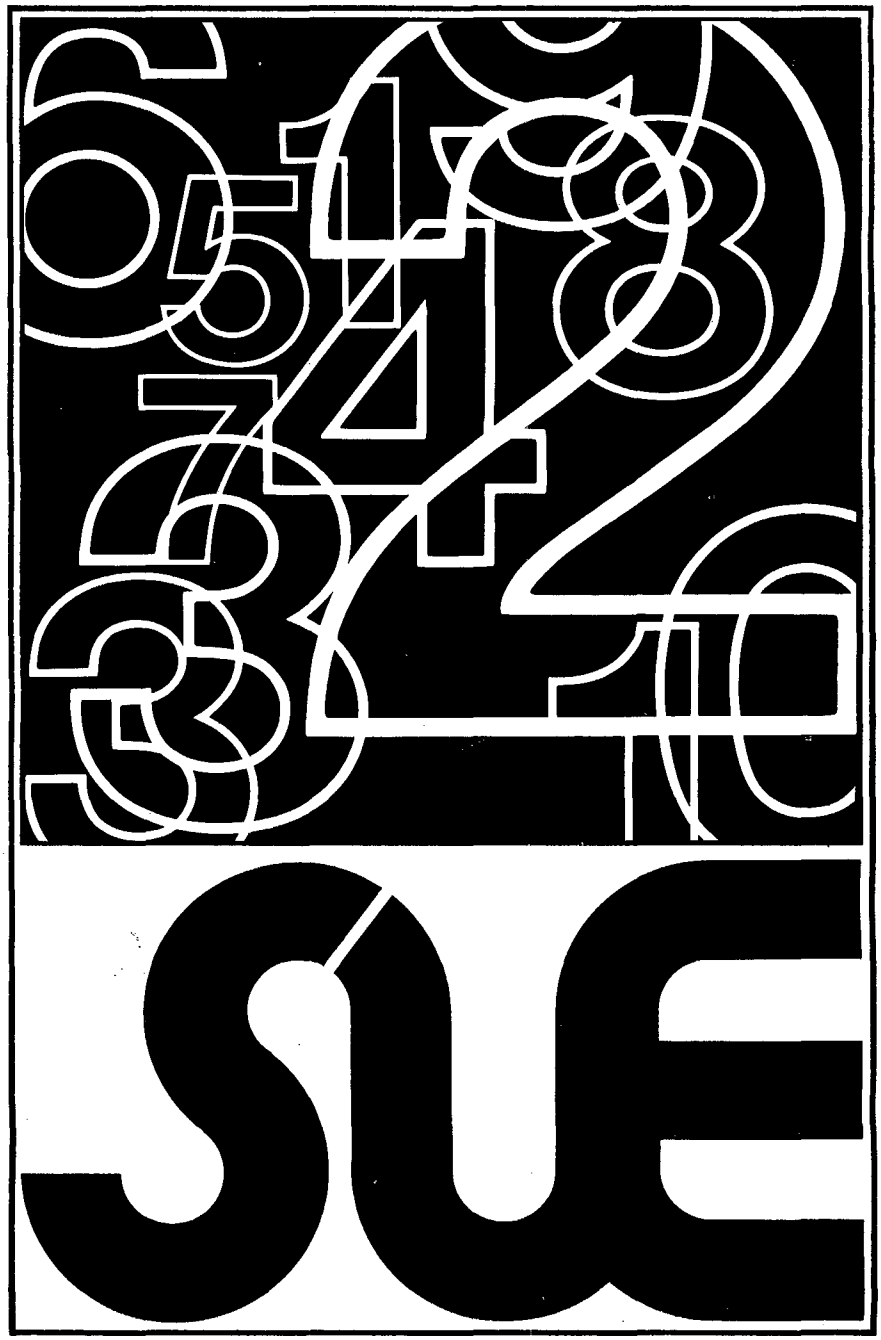
SUE

Lockheed Electronics

Computer Handbook



Lockheed Electronics

Lockheed Electronics Company, Inc.
Data Products Division
6201 East Randolph Street
Los Angeles, California 90040
A Subsidiary of Lockheed Aircraft Corporation

The material in this book is for information only and subject to change without notice.

®SUE is a trademark of Lockheed Electronics Company

Computer Handbook

CONTENTS

CHAPTER 1

THE SUE CONCEPT

The System-User-Engineered minicomputer, SUE, is an entirely new computer concept from Lockheed Electronics. SUE has been designed in detail to permit the system user to specify precisely the computer he needs for his application. SUE incorporates most workable state of the art concepts, in addition, SUE is specifically designed to accommodate new technological advances as they become available without affecting other functions in the system. Also, the system user can easily alter or expand any SUE system to suit changes in application.

