any user to quickly take advantage of the program. One use of the STC program may be to change a fixed format source tape generated by the assembler to a free format which would shorten the source tape size.

GENERAL DEBUGGING AID.

The General Debugging Aid (GAID) is a system of routines which provide on line debugging capability to the user. Utilizing the teletype as the input-output device, the user is able to perform the following functions:

a. Display and alter variable fields of memory.

b. Branch to a program with all conditions of the system specified.

c. Branch from a program to BAID and save all conditions at the time of the branch.

d. Produce a reloadable punched paper object tape of desired areas of core.

e. Execute the binary loader to input object tape into the computer.

f. Construct patches for a program in a variable size and location patch area.

g. Perform program run time displays of variable areas of core.

h. Remove a patch or run time display branch from a program and restore the previous instructions.

GAID is designed to provide the user with a high level of interaction with his program during debugging operations. The patching and dumping capabilities are designed to free the user from the mechanics of constructing the actual sequences involved. The automation of these procedures save the user an appreciable amount of time due to the sophistication of the variable length command and multiple addressing features of the DC 1000. All required addressing mode changes and pointer construction are done automatically, thereby freeing the user from this responsibility. In addition, the user has control over the location and size of the patch area and pointer area. Several different areas may be specified during a debugging operation.

Another feature of GAID is the ability to command it from the paper tape as well as from the keyboard. This enables a user to prepare a program debugging sequence off-line before the computer is available. The user may then load his program and ready his tape, which has commands for GAID, when the computer becomes available. After execution, he may analyze the results of his efforts (again off-line) by reading the teletype listing produced. If additional debugging is required, the user may enter, through the tape reader, previous command tapes to set up his program patches and changes instead of re-entering those commands from the keyboard. This type of usage

permits a much higher utilization of the computer.

GAID is coded to occupy a minimum of core, leaving a maximum amount for the user's program. If the user requires additional core space when debugging, a subset of GAID called the Basic Debugging Aid (BAID) is provided which is about 450 bytes smaller than GAID, but provides basic debugging ability.

BASIC DEBUGGING AID (BAID).

BAID is a system of routines which provides on-line debugging capabilities to the user. This aid is a subset of the General Debugging Aid (GAID) and is provided for use when the program to be debugged is too large to allow space for GAID. This aid provides all the necessary features for debugging, but omits the complex features provided by the automatic patching as defined under the heading GAID. The functions provided by BAID are:

a. Display and alter variable fields of memory.

b. Branch to a program with all conditions of the program specified.

c. Branch from a program to BAID and save all conditions at the time of the branch.

d. Produce a reloadable punched paper object tape of desired areas of core.

One feature of BAID is the ability to command it from the paper tape as well as from the keyboard. This enables a user to prepare a program debugging sequence off-line before the computer is available. The user may then load his program and ready the tapes, which contain the commands for BAID, when the computer becomes available. After execution, he may analyze the results (again off-line) by reading the teletype listing produced. If additional debugging is required, the user may enter through the tape reader, previous command tapes to set up his program changes instead of entering them from the key-

board. This type of usage permits a much higher utilization of the computer.

BAID is coded to occupy a minimum amount of core, leaving a maximum amount for the user's program. In addition, the organization of BAID is designed to allow a user to partially overlay it with his program and still have available the basic features needed for debugging. This is done in steps so that as increasing amounts of BAID are overlayed, an increasing number of functions are lost, saving the basic debugging capabilities.

BASIC COMPUTER TEST.
The Basic Computer Test consists of the following major sections:

a. Executive routine.
b. Instruction tests.
c. Teletype tests.
d. Memory tests.

The test program is designed to enable the user to determine whether the basic DC 1000 computer configuration is functioning properly, or, if not, provide aids to determine the exact nature of the malfunction.

Through the use of the teletype keyboard, and the use of the SENSE switches, the user has full control over the execution of the test program.

The discrete tests are each addressable by typing a corresponding four-letter mnemonic and may be run once or continously through SENSE switch selection.

The consistent assignment of SENSE switch definitions has been maintained throughout all of the tests to provide a uniform operating interface.

The Basic Computer Test program contains all of the tests necessary to check the minimum configuration DC 1000 computer and is contained

on one object tape which will load in approximately two minutes.

The user may select, through SENSE switch setting, a mode which causes the program to halt upon the detection of each error condition; allowing that error to be explored in detail. Or, as an alternative, the test program will continue to completion, producing a standardized error printout upon the detection of each error.

INSTRUCTION BOUNDARY ADDRESS TEST.

The Instruction Address Test program tests all classes of machine instructions at a critical memory address boundary to see if the instruction crosses the boundary correctly. A critical memory address is one where the last eight bits are $(11111111)_2$ or $FF ($ denotes hexadecimal notation). For example, an instruction at $FF crosses the boundary correctly if the P register is incremented to $100 or $101 depending on instruction length. The program tests the instructions by executing them at the critical boundary. If the boundary is crossed successfully, the tests will continue; otherwise, an error message is outputted. The program is needed as a supplement to the Instruction Test diagnostic. The program is easy to use and can be easily modified to perform additional tests.

SECTION 4
DATA COMMUNICATIONS

GENERAL.

The Multi-Line Control (MLC) interfaces the DC 1000 series computers with up to 64 full-duplex or half-duplex lines. The lines receive and supply serial data while the DC 1000 receives and supplies parallel data. Each parallel data character passing between the DC 1000 and the MLC must be accompanied by an address byte specifying the line which is receiving or supplying the character (figure 4-1).

The time-shared, 85-bit W register and an associated 86-bit memory, which can have a capacity of up to 64 words, provide the means for servicing up to 64 lines simultaneously. (IC memory must be incremented in two-bit increments and is therefore actually 86 bits long; however, only 85 bits have significance.) Associated with each line being serviced is a Line Control Word which contains data being assembled or disassembled and in addition contains all necessary control and status information for the line. Lines are scanned under control of a Scan counter which advances at a 1.075 megaHertz rate. Thus, each line is scanned approximately once every 59.5 microseconds for a period of approximately 930 nanoseconds. At the start of each 930 nanosecond period, the Line Control Word associated with the line to be scanned is transferred from memory to the W register. Appropriate updating of the Line Control Word is performed while it resides in the W register. At the end of the scan period, the word is returned to its location in memory.

Each of the lines being serviced is connected continuously to an associated line adapter. When a line is being scanned, its associated line adapter logic is connected to the W register. This permits serial data transfers between the line and the W register. The line scan repetition frequency is high relative to the highest bit rate that is accommodated; thus, the bit transfer timing is not significantly affected by the fact that the W register is time-shared.

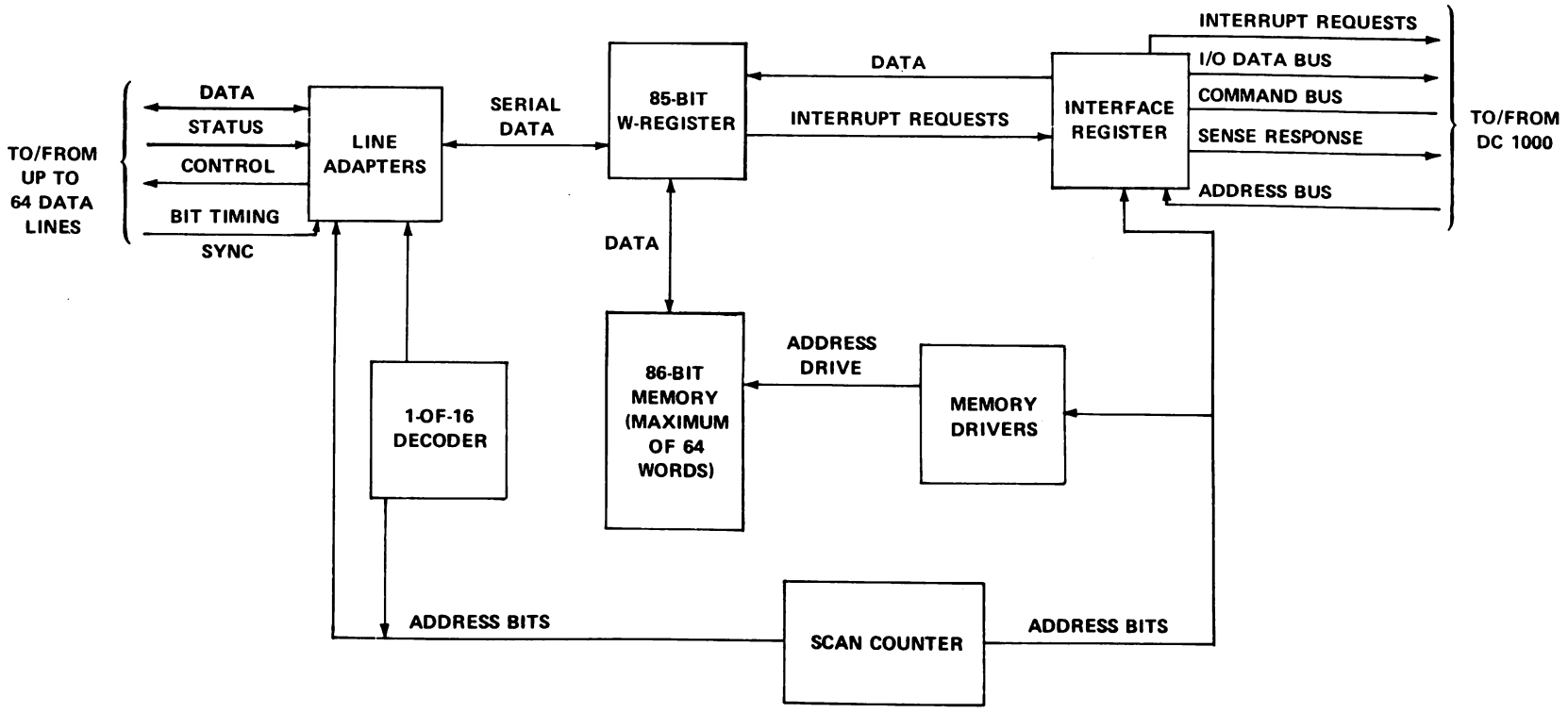


Figure 4-1. MLC Overall Block Diagram

NOTE

These line adapters are
actually line terminators.

When the Line Control Word associated with the line being scanned contains a complete data character, an input transfer to the DC 1000 is required. When a character is needed from the DC 1000 for transmission to the line, an output transfer is required. In either case, an interrupt request is transmitted to the Interface register. If the Interface register is not busy, the address of the line, and if applicable, a complete character from the line are transferred into the Interface register. The interrupt request is then transmitted to the DC 1000. When a character transfer from the line to the DC 1000 is required, the DC 1000 picks up the address byte and the data character. It then sets the Interface register to a Not Busy status. When a character transfer from the DC 1000 to the line is required, the DC 1000 first picks up the address of the requesting line. It then transfers the data character to the Interface register where it is held until the next time that the requesting line is scanned. At this time, it is transferred from the Interface register to the W register.

When an error condition associated with the line being scanned is sensed, an Error Interrupt request is transmitted to the Interface register. This is handled in much the same way as a Data Interrupt request except that it is necessary to transfer only one byte to the DC 1000. This eight-bit byte contains the six address bits and two control bits which define the particular error that has been sensed.

Communication with a switched line is initiated when a ring indication is received from the line during the period when it is being scanned. This causes a Ring Interrupt request to be transmitted to the DC 1000. In response to this request, the DC 1000 picks up the address of the line making the request. It then supplies the instructions required to initialize the Line Control Word associated with that line to place it in the appropriate data

transmission mode.

The DC 1000 can initiate transfers that are implemented by the Interface register. By such transfers, the DC 1000 loads the appropriate control information into each Line Control Word as it resides in the W register or reads various portions of addressed Line Control Words. When the DC 1000 has gained control of the Interface register for such a transaction, it transfers an address byte to the Interface register. The first six bits of this byte specify the line that has its Line Control Word being addressed and the remaining two bits specify the particular field of the Line Control Word that is being addressed. If the DC 1000 is loading data into a Line Control Word, it also transfers an information byte to the Interface register. When the Interface register is ready to unload or accept an information byte, the correct scan period must be selected. The transfer is timed to occur during the scan period when the line with its address held in the Interface register is being scanned.

LINE TO DC 1000 COMMUNICATIONS.

Figure 4-2 illustrates the flow of data from a line to the DC 1000. Serial data enters the MLC via a line adapter and is shifted into the Character Assembly register. When a complete character has been assembled, it is transferred in bit-parallel form to the Character Assembly buffer. Each time that the Character Assembly buffer is filled, an interrupt request is transmitted to the Interface register (IR). As soon as control of the IR has been achieved, the character is transferred from the Character Assembly buffer to the IR data register. At the same time, the line address is transferred from the Scan counter to the IR address register. At this time, the IR address register also receives control bits C and D from the interrupt request logic. These bits provide the means of indicating a bad parity or overflow condition. With the data and line address stored in IR, the Data Interrupt request is transmitted to the DC 1000 which first picks up the address byte and then the data byte. Address and data bytes are transferred to the DC 1000 I/O data bus through the IR input data gates.

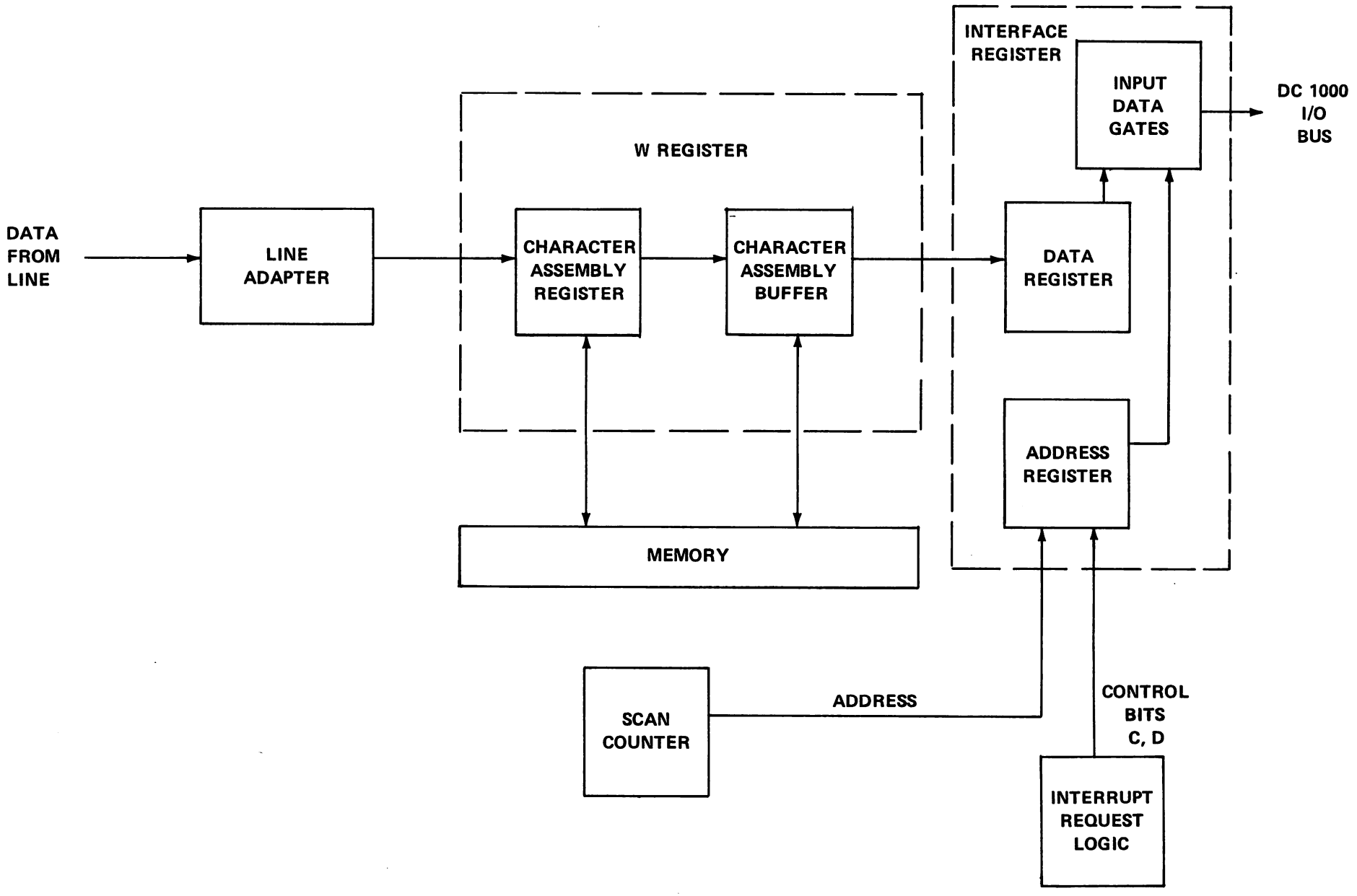


Figure 4-2. Data Flow From Line to DC 1000

The Character Assembly register and the Character Assembly buffer are sections of the time-shared W register. Thus, the data for each line resides in these sections for 930 nanoseconds out of each 59.5 microseconds and resides in its assigned memory location for the remainder of each scan cycle.