Each functional module -- processor, memory, or device controller -- is on a pluggable multilayer circuit card. SUE modules are independent, asynchronous and have identical interfaces. They can be put together in endless configurations starting from a minimal unit with 1K x 16 words of memory on up to a 31K x 16 system loaded with peripherals. Memory size can be expanded even further through the use of a bus coupler.

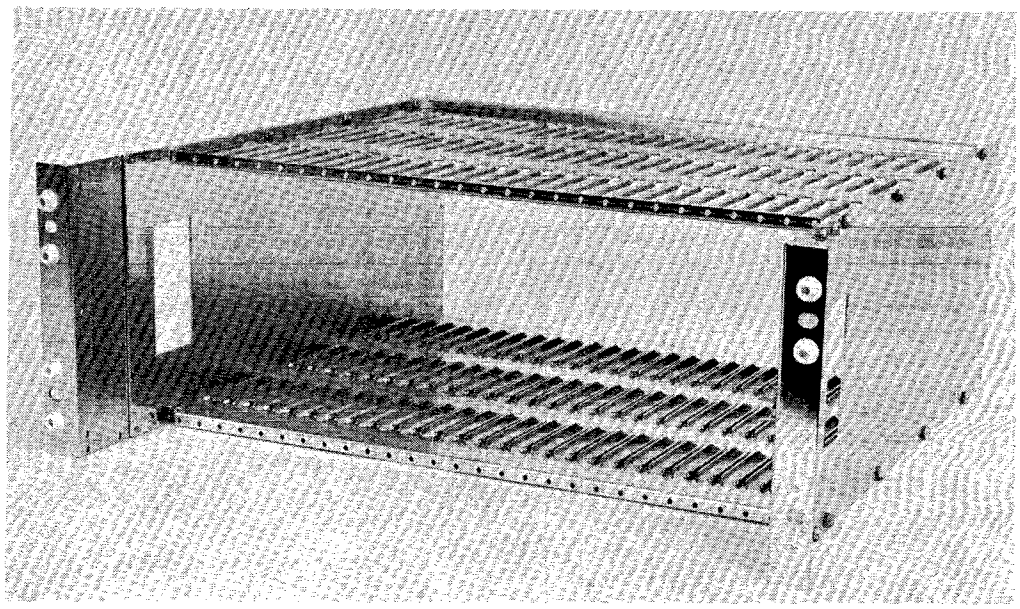
No special wiring is needed to change or add functional modules. Every module can be installed on-site by the system user. System alteration and repair is a simple plug-in operation.

The System-User-Engineered minicomputer solves the most troublesome problems system designers have faced:

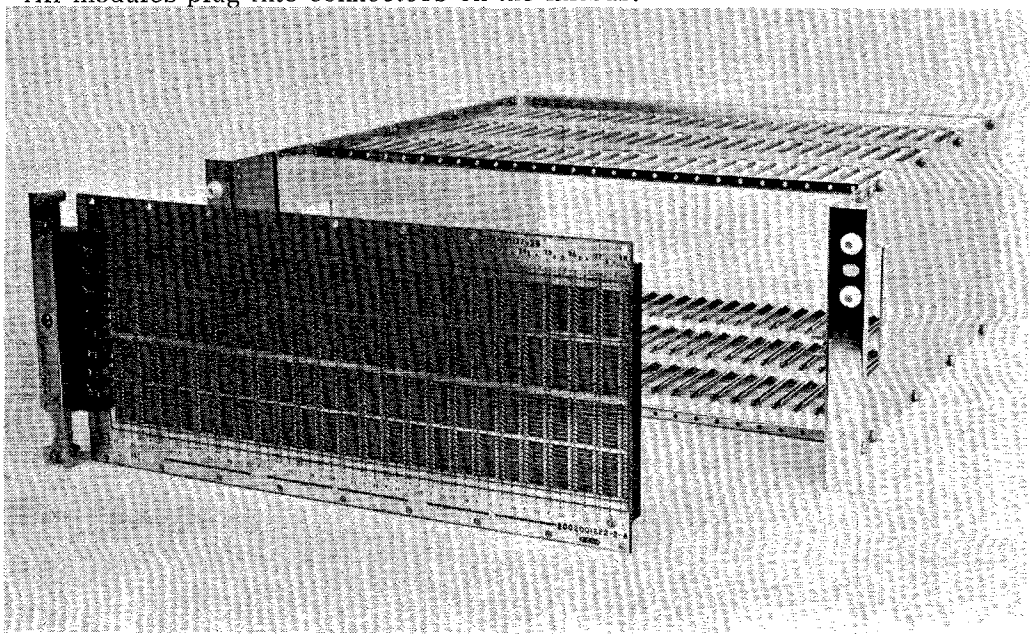
- A minimum cost system with maximum flexibility is easy to specify.
- Design is never frozen. SUE can always be altered or expanded.
- SUE is protected from obsolescence. New technologies can be implemented function by function.
- Architectural design is nearly unlimited. Mixed memories and multiprocessor configurations require no special engineering.