DC 1000 TO LINE COMMUNICATIONS.

Figure 4-3 illustrates the flow of data from the DC 1000 to an addressed line. The DC 1000 transfers the address byte and the data character which are loaded respectively into the IR address register and the IR data register. This occurs after a Data Interrupt request has been received indicating that the line needs a data character. With the address stored in its address register, the IR waits for a signal from its coincidence logic which indicates that the addressed line is being scanned. It uses this signal to time the transfer of the data character from the IR data register to the Character Disassembly buffer section of the W register. As soon as the Character Disassembly register is empty, indicating that the preceding character has been shifted out to the line, the data character is transferred from the Character Disassembly buffer to the Character Disassembly register. Since the Character Disassembly buffer is now available to accept another character, a Data Interrupt request is initiated at this time. The character in the Character Disassembly register is shifted out to the line adapter at the appropriate bit rate for application to the data line.

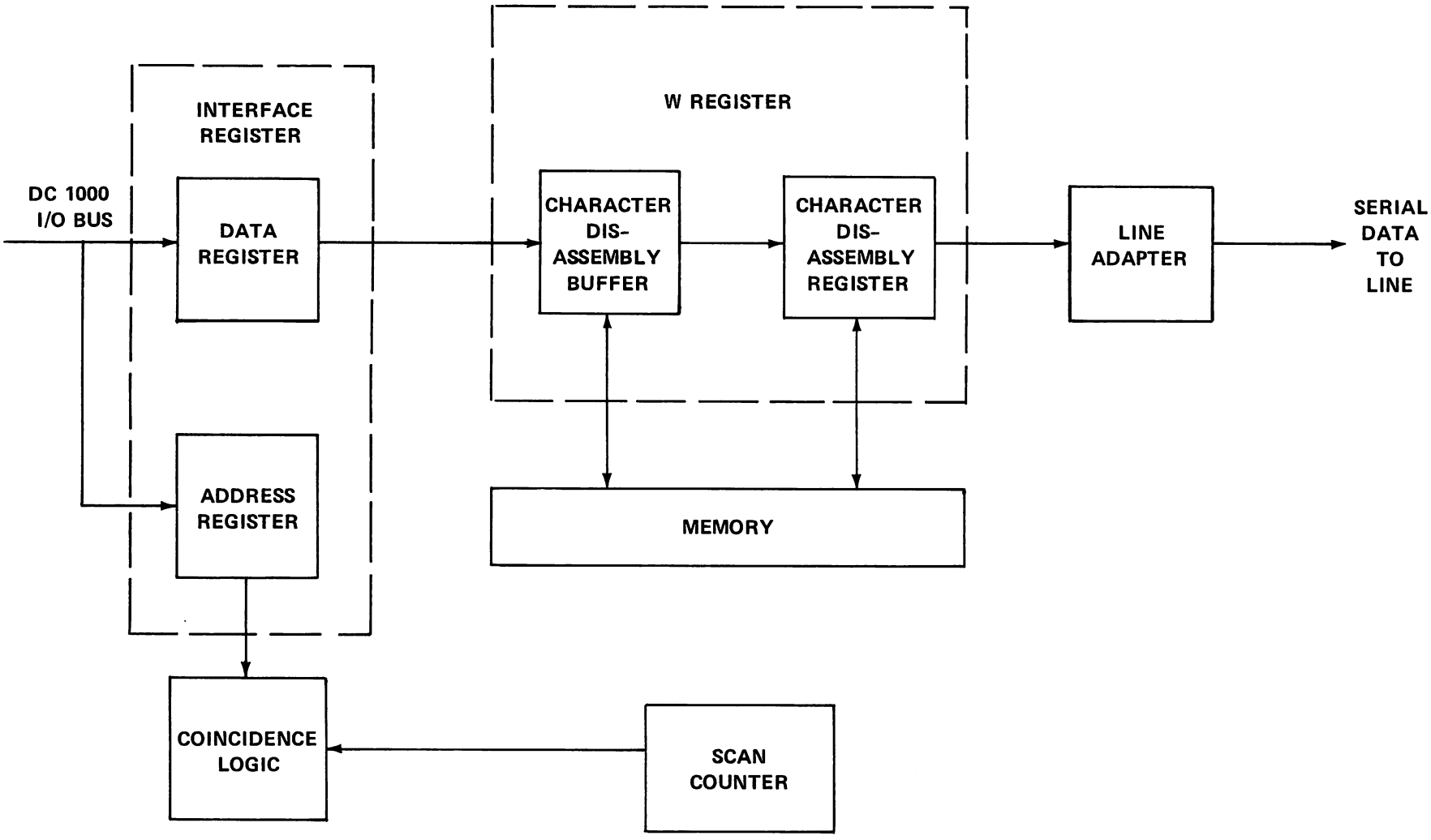


Figure 4-3. Data Flow From DC 1000 to Line

HARDWARE FUNCTIONS.

The Multi-Line Control (MLC) interfaces the DC 1000 with switched, leased and direct connect data communications lines. The bit handling capabilities of the MLC range from 45 to 4800 bits per second. The system block diagram is illustrated in figure 4-4.

The basic hardware functions are:

a. Establish synchronization.

b. Assemble and disassemble characters for bit-serial communication.

c. Provide a one-character buffer for each line.

d. Handle incoming and outgoing line control signals with the modem.

e. Check the parity of the incoming characters.

f. Generate parity for the outgoing characters.

g. Interrupt the processor bus to initate the transfer of data or control information to or from the MLC.

HARDWARE CONFIGURATION.

The MLC modular concept is illustrated in table 4-1. The number and types of modules required to configure a system depends upon the number of lines that the system must service. The modules required to configure a basic system include:

a. The basic electronics (multiplexor); capable of handling up to a maximum of 64 full duplex (or half duplex) lines.

b. The integrated circuit (IC) memory; capable of handling up to 16 full duplex (or half duplex) lines.

c. The Memory Drive logic; capable of handling up to 16 full duplex (or half duplex) lines.

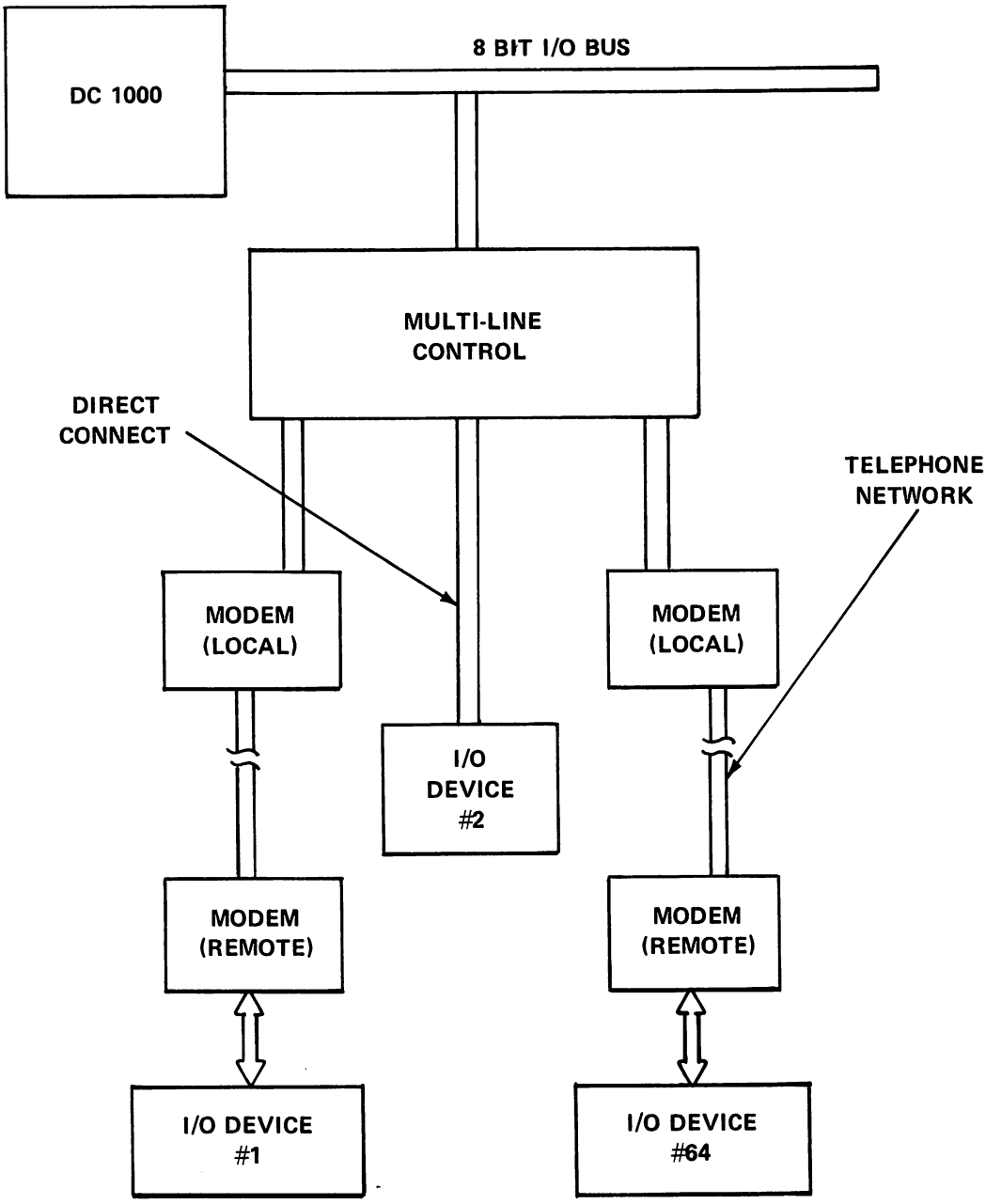


Figure 4-4. System Layout

d. The first power supply in an expansion chassis; capable of handling up to 32 full duplex (or half duplex) lines.

e. The line adapters; each capable of handling up to four full duplex (or half duplex) lines.

Table 4-1
System Configuration

Number of Lines	Number of Modules Required				
	Multi-plexor	IC Memory	Power Supply & Expansion Chassis	Memory Drive Logic	Line Adapters
2	1	1	1	1	1
4	1	1	1	1	1
16	1	1	1	1	4
18	1	2	1	2	5
32	1	2	1	2	8
34	1	3	2	3	9
48	1	3	2	3	12
50	1	4	2	4	13
64	1	4	2	4	16

The bidirectional information transfers between the MLC and the processor I/O bus are transacted in two-byte blocks. The first byte contains the line numbers along with the bits which indicate whether the second byte contains data or control information. Some of the control functions are:

a. Parity error.

b. Buffer overflow/underflow error.

c. Loss of carrier/sync.

d. Line inactivity time-out.

e. Modem ready/not ready.

FUNCTIONAL DESCRIPTION.

The MLC block diagram is shown in figure 4-6. The function, operation, and characteristics of each block are discussed below:

SCAN COUNTER AND DECODER.

The six-bit Scan counter is a synchronous, recycling counter that is capable of counting up to 64. The counter output is used by the memory control logic to generate various read/write signals for the IC memory. The four most significant bits of the Scan counter select one of 16 line terminators via the decoder. The two least significant bits are decoded on the line terminator to select one of four lines that are attached to the LT (figure 4-5).

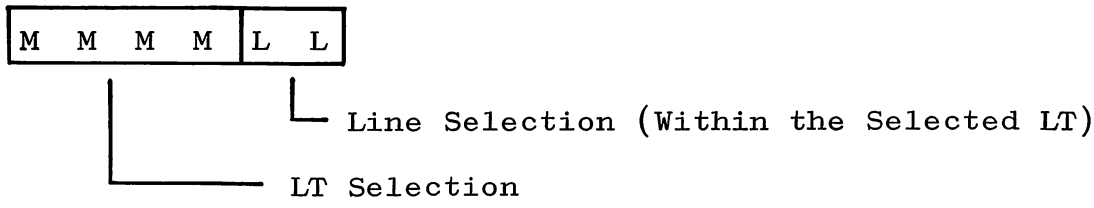


Figure 4-5. Scan Counter

Example:

With a Scan count equal to five, the following control signals are generated to read the appropriate LCW into the W register and to write the appropriate LCW in IC memory:

SCAN 5 to Line 5 (Generated on the LT)

READ 5 from IC Memory (Generated by Memory Control Logic)

WRITE 4 to IC Memory (Generated by Memory Control Logic)

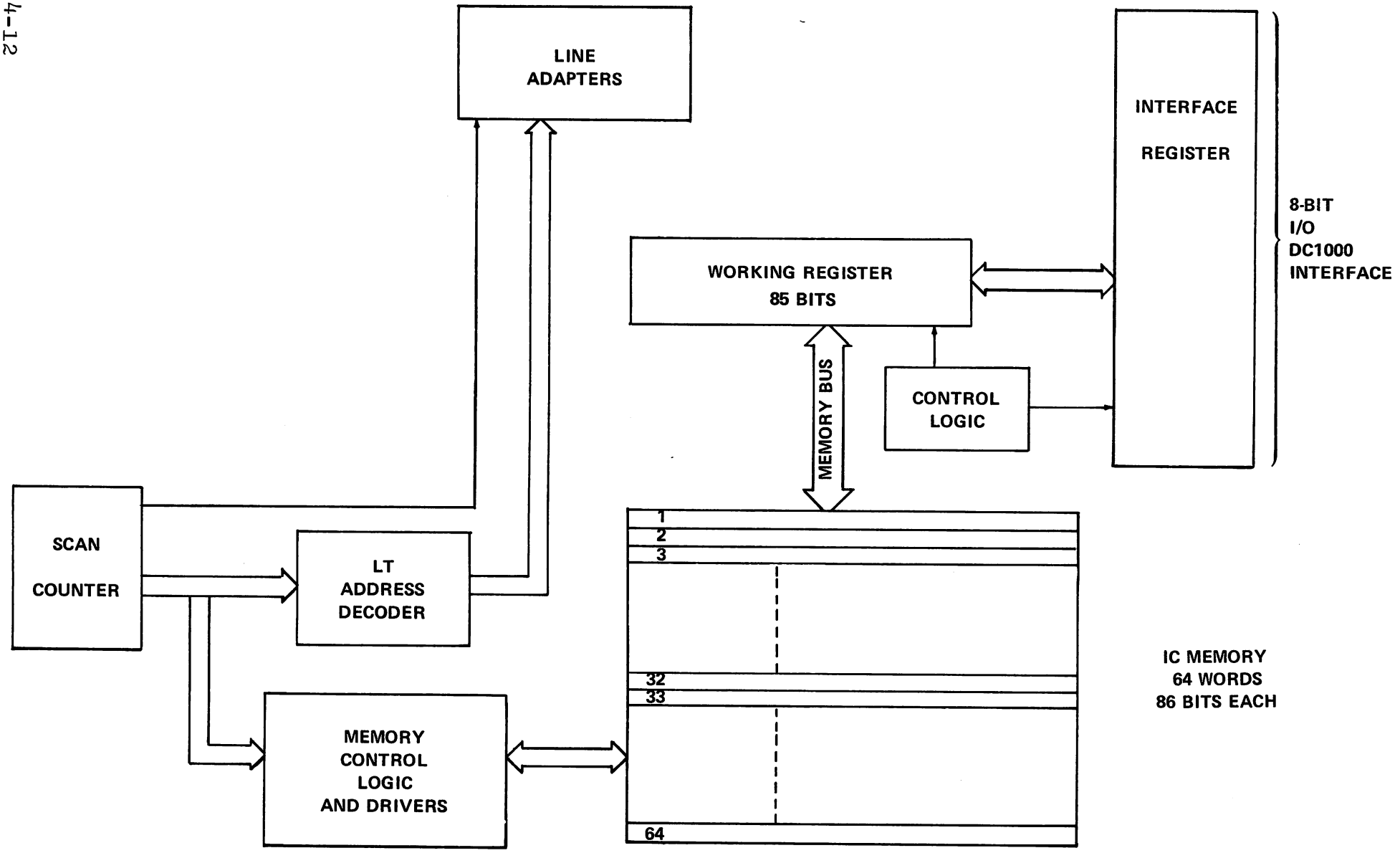


Figure 4-6. Block Diagram Multiline Control

LINE ADAPTERS (TERMINATORS).