For a simple understanding of the modularity and versatility of the SUE concept here is the way a basic system can be built with total plug-in ease.

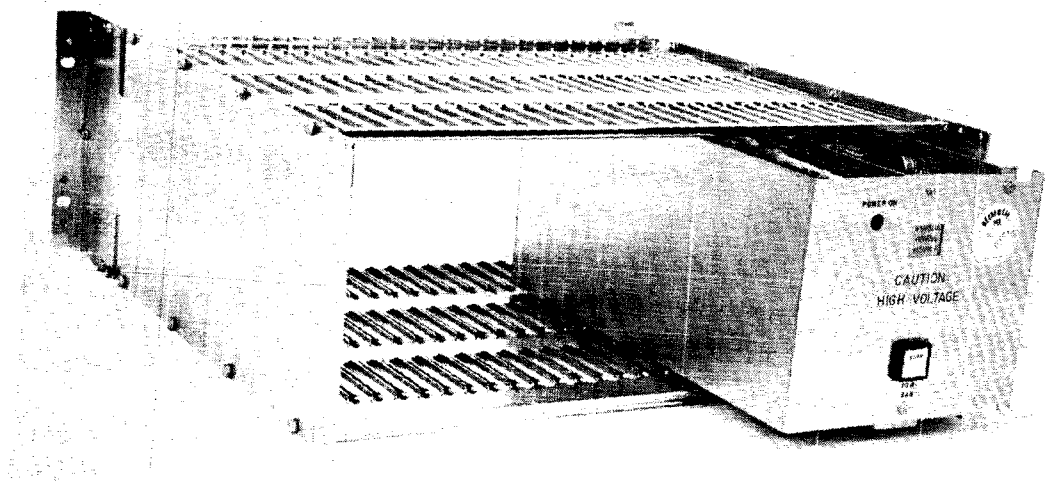
A SUE system starts with a card guide frame.



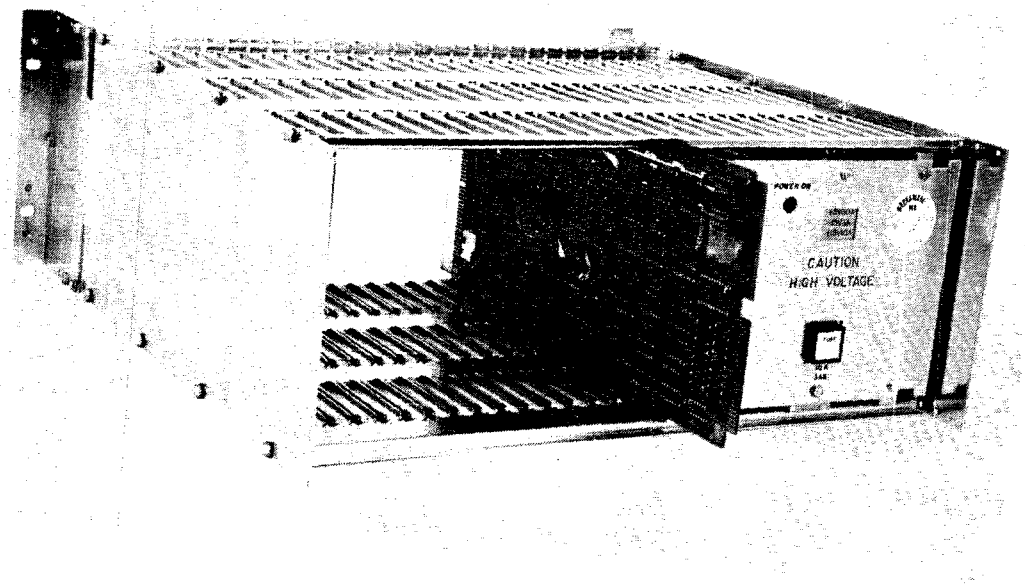
A common bus, called an Infibus, is mounted on one end of the frame. All modules plug into connectors on the Infibus.