Each line adapter handles four full duplex (or half duplex) data communications lines. There are no storage provisions for incoming signals; but, one-bit storage is provided for each outgoing signal. Each line adapter receives the two least significant bits of the Scan counter and one line bearing the adapter number from the decoder. These signals are decoded to generate four scans, one for each of the four ports of the adapter. All incoming and outgoing signals on the line adapter are gated with the appropriate scan signal. The same adapter can handle both the synchronous and the asychronous data transmissions (Bell 201, 202, and 103 Data Sets).

W REGISTER.

This 85-bit register updates the Line Control Words. The memory control logic exchanges LCWs between the IC memory and the W register via the memory bus. The W register and line adapter information exchange is via the W bus. Each LCW is stored in the W register for 930 nanoseconds. The W register fields and their functions are defined in table 4-2.

CHARACTER ASSEMBLY REGISTER (CAR). This ten-bit field [9:10] assembles the bit-serial character as it is received from the data communications line. The input stage is determined by character length which is decoded from bits 44 and 45 of the Line Identification field.

CHARACTER ASSEMBLY BUFFER (CAB). This eight-bit field [18:8] is used to store one character until it is requested by the processor. This field is used so that the Character Assembly register will be available for new information from the data communications line. Input to the buffer is from the Character Assembly register and output is to the Interface register.

Table 4-2

W Register Bit Assignments

Bit	Function	Bit	Function
0	LSB	40	ASYNC/SYNC — MSB
1		41	2 STOP BITS/1 STOP BIT
2	CHARACTER	42	PRIORITY — LINE
3		43	ODD/EVEN PARITY — ID
4	ASSEMBLY	44	CHARACTER — FIELD
5		45	LENGTH
6	BUFFER	46	DEVICE
7	(INPUT)	47	SPEED — LSB
8		48	RECEIVE MODE — LSB
9	MSB	49	TRANSMIT MODE — SEQUENCE
10	TRANSMIT MARKS	50	REQ INPUT TO DC 1000 — CONTROLLER
11	LSB	51	REQ OUTPUT FROM DC 1000
12		X3	OUTPUT PARITY/ACCEPT RING — MSB
13	CHARACTER	52	LSB
14	ASSEMBLY	53	
15		54	RECEIVE
16	BUFFER	55	
17	(INPUT)	56	STROBE
18	MSB	57	TIMER
19	LSB	58	
20		59	MSB
21	CHARACTER	60	HALF BIT TIME FLAG
22	DISASSEMBLY	61	LSB
23	BUFFER	62	
24		63	TRANSMIT
25	(OUTPUT)	64	STROBE
26	MSB	65	TIMER
27	LSB	66	
28		67	
29		68	MSB
30	CHARACTER	69	HALF BIT TIME FLAG
31		70	SYNC START BIT
32	DISASSEMBLY	71	SYNC CONTROL
33	REGISTER	72	PRIORITY MATCH BIT
34		73	OVERFLOW ERROR
35	(OUTPUT)	74	UNDERFLOW ERROR
36		75	PARITY ERROR
37	MSB	76	LOSS OF CARRIER
38	CLEAR TO SEND	77	LOSS OF CLEAR TO SEND
39	CARRIER ON	78	ECHO MODE
X0	TRANSMIT	79	LOSS OF INTERLOCK
X1	BREAK	X2	STOP OUTPUT INTERRUPTS
		Y0	RECEIVED BREAK

CHARACTER DISASSEMBLY BUFFER (CDB). This eight-bit field [26:8] provides temporary storage for the character that is about to be transmitted until such time when the Character Disassembly register is empty (has completed transmission of the present character).

CHARACTER DISASSEMBLY REGISTER (CDR). This 11-bit field [37:11] disassembles a character to provide bit-serial output to the data communications line. The output stage is bit 27 for asynchronous transmission and bit 28 for synchronous transmission as determined by bit 40 of the Line Identification field.

LINE IDENTIFICATION FIELD (LIF). This eight-bit field [47:8] provides the characteristics required by the MLC hardware to handle a data communications line. This field is program addressable, i.e., its contents can be changed by the software. During the initialization process, the software loads this area. The Line Identification field is defined in table 4-3.

Table 4-3
Line Identification Field

Bit	Status	Definition
40 =	0	Synchronous data transmission.
	1	Asynchronous data transmission.
41 =	0	One-stop-bit character.
	1	Two-stop-bit character.
42 =	0	Low priority line.
	1	High priority line.
43 =	0	Even parity for data received or transmitted.
	1	Odd parity for data received or transmitted.

Table 4-3 (cont)
Line Identification Field

Bit	Status	Definition
44 & 45 =	00	Five-bit data character.
	01	Six-bit data character.
	10	Seven-bit data character.
	11	Eight-bit data character .
46 & 47 =	00	Select device speed 1.
	01	Select device speed 2.
	10	Select device speed 3.
	11	Select device speed 4.

NOTE

Bits 46 and 47 are used to select device speeds for asynchronous devices only. Synchronous speed is determined by the data set.

SEQUENCE CONTROLLER (SC). This five-bit field [51:4] and [X3] indicates the state of the line, i.e., Transmit only, Receive only, Full Duplex, etc. Bit X3 is an extension of this field and is used for the detection of the Ring indicator from the data set and generation of output parity. The bit configurations and line states are defined in table 4-4.

Table 4-4

Sequence Controller Field Definition

Bits					Definition
X3	51	50	49	48	
0	0	0	0	0	Idle.
0	0	0	0	1	Receive mode; set by the DC 1000 software and enables the line to receive data.
0	0	0	1	0	Transmit mode; set by the DC 1000 software and enables the line to transmit data.
0	0	0	1	1	Full Duplex mode; set by the DC 1000 to allow the line to transmit and receive simultaneously.
0	0	1	0	1	Receive mode or full duplex and a character has
0	0	1	1	1	been assembled; bit 50 is set by the MLC to request a data transfer to the DC 1000 under interrupt control.
0	1	0	1	0	Transmit mode or full duplex and a character
0	1	0	1	1	has been disassembled; bit 51 is set to request a new character from the DC 1000 under interrupt control.
1	0	0	0	0	Idle and bit X3 is set under software control to detect a Ring Indicator signal from the data set. When detected, the MLC will generate a Ring interrupt and turn off bit X3.
1	0	0	1	1	Full Duplex
1	0	0	1	0	Transmit

Bit X3 on in both of these cases orders the MLC to generate either even or odd parity as determined by bit 43 of the LIF.

RECEIVE STROBE TIMER (RST). This nine-bit field [60:9] is used to determine bit strobe time for an asynchronous operation.

TRANSMIT STROBE TIMER (TST). This nine-bit field [69:9] is used to determine bit transmit times for asynchronous operations.

CONTROL FIELD (CF). This 17-bit field [10], [39:2], [79:10], [XO], [X1], [X2] and [YO] provides the control signals that are required to maintain transmissions between the data communications line and the DC 1000. The control bits are listed below:

Transmit All Mark Character.
This bit [10] is set with a special BTO command, Transmit All Mark Character. This bit is an extension of the Character Disassembly buffer and at CDB to CDR transfer time, will set bit 27 causing all Marks to be sent for one character time.

Clear To Send.
This bit [38] is set to indicate that the Clear To Send signal was received from the data set. This bit can be set only if the Interlock signal from the data set is also true.

Carrier Detect.
This bit [39] is set to indicate that the Carrier Detect signal from the data set was received. The Interlock signal from the data set must also be true before this bit can be set.

Sync/Start Bit.
The function of this bit [70] is dependent upon bit 40 which is the Synchronous/Asynchronous Transmission bit. In synchronous operation, this bit can be set only if bit 71 (Sync Control bit) has been previously set. This bit indicates that two sync characters have been received and the unit is now synchronized with the line. During asynchronous operation, this bit is set to indicate that a start bit has been received.

Sync Control Bit.
This bit [71] is used in conjunction with synchronous operation. It is set when the first sync character is recognized in the Character Assembly register as compared to the character placed in the Character Assembly buffer under software control.

Priority Match Bit.
When a high priority line (designated by bit 42) requests the Interface register and the Interface register is busy with some other line, the Lockout flip-flop is set along with bit [72] of the requesting line. When the Interface register is released, the high priority line with the match bit set will be the next line to obtain the interface.

Overflow Bit.
This bit [73] is used in both asynchronous and synchronous operations. It is set when a complete character is received in CAR and the previous character has not been transmitted to the processor from CAB.

Underflow Bit.
This bit [74] applies to synchronous operation only. It is set when the character in CDR has been completely transmitted and there is no character available in CDB. During synchronous operation, characters must be continuously transmitted in order to maintain synchronization.

Parity Error Bit.
This bit [75] is set when a parity error is detected in CAR during a receive operation. Odd or even parity checking is determined by bit [43] of the Line Identification field.

Loss Of Carrier Bit.
This bit [76] is set to indicate a Loss of Carrier during a receive operation. It works in conjunction with bit [39] (Carrier Detect). It is set if bit [39] had been previously set and the carrier from the data set is lost.

Loss Of Clear To Send.
This bit [77] is set to indicate a Loss of Clear to Send during a transmit operation. It works in conjunction with bit [38] (Clear to Send). This bit is set if bit [38] had been set previously and Clear to Send from the data set is lost.

Echo Mode Bit.
This bit [78] is set with the special BTO command Echo Mode. In this mode, when the character is transferred to CDR, the control field bit [51] remains set and bit [78] is set to inhibit the request of another data character. When a character is received from the line and is transferred to the processor, bit [78] is reset to allow another character request for transmission.

Loss Of Interlock Bit.
This bit [79] is used to indicate a Loss of Interlock from the data set during a transmit or receive operation. It works in conjunction with bit [38] (Clear to Send) or bit [39] (Carrier Detect). This bit is set if bit [38] or [39] had previously been set and the Interlock from the data set is lost.

Transmit Break Bit 1.
This bit [X0] functions as an extension of the Character Disassembly buffer and is set by the special BTO command 8F (Transmit All 0's Character). It allows all spaces to be transmitted for one character time.

Transmit Break Bit 2.
This bit [X1] is set from bit [X0] when a CDB to CDR transfer takes place. It forces the line to a space condition for one character time as a result of the special BTO command 8F.

Stop Output Interrupt.
This bit [X2] is set by the special BTO command $6F (Transmit All Stop Bits). This bit inhibits the request for the next data character that is to be transmitted. This causes a transmission

of all one bits continuously until a normal BTO command is executed.

Received Break Bit.
This bit [YO] is set when a break condition has been received from the line.

IC MEMORY.
The IC memory provides for storage of up to 64, 86-bit Line Control Words (LCWs). There is one LCW for each data communications line to provide for the assembly/disassembly of bits until a complete character has been accommodated. This is accomplished by bringing the contents of IC memory to the W register, adding the new bit coming in on the line or transmitting the next bit, and returning the partial character to the IC memory. The LCW for the next line is then brought to the W register and the process is repeated until all of the data communications lines have been serviced.

INTERFACE REGISTER.
The Interface register terminates the eight-bit I/O bus from the DC 1000. It also communicates with the W register to accomplish data transfers between the MLC and the DC 1000. The fields in this register are briefly defined in figure 4-7.

DATA TRANSFER REGISTER. This eight-bit field provides for bi-directional communications with the DC 1000 via the I/O bus. It also communicates with the Character Assembly buffer, the Character Disassembly buffer, the Sequence Controller and the Line Identification field in the W register. All information exchange (data or control) between the MLC and the DC 1000 takes place through the Data Transfer register.

CONTROL LOGIC. This seven-bit field supervises the data transfers between the MLC and the DC 1000.

INTERRUPT AND SENSE RESPONSE LOGIC. This two-bit field provides the two interrupt requests to the DC 1000 and also presents the MLC status in response to the various Sense commands. The follow-

ing bit functions are listed in the order of their priority:

a. Data Transfer interrupt (input or output).
b. Error Interrupt.
c. CPU Priority interrupt/Ring Indicator interrupt.
d. Break interrupt.

NOTE

The Error interrupt indicates one of the following error conditions:

1) Loss of data line.
2) Loss of Clear to Send.
3) Loss of Carrier.
4) Underflow on output for synchronous transmission.

Interface Register

Field	Description	Bits
DTR	Data Transfer Register.	8
CL	Control Logic.	7
I & SR	Interrupt and Sense Register.	2
AFSD	Address Function and Sense Decoder.	8
AR	Line Address Register.	8

Figure 4-7. Interface Register Fields

DEVICE ADDRESS FUNCTION AND SENSE DECODER. This eight-bit field terminates the unidirectional address lines in the I/O bus of the

DC 1000. The five least significant bits provide the device address and the three most significant bits provide the various Function commands. The decoder determines the type of device and how the Function and Sense commands are to be handled with regard to the device.

LINE ADDRESS REGISTER. This eight-bit field communicates with the data lines in the I/O bus of the DC 1000. The six least significant bits of this register provide the address of one of the 64 data communications lines attached to the MLC. The two most significant bits may have different definitions depending on the transfers to or from the DC 1000 and the MLC. The functions of these bits are defined later in this section.

MISCELLANEOUS LOGIC.

SYNC DETECT LOGIC. For synchronous data transmission, the controller detects two consecutive sync characters in hardware before going into a data mode on input. The DC 1000 is interrupted to pick up the characters (sync or non-sync) only after the synchronization was established (received two consecutive sync characters). An option has been provided to use any sync character code under software control. During line initialization, the sync character code will be loaded in the Character Assembly buffer and is destroyed after the sync is established in hardware.

PARITY LOGIC. This section checks the parity of the incoming data and generates parity for outgoing data. If there is a parity error, the character is still transferred to the Interface register but the parity error flag is turned on.

NOTE

Under program control, even or odd parity can be generated or checked for each character but the parity must remain the same for both input and output characters.

TRANSMISSION TYPES AND SPEEDS.

SYSTEM SPECIFICATIONS.

The MLC handles both synchronous and asynchronous data transmissions. In the asynchronous mode, software control permits the character formats shown in figure 4-8.

In the synchronous mode, the message should be preceded by "n" synchronous characters, where the value of "n" is determined by the user software. Normally, the number of synchronous characters should exceed three.

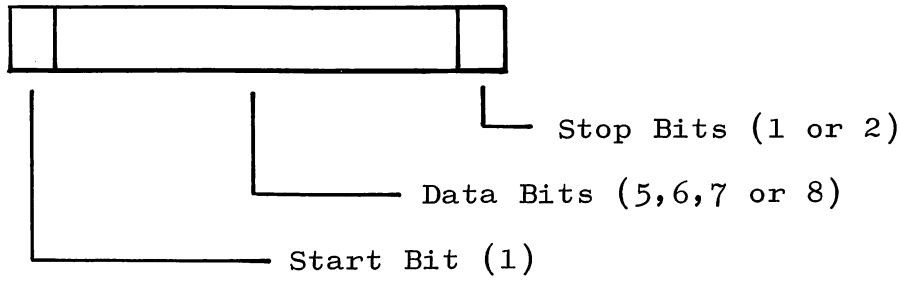


Figure 4-8. Asynchronous Character Formats

Table 4-5 lists the number of data communications lines that can be attached to the MLC, their speeds, mode of transmission, and the corresponding data sets that may be used to configure a system.

Table 4-5

Line and Modem Specifications

Line Speed (Bits per Second)	Mode	Number of Lines		Type of Modem
		Half Duplex	Full Duplex	
45	Asynchronous	64	64	Bell 103
75	Asynchronous	64	64	Bell 103

Table 4-5 (cont)

Line and Modem Specifications

Line Speed (Bits per Second)	Mode	Number of Lines		Type of Modem
		Half Duplex	Full Duplex	
110	Asynchronous	64	64	Bell 103
134.89	Asynchronous	64	64	Bell 103
150	Asynchronous	64	64	Bell 103
300	Asynchronous	64	64	Bell 103
600	Asynchronous	64	32	Bell 202
1200	Asynchronous	32	16	Bell 202
2000	Synchronous	16	8	Bell 201
2400	Synchronous	16	8	Bell 201
4800	Synchronous	8	4	Any data set with standard EIA interface capable of 4800 Baud.