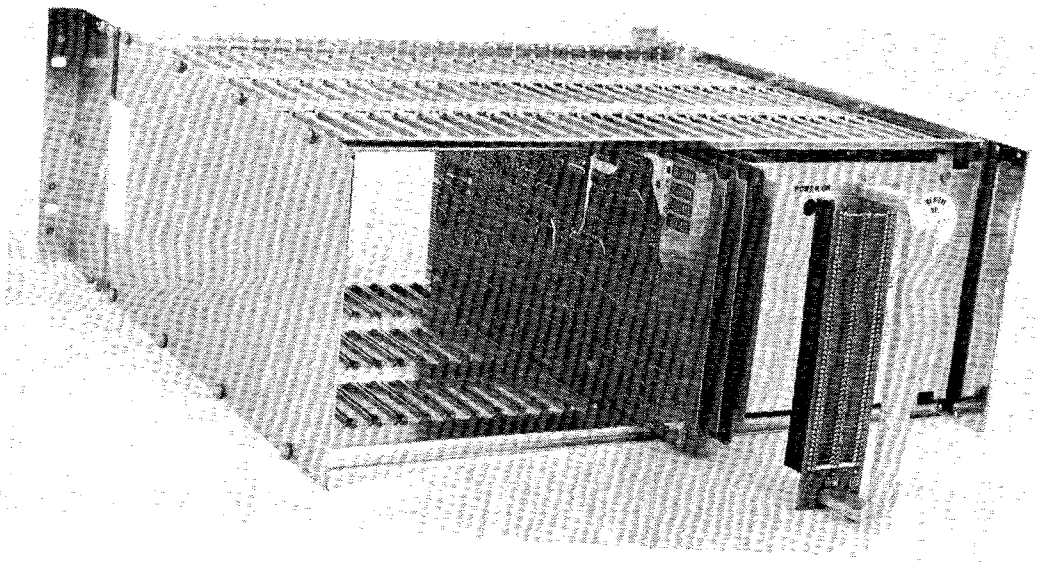
An internal power supply slides in one side of the chassis. This leaves 16 connector slots for system modules.



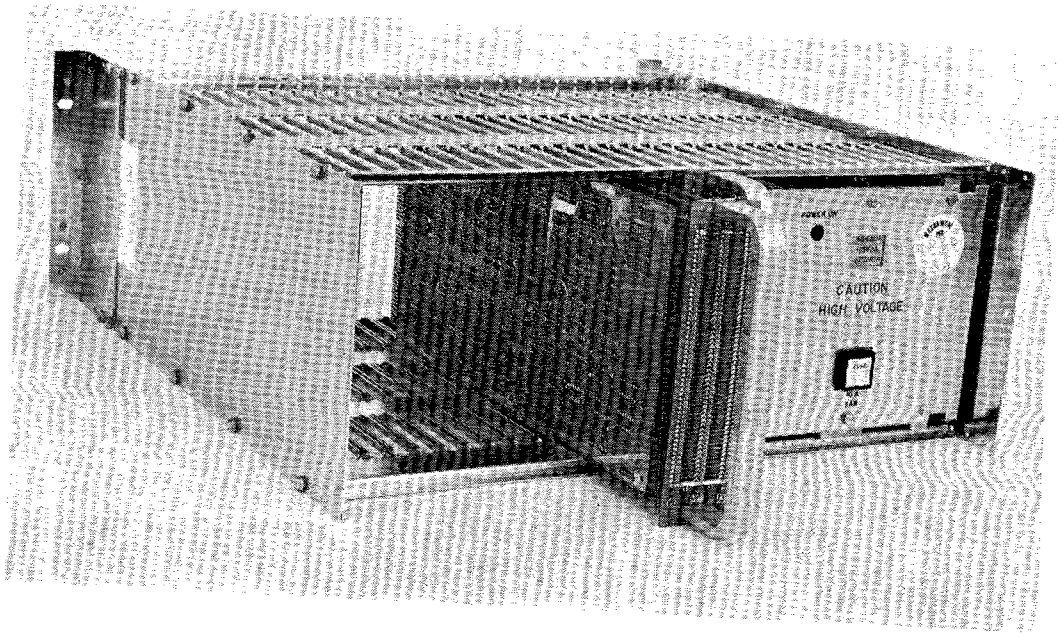
An Infibus Controller Card occupies the slot next to the power supply.