The system can handle four different asynchronous line speeds selected in any combination from table 4-5 in addition to the synchronous speeds that are listed in this table. The clock frequencies are a hardware assignment but the frequency required to service a specific line is selected by the software.

Modems can handle asynchronous line speeds of 45 to 1200 Baud and synchronous speeds of 2000 to 4800 Baud. Direct connection allows asynchronous line speeds of 110 to 9600 Baud.

Under software control, any line (or lines) can be assigned high priority. This enables that line (or lines) to get through the interface to the CPU more frequently than the lines with lower priorities.

The MLC transmits all marks or spaces for one character time (no start/stop bits allowed) under software control.

When a data communications line is calling in, the MLC provides a ring indicator interrupt to the DC 1000.

The MLC selects the order of input/output (I/O). Under software control each character is transferred bit-serially over the transmission line.

The MLC gives an interrupt to the DC 1000 for the CPU priority if the DC 1000 needs to gain access to the Interface Register at a time when it is busy.

The MLC provides data communications facilities on the DC 1000 computer system and uses one of the ports of the I/O bus.

DATA COMMUNICATIONS INTERFACE.
The Multi-Line Control is equipped to transfer bit-serial data and control signals with modems meeting the EIA RS-232-B and CCITT standards. This includes, but is not limited to, the following modems: (Data sets equivalent to those listed may also be used).

Table 4-6

Telephone Company Private Line Modems

Type	Conditioning	Description
1005		75 Baud; used for 100 WPM 5-level teletypes.
1006		150 Baud; used for 100 WPM 8-level teletypes.
3002		Unconditioned voice grade data channel.
3002	C1	Conditioned voice grade data channel.
3002	C2	Conditioned voice grade data channel.
3002	C4	Conditioned voice grade data channel.

The conditioning (C1, C2, C4) refers to the quality of the data transmission channel. Some data sets require a higher quality channel because of speed, method of operation, etc. Conditioning is charged for in addition to the basic line charge.

NOTE

These conditioning requirements are clearly specified by the data set manufacturer when required.

Table 4-7

Telephone Company Switched Line Modems

Type	Description
WE 103A	Used with TWX.
WE 811B	Used with TWX.
WE 103A	Used with Dataphone (Normal DDD Network).
WE 103E	Used with Dataphone (Normal DDD Network).
WE 202C	Used with Dataphone (Normal DDD Network).
WE 201A3	Used with Dataphone (Normal DDD Network).
WE 403E3	Used with Dataphone (Normal DDD Network).
WE 403D5	Used with Dataphone (Normal DDD Network).

Table 4-8

Western Union Private Line Modems

Type	Description
WE 103F	180 BPS, Class D Channel.
WE 202D	2400 BPS, Class E Channel or Class F Channel (2 point only).
WE 201A3	2400 BPS, Class E Channel or Class F Channel (2 point only).
WE 201B1	2400 BPS, Class E Channel or Class F Channel (2 point only).

NOTE

There are no Western Union Switched Line Modems.

Table 4-9

International Leased Line Modems

Type	Description
WE 202D	1200 BPS for Telephonic use.
B.P.O. Datel 1C5	1200 BPS for Telephonic use.
S.E.L. Type GH-2011 Model #5	1200 BPS for Telephonic use.

SOFTWARE INTRODUCTION.

The DC 1000 multi-line routines provide an interface between the Executive programs and the multi-line control. These routines effect the transfer of control information, message data, and error information.

There are three types of multi-line control (MLC) routines. These include the initialization routines, the actual I/O service routines and the user provided Executive program.

In addition to the functions mentioned under the various routines, the Executive program is responsible for insuring proper line discipline. This includes detection or transmission of any acknowledgements and message formatting to include control characters (i.e., SOH, STX, ETX). Polling and selecting functions must also be performed by the Executive program. Character parity is handled by the MLC hardware on input and output. Message parity is handled by the service routine (only on output), with the exception of control characters which must be included in the Executive routine according to the specific line discipline.

The above procedures are necessary for all external communications including those to the central system.

MLC INSTRUCTION SET.

The instructions which can be accepted by the MLC are listed in table 4-10. In general, a sequence of instructions is required to implement a transaction between the MLC and the DC 1000. (Exceptions to this are: System-Reset, Enable-Interrupts, Disable-Interrupts instructions). The response of the MLC to a sequence of instructions is conditioned by a hardware sequence counter. Thus, in designing any sequence of instructions, two factors must be taken into account:

a. The affect of the sequence count on the response to the instruction.

b. The affect of the instruction on the status of the sequence count.

Figure 4-9 illustrates transaction sequences. The first three sequences illustrated are sequences initiated by interrupt requests from the MLC to the DC 1000. In these sequences, the sequence count is advanced to one at the time that the interrupt request is transmitted to the DC 1000.

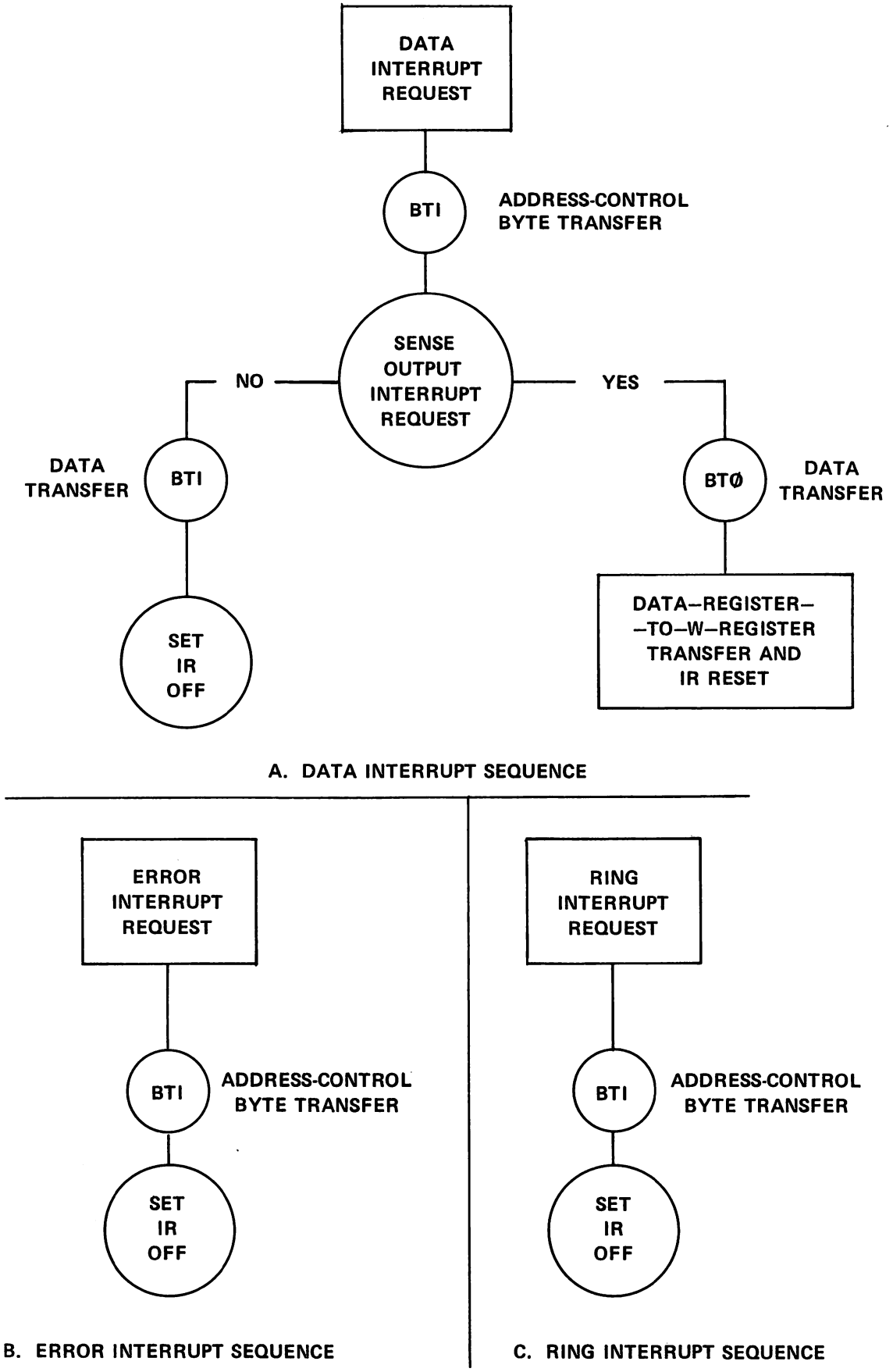


Figure 4-9. MLC/DC 1000 Transaction Sequences

The remaining three sequences shown in figure 4-10 are initiated by the DC 1000. Notice that each of these sequences begins with a Sense-IR-Off instruction. This instruction determines if the Interface register is available to process the transaction. In the event that it is available, the instruction places it in the busy status and advances the sequence count to one.

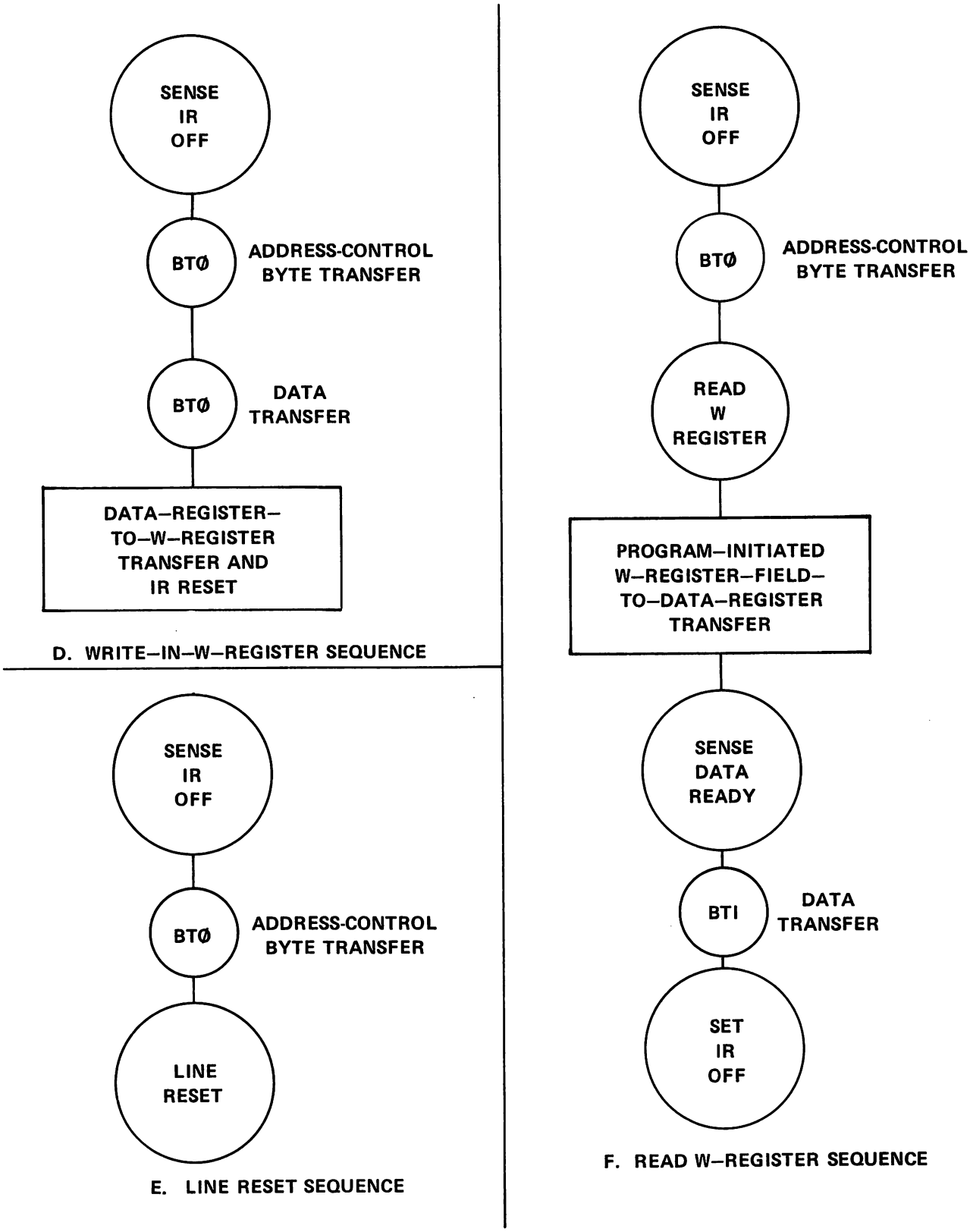


Figure 4-10. MLC/DC 1000 Transaction Sequences

Table 4-10

MLC Instruction Set

Instruction	Type	Hexadecimal Function/Address Code	Response
Enable Interrupts	FUN	0F	Interrupt Enable flip-flop in MLC Interface register sets.
Disable Interrupts	FUN	2F	Interrupt Enable flip-flop in MLC Interface register resets.
Set IRB Off	FUN	4F	Interface register is placed in the not-busy status. This must be the last instruction of any sequence in which the final byte transfer is an input transfer to the DC 1000. Where the final byte transfer is directed to a Line Control Word, the MLC makes the Interface register not busy when the transfer has been completed.
Read W Register	FUN	6F	This instruction is used after an address byte has been transferred to the Address register. It causes the data from the addressed Line Control Word to be transferred to

Table 4-10 (cont)

MLC Instruction Set

Instruction	Type	Hexadecimal Function/Address Code	Response
Read W Register (cont)			the Data register. The sequence count must be two when this instruction is received. This instruction causes the sequence count to advance to four.
System Reset	FUN	8F	This instruction resets all Line Control Words, initializes all line terminators and resets the Interface register to the not-busy status.
Line Reset	FUN	AF	After an address byte has been transferred to the Address register, this instruction is used to reset the Line Control Word of the addressed line. The sequence count must be two when this instruction is received. The Interface register is set to not busy when the instruction has been executed.
CPU Priority	FUN	CF	This instruction conditions the Interface register so that the DC 1000 SEN 4F

Table 4-10 (cont)

MLC Instruction Set

Instruction	Type	Hexadecimal Function/Address Code	Response
CPU Priority (cont)			instruction can override a high priority line lock-out and gain access to the Interface register if it is not busy. The override is reset at the time that access is achieved.
Sense Output Interrupt	SEN	2F	Used after a data interrupt is received from the MLC to determine whether the MLC is requesting a character (output interrupt true) or is requesting to input a character (output interrupt false).
Sense IR Off	SEN	4F	This must be the first instruction in most transactions initiated by the DC 1000. Exceptions are: System-Reset, CPU Priority and set/reset Enable-Interrupt transactions. The sense response indicates whether the Interface register

Table 4-10 (cont)

MLC Instruction Set

Instruction	Type	Hexadecimal Function/Address Code	Response
Sense IR Off (cont)			is available to process the transaction. If it is available, the instruction places it in the busy status and advances the sequence count to one.
Sense Data Ready	SEN	6F	Used after a Read W Register instruction to sense that a data character has been transferred from the addressed field to the Data register.
Sense Break Interrupt	SEN	8F	Used to sense a line break condition.
Normal Byte-Transfer In	BTI	OF	The response depends upon the sequence count at the time that the instruction is received. For a sequence count of one, the instruction transfers the contents of the Address register to the DC 1000 and advances the sequence

Table 4-10 (cont)

MLC Instruction Set

Instruction	Type	Hexadecimal Function/Address Code	Response
Normal Byte-Transfer-In (cont)			count to two. For a sequence count of two, the instruction transfers the contents of the Data register to the DC 1000 and does not affect the sequence count.
IBM Byte-Transfer-In	BTI	4F	Same as BTI OF except the order of the bits is reversed so that the MSB of the Data register is supplied to the LSB of the DC 1000. This compensates for the order of bits in characters received MSB first from the line.
Normal Byte-Transfer Out	BTO	OF	If the sequence count is one when the instruction is received, this instruction transfers a byte from the DC 1000 into the Address register. If the sequence count is two, it transfers the byte into the Data register. In either case, the sequence count is increased by one.