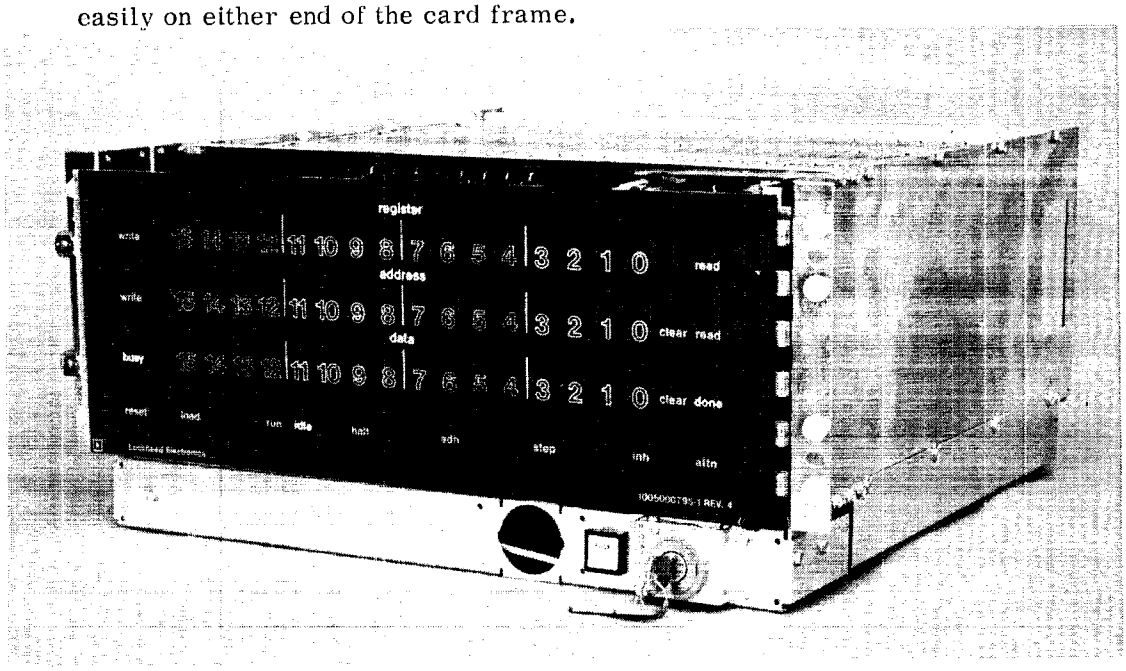
A 2-card SUE processor module is inserted next.



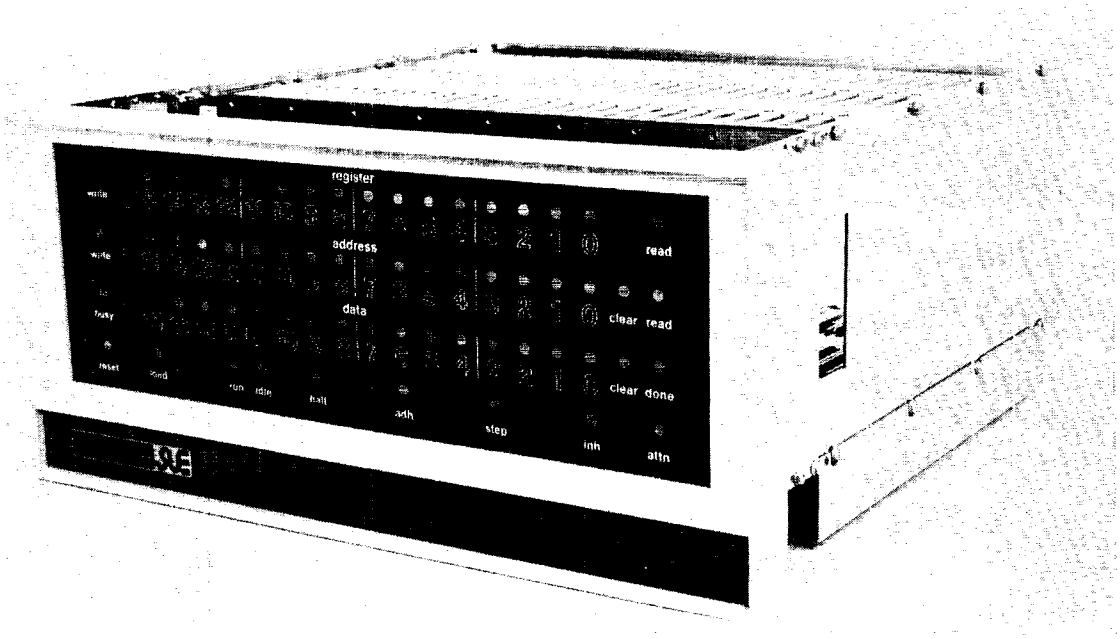
Now, one by one, other modules chosen can be added -- memories, device controllers and the like.



There is a choice of control panels and a decorative bezel. They fasten easily on either end of the card frame.



That's it! With a fan pack mounted under the card frame the system is complete and ready to run.



CHAPTER 2

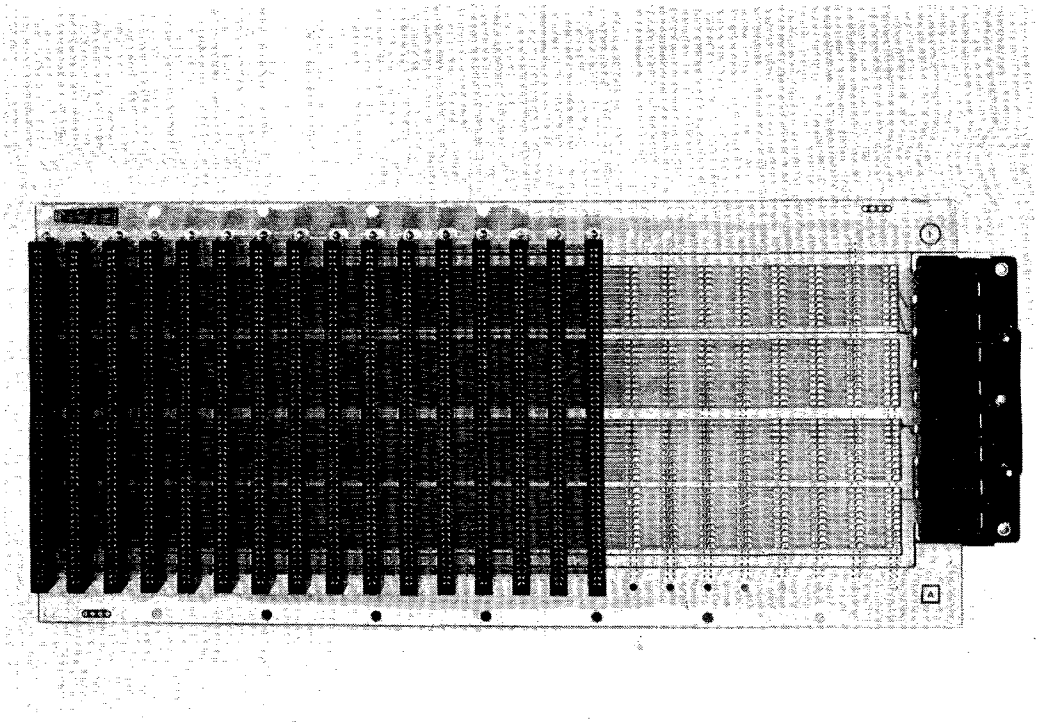
SUE AS A SYSTEM

BASIC SUE SYSTEM ELEMENTS

Basic elements of the SUE system are the Infibus and Infibus Controller, the card guide frame, module cards, and touch response control panel. System functions are subdivided into separate, independent modules, a feature that provides easy expansion and alteration. To protect SUE systems from obsolescence, technological advances can be incorporated in any functional module without affecting others in the system.

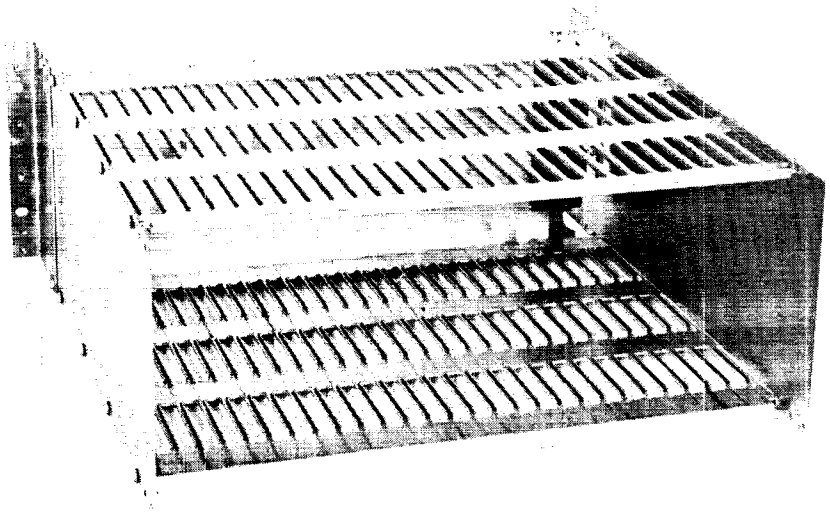
INFIBUS

All communication between SUE system modules takes place on the Infibus and is monitored by the Infibus Controller. The Infibus is a four-layer circuit board that replaces the conventional wire-wrapped back panel. SUE system module cards are inserted vertically into connectors on the Infibus. There can be 24-card slots on an Infibus. If a plug-in power supply is used, 16-card slots are available for system modules.