Table 4-10 (cont)

MLC Instruction Set

Instruction	Type	Hexadecimal Function/Address Code	Response
Special Byte-Transfer Out (Echo)	BTO	2F	Same as the BTO OF but also has the Echo bit set. This places the addressed line in the Echo mode of operation in which a data character is requested from the DC 1000 only after a character is received from the line.
Special Byte-Transfer-Out (IBM)	BTO	4F	Same as BTO OF except that a byte is loaded into the Data register in reverse order so that the MSB from the bus is loaded into the LSB of the Data register. This is used when a character is to be transmitted to a line with the MSB first.
Special Byte-Transfer-Out (Stop)	BTO	6F	Same as BTO OF and in addition the STOP bit is set. This places the addressed line in the stop condition such that it transmits a Mark and

Table 4-10 (cont)

MLC Instruction Set

Instruction	Type	Hexadecimal Function/Address Code	Response
Special Byte-Transfer Out (Stop) (cont)			does not request data character. The stop status is reset when the DC 1000 uses a normal byte-transfer-out instruction to transfer another data character addressed to the Character Disassembly buffer.
Special Byte-Transfer-Out (Space)	BTO	8F	Same as BTO OF but also has the Space bit set. This causes a space (all zeros) character to be transmitted to the line. The instruction is applicable only to asynchronous transmission.
Special Byte-Transfer-Out (Mark)	BTO	AF	Same as BTO OF but also has the Mark bit set. This causes a mark (all ones) character to be transmitted to the line. The instruction is applicable only to asynchronous transmission.

BYTE FORMATS.

Two classes of bytes can be defined: address-control bytes and information bytes. The address-control byte is always the first byte that is transferred between the MLC and the DC 1000 in any transfer transaction. This byte resides in the Address register portion of the Interface register. An information byte is always the second byte transferred between the MLC and the DC 1000 during a transfer transaction. This byte resides in the Data Transfer register portion of the Interface register.

ADDRESS-CONTROL BYTES. The six least significant bits of an address-control byte always specify a line address. The two remaining bits are control bits that are defined by the type of transfer transaction being executed (figure 4-11).

A transaction can be initiated by the DC 1000 to read data from or write data into one of the four addressable fields of Line Control Word. In this case the CD bits specify the field address as follows:

CD = 00 Character Disassembly buffer.

01 Sequence controller.

10 Character Assembly buffer.

11 Line Identification field.

Each data character for a line is transferred to the MLC in response to an Output Data Interrupt request. In this case, the address-control byte must contain CD = 00 to transfer the data character to the CDB field of the Line Control Word being addressed.

Each data character received from a line is transferred from the MLC to the DC 1000 in response to an Input Data Interrupt request. In this case, the C and D bits of the address-control byte specify error conditions as follows:

C = 1 Overflow error.

D = 1 Parity error.

An address control byte is transferred from the MLC to the DC 1000 in response to an Error Interrupt request. In this case, the CD bits specify the particular error condition as follows:

CD = 00 Underflow on Output.

01 Loss of Carrier.

10 Loss of Clear to Send (output).

11 Loss of Interlock.

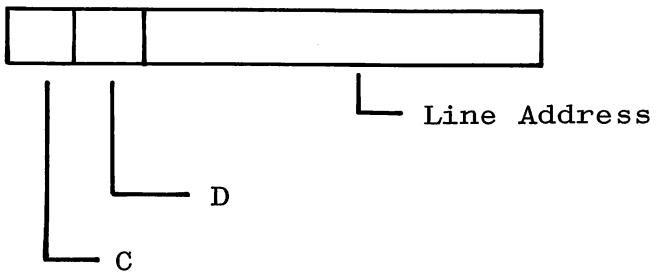


Figure 4-11. Address-Control Byte Format

INFORMATION BYTES. An information byte can contain a data character (figure 4-12) being transferred either to or from a line. The asynchronous transmission mode consists of from five to eight data bits, right-adjusted, with zeros to the left of the most significant data bit. An information byte can also contain a Sequence Control field (figure 4-13) being transferred between a Line Control Word and the DC 1000.

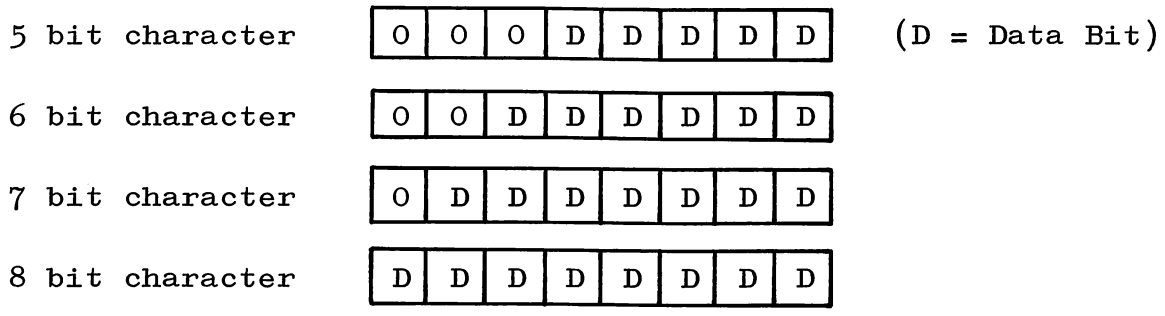


Figure 4-12. Information Byte Format

The sequence control byte format is illustrated in figure 4-13.

Figure 4-13. Sequence Control Byte Format

The bits in the sequence control byte are defined in table 4-11.

Table 4-11

Sequence Control Byte Functions

Bit Position	Status	Function
P	1	Output parity bit required/Ring Interrupt enabled.
	0	Output parity bit not required/Ring Interrupt inhibited.
I	1	Request to input a data character.
	0	No input data character request.
O	1	Request for a character from the DC 1000.
	0	No request for a character from the DC 1000.
T	1	Transmit enabled.
	0	Transmit disabled.
R	1	Receive enabled.
	0	Receive disabled.

The conditions required to set and reset each bit in the sequence controller are listed in table 4-12.

Table 4-12

Bit Control of the Sequence Controller

Bit Position	Set	Reset
P	By the DC 1000	By the DC 1000.
I *	When a data character for the DC 1000 is loaded into CAB.	When the data character is transferred from CAB into the Data register.
O *	When a data character is unloaded from the CDB into the Character Disassembly register.	When a data character is loaded from the Data register into CDB.
T	By the DC 1000	Either by the DC 1000 or in response to the sensing of certain error conditions: Loss of Interlock Loss of Clear-to-Send.
R	By the DC 1000	Either by the DC 1000 or in response to the sensing of certain error conditions: Loss of Interlock Loss of Carrier-On.

* The DC 1000 can read the I and O bits but cannot modify them.

The Line Identification field format is defined in figure 4-14.

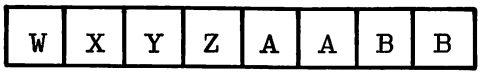


Figure 4-14. Line Identification Field Format

The Line Identification field bit definitions are found in table 4-13.

Table 4-13

Line Identification Field Bit Functions

Bit Position	Status	Function
W	0	Synchronous Transmission.
	1	Asynchronous Transmission.
X	0	One stop bit.
	1	Two stop bits.
Y	0	Normal operation.
	1	High priority.
Z	0	Even receive parity.
	1	Odd receive parity.
AA	00	Five data bits in the character.
	01	Six data bits in the character.
	10	Seven data bits in the character.
	11	Eight data bits in the character.
BB	00	Select device speed one.
	01	Select device speed two.
	10	Select device speed three.
	11	Select device speed four.

The status of the Line Identification field can be read or monitored by the DC 1000 and is not subject to modification initiated by the MLC.

The system can handle four different asynchronous line speeds. The device speeds are determined by hardware assignment configuration but the speed selected to service a specific line is a software function as determined by the BB bit status.

The MLC has the facility, under software control, to define the priority of the lines. A line having a one in bit five of the Line Identification field will have first priority to access the Interface register over the lines that have bit five set to zero. If the IR is busy and a high priority line comes up for service, logic is set in the MLC so that when the IR is next available that priority line is serviced first. In effect, there is a priority established within the priority lines to the extent that the first priority line in queue pre-empts all other lines, including priority lines, when the IR becomes available irrespective of its sequence in the IC memory scan.

SOFTWARE DESCRIPTION.

The multi-line control software package provides the routines in the interruptable environment for effecting information transfer between the MLC and the DC 1000 memory via the MLC Interface register (IR). Information transfers are essentially under the control of an interrupt which results in either byte transfer in (BTI) or byte transfer out (BTO) for input and output respectively. The information bytes are stored in the line buffers and return is then made to the Executive program. No message processing is attempted since the defined software functions, with regard to the MLC operation, relate basically to clearing the input information byte from the IR as soon as practical upon occurrence of an input interrupt condition and presenting the output information byte to the IR as soon as practical upon occurrence of an output interrupt condition, all to maximize throughput. The entire functions of character assembly and disassembly are performed by the MLC.

The appropriate linkages are incorporated into the package, however, to allow the Executive routine to accommodate the various processing

routines that may be required.

Facility is available for bi-directional communication of the MLC system control information between the MLC W register and the DC 1000. This allows reading and writing of the MLC Line Control Words, to facilitate system operational functions and MLC diagnostic requirements. The necessary subroutines can be used either by the Executive program according to defined line control procedures or alternatively via the console teletype.

NOTE

The W register is the 85-bit working register in the MLC, essentially an accumulator, which is the conduit for information moving between the remote terminals, the IC memory and the Interface register.

Each Line Control Word is stored in the MLC Integrated circuit (IC) memory. This word, containing 85 bits is used as: storage for both input and output characters, Character Assembly and Character Disassembly registers and other control information including a Line Identification field.

The separate fields of the Line Control Word are illustrated in figure 4-15.

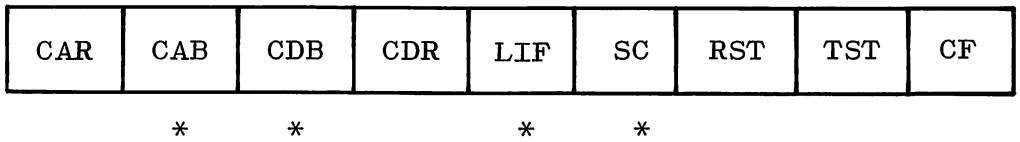


Figure 4-15. Line Control Word Fields

* These four fields are discussed in the following paragraphs. The remaining fields are controlled by hardware functions and are described in the Data Communications Hardware portion of this section.

INITIALIZATION ROUTINES.

The Initialization, or Setup routine is used to initialize the Line Control Words (LCW) in the multi-line control (MLC) integrated circuit (IC) memory. It is the responsibility of the Executive program to pass to the Setup routine the desired parameters as defined in tables 4-11 and 4-13. These parameters are then written into the line control portion of the MLC memory. There are four fields of the 85-bit Line Control Word which must be initialized. These fields are:

a. Line Identification field (LIF).
b. Sequence Controller (SC).
c. Character Assembly buffer (CAB).
d. Character Disassembly buffer (CDB).

Appropriate control information for a line is written in the Line Control Word associated with that line. Once this has been accomplished, the MLC effects transfers between that line and the DC 1000 automatically, under control of the program that is stored in the DC 1000.

CHARACTER ASSEMBLY BUFFER. Bits [18:8] of the Line Control Word. A character is held in CAB during the time that a line is requesting input to the DC 1000. For a synchronous line, the Sync character used by that line must be loaded into CAB to achieve synchronization with the incoming message.

CHARACTER DISASSEMBLY BUFFER. Bits [26:8] of the Line Control Word. Outgoing characters from the DC 1000 are held in CDB before passing to a Character Disassembly register where they are shifted out to the line. Except following a line or system reset, the CDB for a line contains the last character received for the line from the DC 1000.

SEQUENCE CONTROLLER. Bits [51:4] and [X3] in the Line Control Word. All five bits can be read but only three bits can be written. The

three bits that can be written are:

a. Receive bit [48].
b. Transmit bit [49].
c. Enable Ring Indicator/Generate Output Parity bit [X3].

When the Receive bit is set, the line is enabled to receive incoming characters and transfer them to the DC 1000. When the Transmit bit is set, the requesting of characters for the line and the transmission of characters to the line is enabled. The Enable Ring Indicator/Generate Output Parity bit has two separate functions depending on the transmit bit [49]. If bit [49] is off, indicating that the line is not transmitting, bit [X3] on indicates that the line is conditioned to detect a Ring Indicator. The second function of this bit is to indicate that the MLC will generate either even or odd parity depending on bit [43] of the Line Identification field. This second function occurs when the Transmit bit [49] is set, indicating the Transmit mode.

The two bits of the Sequence Controller that can be read only are:

a. Request-to-Input-Data bit [50].
b. Request-Data bit [51].

The Request-to-Input-Data bit is set when CAB contains a character for the DC 1000. The Request-Data bit is set when CDB is available to accept a character from the DC 1000.

LINE IDENTIFICATION FIELD. Bits [47:8] of the Line Control Word. These bits specify:

a. Transmission characteristics (synchronous or asynchronous).
b. Code characteristics (number of bits).
c. Line speed.
d. Line priority.
e. Parity (odd or even).

REQUEST INTERFACE.

This interface relates to a user Executive program request to read from or write into the MLC W register. When the user wants to write the Line Control Word, the low-order byte of the accumulator is loaded with the address-control byte, and the high-order byte is loaded with the information byte to be transferred to the designated field of the W register. A jump to the Control Information Out subroutine (WT) then performs the necessary IRB sensing and resultant information transfer as described under Control Information Transfer Out.

When the user wants to read the W register, the address-control byte is loaded into the lower-order byte of the accumulator and a jump is taken to the Control Information In subroutine (RD) which performs the necessary IRB sensing and the resultant information transfer as described under Control Information Transfer In.

Total operation for the above processes is executed in the interruptable environment because acquisition of the Interface register for the stated purposes is dependent upon line data transfer usage under the interrupt system at the time the request is made.

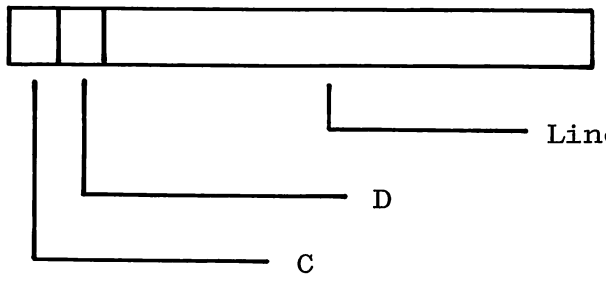
Since line data transfers have the highest priority use of the IR, the non-interruptable environment is reserved for interrupt controlled data transfers.

The routines which set up the various portions of the Line Control Word are the control information input/output (or RD and WT) routines.

CONTROL INFORMATION TRANSFER IN.

When information is to be read from a particular Line Control Word of the MLC integrated circuit memory, the DC 1000 must first gain control (through interrupt) of the Interface register. The Executive program enters the Control Information In subroutine (RD) with the address control byte in the low order eight bits of the

lower accumulator (C register). A Sense IRB Off (SEN $4F) instruction is executed. When the IR does become available and the SEN command responds True, the MLC sets IRB on. This True response causes a transfer to a BTO instruction, which transmits the line control information to IR. The code bits (CD) indicate what information is to be transferred on the next byte transfer in, thereby, selecting the W register field to be read. The control byte format is shown in figure 4-16.



CD = 00 Character Disassembly Buffer

01 Sequence Controller

10 Character Assembly Buffer

11 Line Identification

Figure 4-16. Control Byte Format (Read)

The MLC transfers the contents from the requested field of the W register to IR. The execution of a BTI command causes a data transfer from IR to the low-order byte of the lower accumulator. The execution of the appropriate FUN command (FUN $4F) sets IRB off and makes the Interface register available for line data transfers under interrupt control.

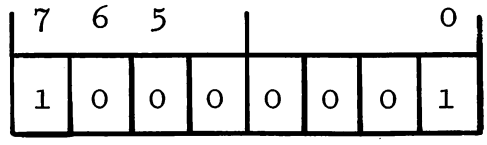
An example of a Control Information In subroutine (RD) is as follows:

```
RD      DA2     0       Read Line Info
        FUN     $CF     DC 1000 Priority
        SEN     $4F     Wait for IRB Off
        BRU     *-2     Loop
        BTO     $0F     Output Line Address
        FUN     $6F     Set for Read
        SEN     $6F     Wait for Data
        BRU     *-2     Loop
        BTI     $0F     Input Data
        FUN     $4F     Turn IRB Off
        BRUI    RD      Exit
```

The instructions necessary to enter the Control Information In subroutine (RD) are:

```
SP1          Set operand precision to one.
LOD1 =$81    Load subroutine parameters.
BRM  RD      Branch and Mark to symbolic location RD.
```

An example of the parameters that may be passed to the RD subroutine for examination of the Character Assembly buffer are shown in figure 4-17.



Bits 0-5 = Line Address
Bits 6-7 = Code bits (CD) for the Character Assembly buffer

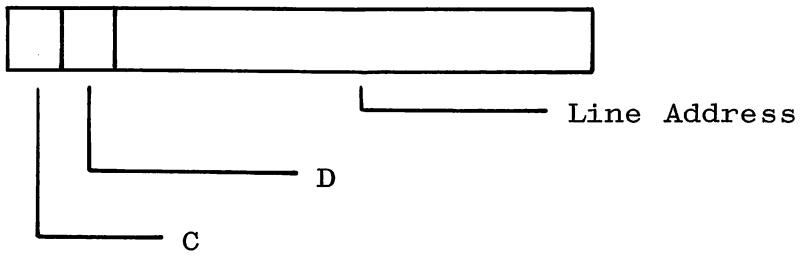
Figure 4-17. Control Byte Format of the Character Assembly Buffer

Examination of the remaining fields of the Line Control Word can be accomplished by changing code bits 6 and 7 as follows:

00 = Character Disassembly buffer.
01 = Sequence Controller.
11 = Line Identification field.

CONTROL INFORMATION TRANSFER OUT.

When information is to be written into a particular Line Control Word of the MLC integrated circuit memory, the DC 1000 must first gain control (through interrupt) of the Interface register. The Executive program enters the Control Information Out subroutine (WT) with the address-control byte in the low-order eight bits of the lower accumulator (C register) and the information byte to be transferred to the MLC in the high-order eight bits of the lower accumulator. A Sense IRB Off (SEN $4F) instruction is executed. When the IR does become available and the SEN command responds True, the MLC sets the IRB on. This True response causes a transfer to a BTO instruction, which transmits the line control information to the IR. The code bits (CD) indicate what information is to be transferred on the next BTO and which field of the W register to write. The control byte format is shown in figure 4-18.



CD = 00 Character Disassembly buffer
01 Sequence Controller
10 Character Assembly buffer
11 Line Identification

Figure 4-18. Control Byte Format (Write)

The upper eight bits of the accumulator are shifted to the low-order position and a second BTO instruction is executed to transfer a data information byte to the MLC. Detection of the second BTO execution by the MLC turns IRB off.

An example of a Control Information Out subroutine (WT) is as follows:

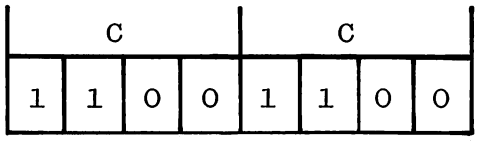
```
WT      DA2     0       Write Line Information
        FUN     $CF     DC 1000 Priority
        SEN     $4F     Wait for IRB Off
        BRU     *-2     Loop Wait
        BTO     $0F     Output Line Address
        RR8
        BTO     $0F     Output Data
        BRUI    WT      Exit.
```

The instructions necessary to enter the Control Information Out subroutine (WT) are:

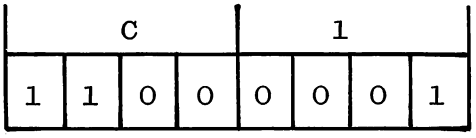
```
        SP2     Set operand precision to two
```

```
LOD2   =$CCC1   Load subroutine parameters
BRM    WT       Branch and Mark to symbolic location WT
```

An example of the parameters that may be passed to the WT subroutine for the Line Identification field are shown in figure 4-19.



Second Information Byte



First Control Byte

Control Byte:

Bits 0-5 = Line Address

Bits 6-7 = Code Bits (CD) for the Line Identification field

Information Byte:

Bits 0-1 = Device Speed One

Bits 2-3 = Eight data bits in a character

Bit 4 = Even receive parity

Bit 5 = Normal operation

Bit 6 = Two stop bits

Bit 7 = Asynchronous transmission

Figure 4-19. Line Identification Field Parameters

Additional parameters for the Line Identification field are found in table 4-13.

The parameters required to initialize the Sequence Controller are found in table 4-11.

I/O SERVICE ROUTINES.

The Service routines are responsible for actual I/O data transfer.

When the Service routine is first entered, it begins transmitting to or from the Executive buffer. Control is returned to the user Executive program and further information transfer is accomplished by entering the Service routine automatically when an interrupt occurs.

Special exits to the Executive program are provided for the cases of error recovery, control character detection and handling, and detection of end of buffer. It is the responsibility of the Executive program to determine the required action to be taken upon the detection of an error or a control character.

The Service routines operate with parameters obtained from a table called the Control Stack. There is one Control Stack for each active line. It is the responsibility of the Executive program to define the Control Stack as described in table 4-14.

CONTROL STACK.

A Control Stack is defined by the Executive program for each data line. The information contained in the Control Stack is necessary to properly perform the line driver function. Each Control Stack is 30 bytes in length and has the format illustrated in table 4-14.

The 30-byte Control Stack format is designed to provide a generally applicable software package; however, it is not intended to constrict the innovative user with special requirements. In some cases, various entries in the Control Stack can be used for other purposes. For instance, if only synchronous lines are to be used, the sync code field is not used, or for half-duplex lines, both input and output buffer addresses may not be needed.

The Control Stack must be established before data transfers can take place.

The Control Stacks for each of the 64 lines occupy contiguous locations in memory, beginning at hexadecimal location 400. If another starting location is desired, the operand of the instruction,

SA SET $400 (Hexadecimal)

may be changed.

NOTE

SA should never be set lower than $400 because there would be interference with the MLC routines.

The Control Stack starting location can be determined as follows:

Location = Line Number x 24 + SA

where the line number ranges from zero to the maximum number of lines on the system ($\leq$63).

Table 4-14
Control Stack

Byte		Contents
Hex	Dec	
0	0	Last Line Address Word.
1	1	Last Input Character.
2	2	Synchronous Code.
3	3	Control Character Mask.
4	4	Longitudinal Parity Character.
5	5	Line Identification Word.
6	6	Input Buffer Start Address.
7	7	
8	8	Next Character Position.

Table 4-14 (cont)

Control Stack

Byte		
Hex	Dec	Contents
9	9	Length of Input Buffer.
A	10	Output Buffer Start Address.
B	11	
C	12	Next Output Character.
D	13	Length of Output Buffer.
E	14	End of Input Buffer Subroutine Address.
F	15	
10	16	End of Output Buffer Subroutine Address.
11	17	
12	18	Control Character Subroutine Address.
13	19	
14	20	Input Error Subroutine Address.
15	21	
16	22	Line Error Subroutine Address.
17	23	
18	24	Ring Interrupt Subroutine Address.
19	25	
1A	26	Break Interrupt Subroutine Address.
1B	27	
1C	28	Input Function Code (bits 4-7).
1D	29	Output Function Code (bits 4-7).

INITIALIZING THE CONTROL STACK. It is the responsibility of the user to set the initial entries in the Control Stack.

The first two bytes, the Line Address word and the Last Input Character do not require initialization because these fields are filled by the Interrupt routine.

Synchronous Code.

The Synchronous Code field contains the sync character for the synchronous lines. When this field contains a sync character, the Input routine will not put sync characters into the Input buffer. When an input character is not a sync character, this field is zeroed and successive characters are processed. If sync code processing is not desired, the field should be initialized to zero. This field must be reinitialized before examining each new message.

Control Character Mask.

The Control Character Mask field is initialized to zero or to a mask, which, when logically ANDed with the input character, will yield a zero result if the character is a control character. With appropriate masks, both EBCDIC and USASCII control characters can be recognized using this method. If this field is zero, control characters will not be recognized.

The following example of a control character mask uses a hexadecimal 60:

```
0110 0000      Control Character Mask
0010 0000      Input Character
0010 0000      Logical AND result indicates no control character

0110 0000      Control Character Mask
0000 0011      Input Character
0000 0000      Logical AND result indicates a control character
```

Longitudinal Parity Character.

The longitudinal parity character should be initialized to the appropriate value. This value depends on the message format being used. The input characters are Exclusively ORed with the current LPC to form the new longitudinal parity character. Neither sync codes nor control characters are included in this computation. Hence, when

the Buffer Full routine receives control, this field contains the Exclusive OR of all data characters.

Line Identification Word.
The contents of this field are optional. However, one suggestion for the use of this field is to store the contents of the Line Identification for its line.

Input Buffer Start Address.
The Input Buffer Start Address is to be supplied by the Executive program at initialization or whenever a new buffer is required. If there is no buffer, this field should contain zero. If this field is zero and an input interrupt occurs, control will be transferred to the end of Input Buffer routine with the overflow indicator set.

Next Character Position.
The Next Character Position field is initialized to the current character position being accessed or stored. This field should be zero when starting at the beginning byte of the buffer. This field should also be zero if no input is expected.

Length of Input Buffer.
The Length of Input buffer contains the actual length in bytes of the input data plus the starting position value. For example, if the Input buffer starts at location $800 ($ denotes hexadecimal) the data starts at byte position number three, and if the length of the input data is 16 bytes and the total length of the buffer is 18 bytes, the initial values for the Buffer Address, the Next Character Position and Length of Input Buffer are shown in figure 4-20.

800	Buffer Address
01	
02	Next Character Position
12	Length of Input Buffer

Figure 4-20. Input Buffer Control Stack Entry

The maximum number of characters that can be stored without updating the count fields is 256. In this case, the Next Character Position and the End of Buffer fields are initialized to zero.

Output Buffer Start Address.
This field must be initialized to the Buffer Address or zero. If the field is zero, it is assumed that no output is provided. If an output interrupt occurs and this field is zero, the Overflow indicator is set and control is transferred to the End of Output Buffer routine.

Next Output Character, End of Output Buffer Fields.
These fields are initialized in the same manner as their input counterparts.

The remaining fields are all routine addresses. These fields must contain the address of the appropriate routine.

The I/O function codes in bytes 28 and 29 are used in the I/O interrupt routines. Bits 4-7 of these bytes are inserted in the function fields (bits 4-7) of the BTI and BTO commands, so that any of the I/O commands listed in Table 4-10 may be executed under interrupt control.

EXIT ROUTINES. All of the Exit routines are entered with the following conditions:

a. P (accumulator precision) = 2.

b. A register contains the Line Address word in the upper eight bits and the last input character, or zero if operation is not input, in the lower eight bits.

c. X register contains the address of the Control Stack for the line.

d. C register contains the address of the Exit routine.

e. O (overflow indicator) contains a flag applicable to the routine entered.

The End Of Input Buffer Routine.
This routine is entered when the input buffer is filled (when the next character position counter is equal to the length of the input buffer) or the computed character address is zero, (when the start of Input Buffer address and the next character position are zero). If the routine is entered for the latter of the above two cases, the Overflow indicator is set and the routine receiving control must set IRB off (a FUN command).

End of Output Buffer Routine.
This routine is entered when the output buffer is empty or the computed character address is zero. In the case of Echo mode, the character address is computed using the input buffer address, hence if no input is specified and Echo mode is selected, the routine will receive control. In the latter two cases, the routine is entered with the Overflow indicator set and must supply a character or turn off IRB.

Control Character Routine.
This routine is entered when the control character mask has been applied to the input character and the result is zero. This routine must set IRB off.

Input Error Routine.
This routine is entered when either the A or B bit is on. These bits are discussed under Data Transfer In. The Overflow indicator is set when the A bit (Overflow) is one.

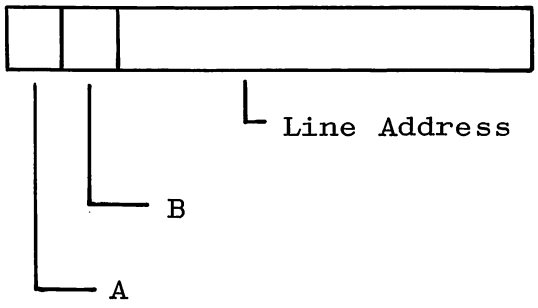
Line Error Routine.
This routine is entered whenever there is an Error interrupt. The A and B bits are stored in the Control Stack and are also in the A register. These bits are discussed under Data Transfer In. The overflow switch is reset. This routine is also entered when a Break

Interrupt occurs but the input buffer is not zero; in this case, the overflow switch is set.

Ring Interrupt Routine.
This routine is entered whenever there is a ring interrupt.

Break Interrupt Routine.
This routine is entered whenever there is a line break interrupt and the input buffer is zero.

DATA TRANSFER IN.
The incoming bit stream on a given line is assembled by the MLC in the Character Assembly register, and when assembled, it is transferred to the Character Assembly buffer of the Line Control Word. All operations on the Line Control Word are performed in the MLC working (W) register. At input data transfer time for that line, the MLC sets the Interface Register Busy bit (IRB) on, and transfers the line control information to the Interface register. The format for this byte is shown in figure 4-21.



AB = 00 Data, good parity

01 Data, bad parity

10 Overflow, good parity

11 Overflow, bad parity

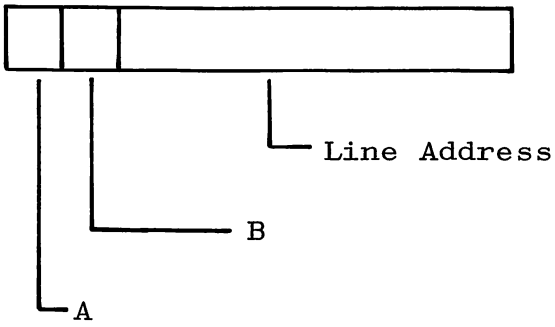
Figure 4-21. Control Byte Format (Data Transfer In)

An input interrupt request is made to the DC 1000. A program jump is made, through the I/O interrupt trap location, from the Executive program to the I/O service routine (IN). The interrupt system is enabled and the line control byte is transferred to the current accumulator. The AB code bits are saved and the line control stack entry address is computed. The Sense Input Interrupt command (SEN $2F) is executed. On a True response, program flow continues into the Input Service routine (IP). A False response results in a jump to the Output Service routine (OT). The input routine continues with a BTI to retrieve the information character byte from the Interface register. The character is saved in memory and the previously saved error bits (AB) are tested. If A (bit seven) is a one, a program jump is made to the Executive Input Overflow routine (IE). If B (bit six) is a one, a program jump is made to the Executive Parity Error routine (IF). If both A and B are zero, the character is retrieved again and tested for control character characteristics. If it is a control character, a program jump is made to the Control Character subroutine (CC). Otherwise, the longitudinal parity is computed and restored to the Control Stack. The input buffer character pointer is picked up, the character is stored in the buffer, and the pointer is incremented and tested for the end of buffer condition. Dependent upon the result of the EOB test, exit is made from the service routine to the End of Input Buffer routine or the normal line exit. The IR Busy bit is turned off by the execution of a FUN $4F command to make the Interface register available for further data transfers and return is made to the Executive program.

DATA TRANSFER OUT.

The outgoing bit stream on a given line is disassembled by the MLC in the Character Disassembly register of the Line Control Word.

At output data transfer time for a given line, the MLC sets the Interface Register Busy bit (IRB) on and transfers the line control information to the Interface register. The format of this control byte is shown in figure 4-22.



AB = 00

Figure 4-22. Control Byte Format
(Data Transfer Out)

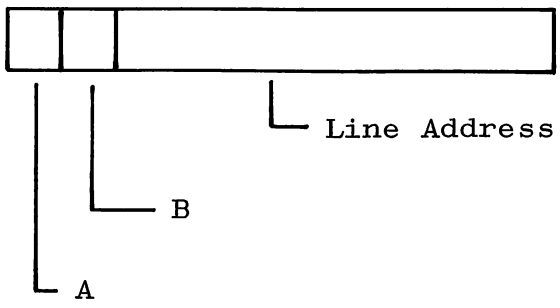
An output interrupt request is made to the DC 1000. A program jump is made, through the I/O interrupt trap location, from the Executive program to the I/O service routine. The False response on execution of the Sense Input Interrupt (SEN $2F) causes a transfer to the Output Service routine (OT). The service routine executes a BTI to transfer the line control information byte into the least significant eight bits of the current accumulator. The line buffer entry address is computed to retrieve the output character. The character is transferred to the IR by execution of a BTO instruction. Then the output buffer character pointer is incremented, the updated value is stored in the Control Stack and a check is performed for the end of buffer condition. Dependent upon the result of the EOB test, exit is made from the service routine to the End of Output Buffer routine or the normal line exit. In either event, the exit is made via the respective jump addresses in the line Control Stack.