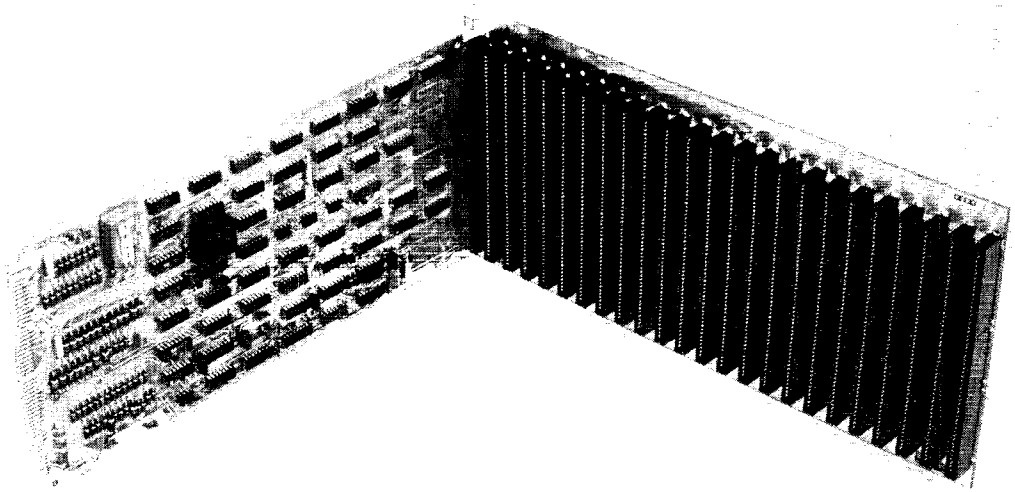
CARD GUIDE FRAME

The card guide frame is rugged, rigid, and inexpensive. The Infibus is screw-mounted onto the back. Vertical airflow is featured through the 7-inch high, 17-inch wide, and 18-inch deep card-guide frame.



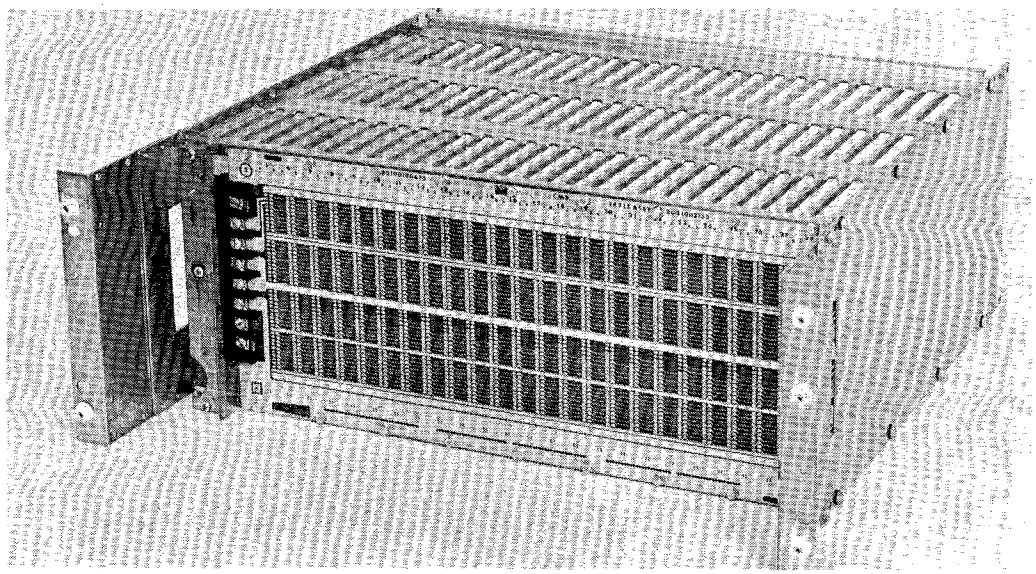
CARDS

All system modules are on 6-1/4 inch logic cards. One or more cards are used for each function. All cards plug directly into the Infibus.