The Data Transfer Out routine will compute and attach a parity bit to each output character. The nature of the parity (odd or even) is determined by the Line Identification word.

NON-DATA INTERRUPTS.

LINE ERROR INTERRUPT SERVICE. When the MLC detects a system error condition, a line control byte is synthesized and transferred to IR.

Then an error interrupt occurs causing a jump from the Executive program flow to the Error Interrupt Service routine. The line control byte is transferred from IR to the DC 1000 accumulator. The line control byte contains an error code and the line address in the format shown in figure 4-23.



AB =	00	Underflow (output) or Timeout (input)
	01	Loss of Carrier
	10	Loss of Clear to Send (Output)
	11	Loss of Interlock (Input or Output)

Figure 4-23. Control Byte Format (Error Conditions)

The line entry address to the Control Stack is computed to locate the Error Analysis routine address. The IRB is set to the Off state by execution of the appropriate FUN command (FUN $4F), the overflow switch is reset in the DC 1000, and a jump is made to an Error Analysis routine in the Executive program for a determination of the type of error and the resulting execution of the appropriate recovery or continuation procedure. While the nature of the Error Analysis routine is essentially a function of the user program requirements, the appropriate Control Stack linkages are included for the Error Interrupt Service routine to facilitate the incorporation of a user Error Analysis routine.

RING INTERRUPT SERVICE. When the MLC detects a ring (request to use a line), a line control byte is synthesized and transferred to the

IR. When an interrupt occurs, causing transfer of control to jump from the Executive program flow to the Interrupt Service routine, the line control byte is transferred from the IR to the DC 1000 accumulator and is used to compute the Stack address corresponding to the line. The program exits to the Ring exit address in the Control Stack.

LINE BREAK INTERRUPT SERVICE. When the MLC detects a Line Break, a line control byte is synthesized and transferred to the IR, together with the contents of the corresponding Input buffer. Then an interrupt occurs, causing transfer of control to the Interrupt Service routine. The line control byte is transferred to the DC 1000 accumulator and is used to compute the Stack address corresponding to the line. The Input buffer byte is checked for zero, which indicates a valid Line Break. If it is not zero, the Line Error exit from the Control Stack is taken with the DC 1000 overflow switch set to identify it. Otherwise, the program exits to the Break Exit address in the Control Stack.

CPU PRIORITY INTERRUPT SERVICE. After the DC 1000 has executed the FUN command requesting CPU priority, the MLC will interrupt as soon as the Interface register is not busy, transferring control to the Interrupt Service routine. This routine senses IR not busy (SEN), upon which the MLC sets the IRB. The routine then exits to the Executive program.

SAMPLE EXECUTIVE PROGRAM.

A sample Executive program will be provided that concentrates and forwards information from the TC 500's, B 9352's and B 9351's to the B 3500. This program illustrates techniques used with the DC 1200 that will be helpful in structurizing a customized concentration program by the user. Such techniques as polling, selecting, and Block Check Character accumulation will be illustrated.

APPENDIX A

STANDARD CHARACTER CODES

Character	USASCII Binary	USASCII Hexadecimal	Hollerith
@	11000000	C0	0-2-8
A	11000001	C1	12-1
B	11000010	C2	12-2
C	11000011	C3	12-3
D	11000100	C4	12-4
E	11000101	C5	12-5
F	11000110	C6	12-6
G	11000111	C7	12-7
H	11001000	C8	12-8
I	11001001	C9	12-9
J	11001010	CA	11-1
K	11001011	CB	11-2
L	11001100	CC	11-3
M	11001101	CD	11-4
N	11001110	CE	11-5
O	11001111	CF	11-6
P	11010000	D0	11-7
Q	11010001	D1	11-8
R	11010010	D2	11-9
S	11010011	D3	0-2
T	11010100	D4	0-3
U	11010101	D5	0-4
V	11010110	D6	0-5
W	11010111	D7	0-6
X	11011000	D8	0-7
Y	11011001	D9	0-8
Z	11011010	DA	0-9
[	11011011	DB	12-5-8
\	11011100	DC	0-6-8
]	11011101	DD	11-5-8
↑	11011110	DE	7-8
←	11011111	DF	2-8

Character	USASCII Binary	USASCII Hexadecimal	Hollerith
blank	10100000	A0	no punch
!	10100001	A1	11-2-8
"	10100010	A2	0-5-8
#	10100011	A3	0-7-8
$	10100100	A4	11-3-8
%	10100101	A5	11-7-8
&	10100110	A6	12-7-8
'	10100111	A7	4-8
(	10101000	A8	0-4-8
)	10101001	A9	12-4-8
*	10101010	AA	11-4-8
+	10101011	AB	12
,	10101100	AC	0-3-8
–	10101101	AD	11
.	10101110	AE	12-3-8
/	10101111	AF	0-1
0	10110000	B0	0
1	10110001	B1	1
2	10110010	B2	2
3	10110011	B3	3
4	10110100	B4	4
5	10110101	B5	5
6	10110110	B6	6
7	10110111	B7	7
8	10111000	B8	8
9	10111001	B9	9
:	10111010	BA	5-8
;	10111011	BB	11-6-8
<	10111100	BC	12-6-8
=	10111101	BD	3-8
>	10111110	BE	6-8
?	10111111	BF	12-2-8

APPENDIX B

INSTRUCTIONS

This appendix gives the mnemonic, hexadecimal code, definition, and number of machine cycles required for each valid instruction.

Table B-1

One-Byte Non-Memory-Reference Instructions

Mnemonic	Hexa-decimal Code	Description	Machine Cycles by Operand Precision			
			1	2	3	4
HLT	00	Halt with program counter at next location.	1	1	1	1
CST	01	Change environmental status.	1	1	1	1
SOV	02	Skip on current accumulator overflow and reset overflow indicator.	1	1	1	1
SOVN	03	Skip on NO current accumulator overflow and reset overflow indicator.	1	1	1	1
SOD	04	Skip if current accumulator is odd.	1	1	1	1
SODN	05	Skip if current accumulator is NOT odd.	1	1	1	1
SAN	06	Skip if current accumulator is negative.	1	1	1	1
SANN	07	Skip if current accumulator is positive.	1	1	1	1
SAZ	08	Skip if current accumulator is zero.	1	2	2	2
SAZN	09	Skip if current accumulator is NOT zero.	1	2	2	2
SXZ	0A	Skip if current index register is zero.	2	2	2	2

Table B-1 (cont)

Mnemonic	Hexa-decimal Code	Description	Machine Cycles by Operand Precision			
			1	2	3	3
SXZN	0B	Skip if current index register is NOT zero.	2	2	2	2
CLA	0C	Clear current accumulator to the set precision.	1	2	2	2
CMA	0D	Complement current accumulator to the set precision.	1	2	2	2
IAR	0E	Increment current accumulator to the set precision.	1	2	2	2
CIA	0F	Complement and increment current accumulator to the set precision.	1	2	2	2
SP1	10	Set current operand precision to one byte.	1	1	1	1
SP2	11	Set current operand precision to two bytes.	1	1	1	1
SP3	12	Set current operand precision to three bytes.	1	1	1	1
SP4	13	Set current operand precision to four bytes.	1	1	1	1
SOF	14	Set current overflow indicator.	1	1	1	1
ROF	15	Reset current overflow indicator.	1	1	1	1
NOP	16	No operation.	1	1	1	1
IXO	17	Increment current index register by the set precision.	2	2	2	2
SR1	18	Shift current accumulator right one bit to the set precision.	1	2	2	2
SR8	19	Shift current accumulator right eight bits to the set precision.	1	2	2	2

Table B-1 (cont)

Mnemonic	Hexa-decimal Code	Description	Machine Cycles by Operand Precision			
			1	2	3	4
RR1	1A	Rotate current accumulator right one bit to the set precision.	2	2	2	3
RR8	1B	Rotate current accumulator right eight bits to the set precision.	1	2	2	2
SL1	1C	Shift current accumulator left one bit to the set precision.	1	2	2	2
SL8	1D	Shift current accumulator left eight bits to the set precision.	1	2	2	2
RL1	1E	Rotate current accumulator left one bit to the set precision.	2	2	2	3
RL8	1F	Rotate current accumulator left eight bits to the set precision.	1	2	2	2

Table B-2
Source And Destination Register Instructions

The source (s) and the destination (d) portion of the following instructions can be replaced with the corresponding mnemonic and hexadecimal codes that are listed in table B-4.

Example:

Mnemonic TZA or hexadecimal 2001 transfers zeros to the current accumulator most-significant 16 bits.

Mnemonic	Hexa-decimal Code	Definition	Machine Cycles by Operand Precision			
			1	2	3	4
Ts,d	20s,d	Transfer source register to destination register.	3	3	3	3
Cs,d	21s,d	Complement source register and transfer to destination register.	3	3	3	3
Is,d	22s,d	Increment source register and transfer to destination register.	3	3	3	3
Ds,d	23s,d	Decrement source register and transfer to destination register.	3	3	3	3
Ms,d	24s,d	*AND source register into destination register.	3	3	3	3
Os,d	25s,d	*OR source register into destination register.	3	3	3	3
Es,d	26s,d	*Exclusive OR the source register into the destination register.	3	3	3	3
As,d	27s,d	Add source register to destination register.	3	3	3	3

* See table B-3 for the results of these logical instructions.

Table B-3
Logical Results

The logical instructions in table B-2 produce the following results:

Function	Register Value		Result
	Source	Destination	
AND	0	0	0
	0	1	0
	1	0	0
	1	1	1
OR (inclusive)	0	0	0
	0	1	1
	1	0	1
	1	1	1
OR (exclusive)	0	0	0
	0	1	1
	1	0	1
	1	1	0

Table B-4
Instruction Modifiers

The allowable mnemonic and hexadecimal codes that may be used in the source and destination portion of the instructions in table B-2 are listed below:

Mnemonic	Hexa-decimal Code	Definition
s,d = Z	s,d = 0	Nonexistant zeros register.
s,d = A	s,d = 1	Current accumulator most-significant 16 bits.
s,d = C	s,d = 2	Current accumulator least-significant 16 bits.
s,d = X	s,d = 3	Current index register.
s,d = P	s,d = 4	Current program counter.

Table B-4 (cont)

Mnemonic	Hexa-decimal Code	Definition
s,d = B	s,d = 5	Noncurrent accumulator most-significant 16 bits.
s,d = D	s,d = 6	Noncurrent accumulator least-significant 16 bits.
s,d = Y	s,d = 7	Noncurrent index register.
s,d = 0	s,d = 8	Noncurrent program counter.
s,d = F	x,d = 9	Peripheral adapter input/output register.

Table B-5

Two-Byte Non-Memory-Reference Instructions

Mnemonic	Hexa-decimal Code	Definition	Machine Cycles by Operand Precision			
			1	2	3	4
MIN	3000	Modify interrupts according to current accumulator.	2	2	2	2
CIS	3100	Copy interrupt status into current accumulator.	2	2	2	2
CIM	3120	Copy interrupt mask into current accumulator.	2	2	2	2
SIE	3200	Skip if interruptable environment.	3	3	3	3
SSH	3260	Skip on power fail interrupt in halt mode.	3	3	3	3
SS1	3220	Skip if SENSE switch 1 is off.	3	3	3	3
SS2	3240	Skip if SENSE switch 2 is off.	3	3	3	3
SS3	3280	Skip if SENSE switch is off.	3	3	3	3
IBD	3300	Interrupt system disable.	2	2	2	2
IBE	3320	Interrupt system enable.	2	2	2	2

Table B-6

Three-Byte Non-Memory-Reference Instructions

Mnemonic	Hexa-decimal Code	Definition	Machine Cycles by Operand Precision			
			1	2	3	4
BRM	38----	Branch and mark (store) current program counter.	6	6	6	6
CSQ	39----	Change environmental status and load noncurrent program counter.	3	3	3	3

Table B-7

Two-Byte Memory-Reference Instructions

Mnemonic	Hexa-decimal Code	Addressing Mode	Definition	Machine Cycles by Operand Precision			
				1	2	3	4
LOD	40--	Direct	Load the current accumulator from memory.	3	4	5	6
	50--	Direct, current sector		3	4	5	6
	54--	Indirect, current sector*		5	6	7	8
	58--	Indexed		3	4	5	6
	5C--	Indirect, zero sector*		5	6	7	8
STO	60--	Direct	Store the current accumulator in memory.	3	4	5	6
	70--	Direct, current sector		3	4	5	6
	74--	Indirect, current sector*		5	6	7	8
	78--	Indexed		3	4	5	6
	7C--	Indirect, zero sector*		5	6	7	8
SUM	80--	Direct	Add memory to the current accumulator.	3	4	5	6
	90--	Direct, current sector		3	4	5	6

Table B-7 (cont)

Mnemonic	Hexadecimal Code	Addressing Mode	Definition	Machine Cycles by Operand Precision			
				1	2	3	4
SUM (cont)	94--	Indirect, current sector*		5	6	7	8
	98--	Indexed		3	4	5	6
	9C--	Indirect, zero sector*		5	6	7	8
SUB	A0--	Direct	Subtract memory from the current accumulator.	3	4	5	6
	B0--	Direct, current sector		3	4	5	6
	B4--	Indirect, current sector*		5	6	7	8
	B8--	Indexed		3	4	5	6
	BC--	Indirect, zero sector*		5	6	7	8
AND	C0--	Direct	AND	3	4	5	6
	D0--	Direct, current sector	memory to the current accumulator.	3	4	5	6
	D4--	Indirect, current sector*		5	6	7	8
	D8--	Indexed		3	4	5	6
	DC--	Indirect, zero sector*		5	6	7	8
BRU	E0--	Direct	Branch unconditionally.	2	2	2	2
	F0--	Direct, current sector		2	2	2	2
	F4--	Indirect, current sector*		4	4	4	4
	F8--	Indexed		3	3	3	3
	FC--	Indirect, zero sector*		4	4	4	4

* Two machine cycles are required for each additional level of indirect addressing.

Table B-8

Two-Byte Input/Output Instructions

Mnemonic	Hexa-decimal Code	Definition	Machine Cycles by Operand Precision			
			1	2	3	4
BTO	30--	Byte transfer out of current accumulator.	2	2	2	2
BTI	31--	Byte transfer into current accumulator.	2	2	2	2
SEN	32--	Sense device and skip if sense true.	3	3	3	3
FUN	33--	Transfer function out and transfer a byte out of the current accumulator.	2	2	2	2

Table B-9

Reserved Teletype Controller Instructions

Mnemonic	Hexa-decimal Code	Definition	Machine Cycles by Operand Precision			
			1	2	3	4
BTO 01	3001	Transfer least significant byte of current accumulator to the teletype controller input buffer.	2	2	2	2
BTI 01	3101	Transfer teletype controller buffer to the least significant byte of the current accumulator.	2	2	2	2
SEN 21	3221	Sense for teletype controller output buffer readiness and skip if ready.	3	3	3	3
SEN 41	3241	Sense for teletype controller input buffer readiness and skip if ready.	3	3	3	3
FUN 01	3301	Enable teletype controller write interrupt.	2	2	2	2

Table B-9 (cont)

Mnemonic	Hexa-decimal Code	Definition	Machine Cycles by Operand Precision			
			1	2	3	4
FUN 21	3321	Reset teletype controller.	2	2	2	2
FUN 41	3341	Start teletype punched-tape reader.	2	2	2	2
FUN 61	3361	Enable teletype controller read interrupt.	2	2	2	2
Fun C1	33C1	Read one character on the teletype punched-tape reader.	2	2	2	2

Table B-10

Reserved Punched-Tape Instructions

Mnemonic	Hexa-decimal Code	Definition	Machine Cycles by Operand Precision			
			1	2	3	4
BTO 02	3002	Punch least-significant byte of current accumulator.	2	2	2	2
BTI 02	3102	Read one tape character into least-significant byte of current accumulator.	2	2	2	2
SEN 02	3202	Sense for punched-tape transfer readiness and skip if ready.	3	3	3	3
FUN 22	3322	Start punched-tape reader.	2	2	2	2
FUN 42	3342	Enable punched-tape interrupt.	2	2	2	2
FUN A2	33A2	Stop punched-tape reader.	2	2	2	2
FUN C2	33C2	Disable punched-tape interrupt.	2	2	2	2

Table B-11

Reserved Single Line Control Instructions

Mnemonic	Hexa-decimal Code	Definition	Machine Cycles by Operand Precision			
			1	2	3	4
FUN	3306	Enable receive bit mode.	2	2	2	2
FUN	3326	Enable receive interrupts.	2	2	2	2
FUN	3346	Disable receive interrupts.	2	2	2	2
FUN	3366	Turn on data terminal ready line.	2	2	2	2
FUN	3386	Turn on request to send line.	2	2	2	2
FUN	33A6	Turn off request to send line.	2	2	2	2
FUN	33C6	Enable receive word mode.	2	2	2	2
FUN	33E6	Initialize controller.	2	2	2	2
BTO	3006	Transfer one byte from the DC-1000 to the write shift register.	2	2	2	2
BTI	3106	Transfer contents of receive shift register to the DC-1000.	2	2	2	2
SEN	3206	Sense receive bit mode.	3	3	3	3
SEN	3226	Sense if output register is ready.	3	3	3	3
SEN	3246	Sense if input register is ready.	3	3	3	3
SEN	3266	Sense carrier on.	3	3	3	3
SEN	3286	Sense clear to send.	3	3	3	3
SEN	32A6	Sense interlock.	3	3	3	3
SEN	32C6	Sense ring indicator.	3	3	3	3
SEN	32E6	Sense odd parity in receive shift register.	3	3	3	3

Table B-12

Reserved Line Printer Control Instructions

Mnemonic	Hexa-decimal Code	Definition	Machine Cycles by Operand Precision			
			1	2	3	4
FUN	331D	Enable interrupts.	2	2	2	2
FUN	333D	Disable interrupts.	2	2	2	2
FUN	335D	Initialize controller.	2	2	2	2
BTO	301D	Load data register.	2	2	2	2
BTO	303D	Load format register.	2	2	2	2
SEN	321D	Sense parity or sync error.	3	3	3	3
SEN	323D	Sense end-of-page.	3	3	3	3
SEN	325D	Sense paper motion complete.	3	3	3	3
SEN	327D	Sense print cycle complete.	3	3	3	3
SEN	32BD	Sense printer ready.	3	3	3	3
SEN	32DD	Sense data buffer ready.	3	3	3	3

Table B-13

Reserved Card Reader Control Instructions

Mnemonic	Hexa-decimal Code	Definition	Machine Cycles by Operand Precision			
			1	2	3	4
FUN	3305	Enable interrupts.	2	2	2	2
FUN	3325	Initiate card cycle.	2	2	2	2
FUN	3345	Disable interrupts.	2	2	2	2
FUN	3365	Initialize controller.	2	2	2	2
BTI	3105	Input data from card to the DC-1000.	2	2	2	2
SEN	3205	Sense for card reader on-line.	3	3	3	3

Table B-13 (cont)

Mnemonic	Hexa-decimal Code	Definition	Machine Cycles by Operand Precision			
			1	2	3	4
SEN	3225	Sense for reader malfunction.	3	3	3	3
SEN	3245	Sense for data ready.	3	3	3	3
SEN	3265	Sense for card reader ready.	3	3	3	3

Table B-14

Reserved Card Punch Control Instructions

Mnemonic	Hexa-decimal Code	Definition	Machine Cycles by Operand Precision			
			1	2	3	4
FUN	3307	Enable interrupts.	2	2	2	2
FUN	3327	Disable interrupts.	2	2	2	2
FUN	3347	Select main stacker.	2	2	2	2
FUN	3367	Select auxiliary stacker.	2	2	2	2
FUN	3387	Initiate card cycle.	2	2	2	2
FUN	33A7	Initiate controller.	2	2	2	2
BTO	3007	Output data from DC-1000 to the card punch.	2	2	2	2
SEN	3207	Sense for card punch ready.	3	3	3	3
SEN	3227	Sense for card punch busy.	3	3	3	3
SEN	3247	Sense for card punch error.	3	3	3	3
SEN	3267	Sense first four rows.	3	3	3	3
SEN	3287	Sense data buffer ready.	3	3	3	3

Table B-15

Reserved Multi-Line Control Instructions

Mnemonic	Hexa-decimal Code	Definition	Machine Cycles by Operand Precision			
			1	2	3	4
FUN	330F	Enable interrupts.	2	2	2	2
FUN	332F	Disable interrupts.	2	2	2	2
FUN	334F	Set IRB off.	2	2	2	2
FUN	336F	Read W-register.	2	2	2	2
FUN	338F	System reset.	2	2	2	2
FUN	33AF	Line reset.	2	2	2	2
FUN	33CF	CPU priority.	2	2	2	2
FUN	33EF		2	2	2	2
BTI	310F	Normal BTI.	2	2	2	2
BTO	300F	Normal BTO.	2	2	2	2
BTO	302F	Special BTO.	2	2	2	2
SEN	320F		3	3	3	3
SEN	322F	Sense output interrupt.	3	3	3	3
SEN	324F	Sense IRB off.	3	3	3	3
SEN	326F	Sense data ready.	3	3	3	3
SEN	328F		3	3	3	3
SEN	32AF		3	3	3	3
SEN	32CF		3	3	3	3
SEN	32EF		3	3	3	3

Table B-16

Assembler Pseudo Instructions

Instructions	Description
ORG	Set location counter.
BSS	Block storage specification.
LIT	Begin literal block.
IOR	Begin indirect-pointer block.
SET	Set symbol value.
EQU	Equate symbols.
MOR	Halt for more input.
END	End assembly.
DA1	Define constant, precision 1.
DA2	Define constant, precision 2.
DA3	Define constant, precision 3.
DA4	Define constant, precision 4.

TABLE B-17

Reserved Paper Tape Reader Instructions

Mnemonic	Hexadecimal Code	Definition	Machine Cycles by Operand Precision			
			1	2	3	4
FUN	3302	Start reader.	2	2	2	2
FUN	3322	Stop reader.	2	2	2	2
FUN	3342	Select binary mode of input.	2	2	2	2
FUN	3362	Select alphanumeric mode of input.	2	2	2	2
FUN	3382	Enable controller interrupts.	2	2	2	2
FUN	33A2	Disable controller interrupts.	2	2	2	2
FUN	33C2	Backspace tape.	2	2	2	2
FUN	33E2	Rewind tape (reel mode only).	2	2	2	2

TABLE B-17 (cont)

Mnemonic	Hexadecimal Code	Definition	Machine Cycles by Operand Precision			
			1	2	3	4
SEN	3202	Sense not parity error.	3	3	3	3
SEN	3222	Sense data ready.	3	3	3	3
SEN	3262	Sense not end-of-tape.	3	3	3	3
SEN	3282	Sense not end-of-message (stop code).	3	3	3	3
BTI	3102	Input 1 byte from reader to accumulator.	2	2	2	2
BTO	3002	Initialize reader.	2	2	2	2

TABLE B-18

Reserved Paper Tape Punch Instructions

Mnemonic	Hexadecimal Code	Definition	Machine Cycles by Operand Precision			
			1	2	3	4
FUN	3309	Select binary mode.	2	2	2	2
FUN	3329	Select alphanumeric mode.	2	2	2	2
FUN	3349	Enable interrupts.	2	2	2	2
FUN	3369	Disable interrupts.	2	2	2	2
FUN	3389					
FUN	33A9	Start punch.	2	2	2	2
FUN	33C9	Stop punch.	2	2	2	2
FUN	33E9	Reset controller.	2	2	2	2
SEN	3209	Sense motors not stopping.	3	3	3	3
SEN	3229	Sense not stop code.	3	3	3	3
SEN	3249	Sense not end-of-tape.	3	3	3	3
SEN	3289	Sense punch ready.	3	3	3	3
SEN	32A9	Sense data needed.	3	3	3	3
BTO	3009	Output 1 byte to punch.	2	2	2	2

APPENDIX C

Table C-1

Summary of DC 1000 Math Routine Size and Timing

Routine Symbol and Name		Storage-bytes		Time-Cycles		Average Time
		Routine	Temp	Minimum	Maximum	Milliseconds
	FIXED POINT					
MU	Multiply - 2 byte	107	4	101	154	.19
DI	Divide - 2 byte	133	10	261	274	.40
DB	Decimal to Binary	98	8	106	880	.70
BD	Binary to Decimal	154	10	154	2084	2.00
	FLOATING POINT					
RX	Load X Immediate	20	2	35	35	.05
SE	Separate Exponent	28	3	34	42	.05
FC	Construct Floating Point	28	3	36	42	.06
FN	Normalize	70	6	82	261	.26
IF	Integer to Float	54	7	74	319	.35
FI	Float to Integer	70	7	71	346	.30
FA	Add	144	8	336	634	.74
FS	Subtract	32	4	388	687	.82
FM	Multiply	128	10	909	996	1.42
FD	Divide	146	14	1125	1236	1.76
FP	Fractional Part			127	648	.60
IP	Integer Part	110	2	124	704	.60
SQ	Square Root	105	12	5090	5455	7.83
EX	Exponential e^X	198	10	9440	9950	14.54
LN	Natural Log.	173	10	——	——	15.82
FE	Exponentiation A**B	23	0	——	——	31.81
QF	Decimal to Float	163	6	——	——	18.15
FQ	Float to Decimal	138	8	——	——	22.80
SI	Sine	206	10	13250	16100	19.95
CO	Cosine	27	0	13700	16500	20.50
AT	Arctangent	203	6	——	——	27.00

NOTES

1. One cycle equals 1.5 microseconds.
2. Times include the time of all other routines during execution.

Table C-2
Math Routine Matrix

Symbol	Routine	OTHER ROUTINES REQUIRED IN MEMORY																						SIZE	
		RX	SE	FC	FN	IF	FI	FA	FS	FM	FD	FP	IP	SQ	EX	LN	FE	QF	FQ	SI	CO	AT	DB	BD	Routine	Total*
RX	Load X Immediate	X																							22	22
SE	Separate Exponent		X																						31	31
FC	Construct Floating Point			X																					31	31
FN	Normalize				X																				76	76
IF	Integer to Floating					X																			61	61
FI	Floating to Integer						X																		77	77
FA	Add	X	X		X			X																	152	281
FS	Subtract	X	X		X			X	X																36	317
FM	Multiply									X															138	138
FD	Divide										X														160	160
FP	Fractional Part	X	X	X	X							X													112	272
IP	Integer Part	X	X	X	X								X												110	280
SQ	Square Root	X	X		X			X			X			X											117	558
EX	Exponential	X	X	X	X		X	X	X	X	X	X			X										203	1038
LN	Natural Log	X	X		X	X		X	X	X	X					X									183	859
FE	Exponentiation A**B	X	X	X	X	X	X	X	X	X	X	X			X	X	X								23	1305
QF	Decimal to Floating	X	X	X	X	X	X	X	X	X	X	X			X			X					X		169	1374
FQ	Floating to Decimal	X	X	X	X	X	X	X	X	X	X	X			X				X					X	146	1409
SI	Sine	X	X	X	X			X	X	X	X	X								X					216	974
CO	Cosine	X	X	X	X			X	X	X	X	X								X	X				27	1001
AT	Arctangent	X	X		X			X	X	X	X	X										X			209	824

* Total size includes the size of the routine plus the sizes of the routines which it calls.

APPENDIX D

Paper Tape Conversion Codes

BCL PT CODE	HEXADECIMAL	USASCII	CHARACTER
P B A 8 4 2 1			
0 0 0 0 0 0 0	$00	BF	?
1 0 0 0 0 0 1	41	B1	1
1 0 0 0 0 1 0	42	B2	2
0 0 0 0 0 1 1	03	B3	3
1 0 0 0 1 0 0	44	B4	4
0 0 0 0 1 0 1	05	B5	5
0 0 0 0 1 1 0	06	B6	6
1 0 0 0 1 1 1	47	B7	7
1 0 0 1 0 0 0	48	B8	8
0 0 0 1 0 0 1	09	B9	9
0 0 0 1 0 1 0	0A	B0	0
1 0 0 1 0 1 1	4B	A3	#
0 0 0 1 1 0 0	0C	C0	@
1 0 0 1 1 0 1	4D	BA	:
1 0 0 1 1 1 0	4E	BE	>
0 0 0 1 1 1 1	0F	DE	≥
1 0 1 0 0 0 0	50	A0	∅
0 0 1 0 0 0 1	11	AF	/
0 0 1 0 0 1 0	12	D3	S
1 0 1 0 0 1 1	53	D4	T
0 0 1 0 1 0 0	14	D5	U
1 0 1 0 1 0 1	55	D6	V
1 0 1 0 1 1 0	56	D7	W
0 0 1 0 1 1 1	17	D8	X
0 0 1 1 0 0 0	18	D9	Y
1 0 1 1 0 0 1	59	DA	Z
1 0 1 1 0 1 0	5A	A7	≠
0 0 1 1 0 1 1	1B	AC	'

Paper Tape Conversion Codes (contd)

BCL PT CODE	HEXADECIMAL	USASCII	CHARACTER
P B A 8 4 2 1			
1 0 1 1 1 0 0	$5C	A5	%
0 0 1 1 1 0 1	1D	BD	=
0 0 1 1 1 1 0	1E	DD	]
1 0 1 1 1 1 1	5F	A2	"
1 1 0 0 0 0 0	60	AD	-
0 1 0 0 0 0 1	21	CA	J
0 1 0 0 0 1 0	22	CB	K
1 1 0 0 0 1 1	63	CC	L
0 1 0 0 1 0 0	24	CD	M
1 1 0 0 1 0 1	65	CE	N
1 1 0 0 1 1 0	66	CF	O
0 1 0 0 1 1 1	27	DO	P
0 1 0 1 0 0 0	28	D1	Q
1 1 0 1 0 0 1	69	D2	R
1 1 0 1 0 1 0	6A	DC	X
0 1 0 1 0 1 1	2B	A4	$
1 1 0 1 1 0 0	6C	AA	*
0 1 0 1 1 0 1	20	A9	)
0 1 0 1 1 1 0	2E	BB	;
1 1 0 1 1 1 1	6F	A1	≤
0 1 1 0 0 0 0	30	A6	&
1 1 1 0 0 0 1	71	C1	A
1 1 1 0 0 1 0	72	C2	B
0 1 1 0 0 1 1	33	C3	C
1 1 1 0 1 0 0	74	C4	D
0 1 1 0 1 0 1	35	C5	E
0 1 1 0 1 1 0	36	C6	F
1 1 1 0 1 1 1	77	C7	G
1 1 1 1 0 0 0	78	C8	H

Paper Tape Conversion Codes (contd)

BCL PT CODE							HEXADECIMAL	USASCII	CHARACTER
P	B	A	8	4	2	1			
0	1	1	1	0	0	1	$39	C9	I
0	1	1	1	0	1	0	3A	AB	+
1	1	1	1	0	1	1	7B	AE	.
0	1	1	1	1	0	0	3C	DB	[
1	1	1	1	1	0	1	7D	A8	(
1	1	1	1	1	1	0	7E	BC	<
0	1	1	1	1	1	1	3F	DR	←

INDEX

BURROUGHS CORPORATION
DATA PROCESSING PUBLICATIONS
REMARKS FORM

TITLE: _____ FORM: _____

_____ DATE: _____

_____

CHECK TYPE OF SUGGESTION:

☐ ADDITION ☐ DELETION ☐ REVISION ☐ ERROR

cut along dotted line

GENERAL COMMENTS AND/OR SUGGESTIONS FOR IMPROVEMENT OF PUBLICATION:

FROM: NAME _____ DATE _____

TITLE _____

COMPANY _____

ADDRESS _____

_____

STAPLE

FOLD DOWN SECOND FOLD DOWN

Postage Will Be Paid by Addressee

No Postage Stamp Necessary If Mailed in the United States

BUSINESS REPLY MAIL
First Class Permit No. 817, Detroit, Mich. 48232

Burroughs Corporation
6071 Second Avenue
Detroit, Michigan 48232

attn: Sales Technical Services
Systems Documentation

FOLD UP FIRST FOLD UP

1046380

5-70

Printed in U.S. America