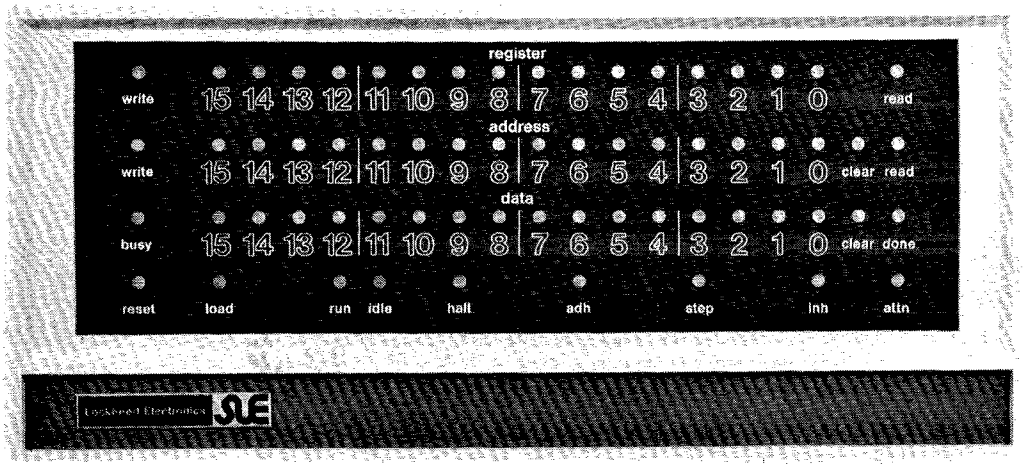
CHASSIS

Together, the Infibus and card guide frame constitute a chassis. The chassis accommodates either front or rear insertion of modules by a simple change in the mounting flanges. Expansion and external device connections are provided via plug-in cables.



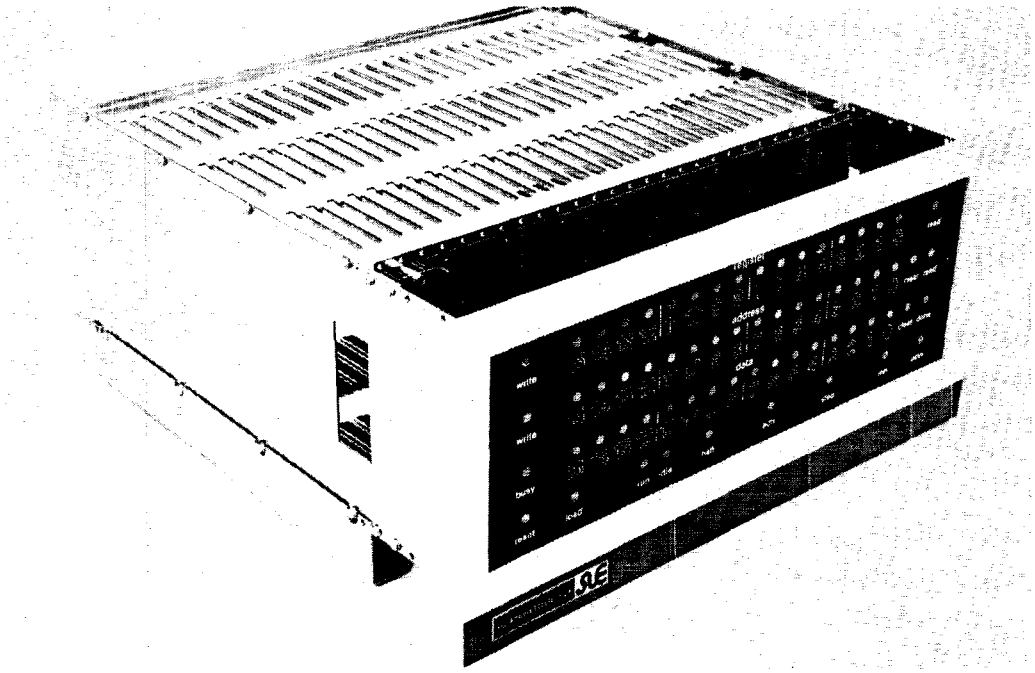
CONTROL PANEL

A SUE system will operate with or without a control panel plugged into the Infibus. This feature allows the control panel to be used only when required — for maintenance or program development. Two control panel options (discussed in Chapter 11) are offered; the low cost System Control Panel and the elaborate Program Maintenance Control Panel which features touch response switches.



SUE, ALL TOGETHER

With the Program Maintenance Panel, system modules, and power supply plugged in, SUE is ready for a-c power.



SYSTEM MODULES

Fundamental to the SUE concept is high-resolution modularity which is completely available to the system designer. For example, starting with the SUE Infibus Controller the system designer can elect to provide his own hard-wired logic, or mechanical mounting, or d-c power, or all three.

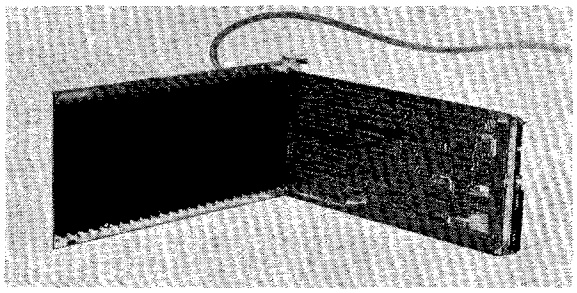
SUE design offers the system designer many choices in addition to fully pre-assembled systems. A typical basic system is described here.

BASIC SYSTEM

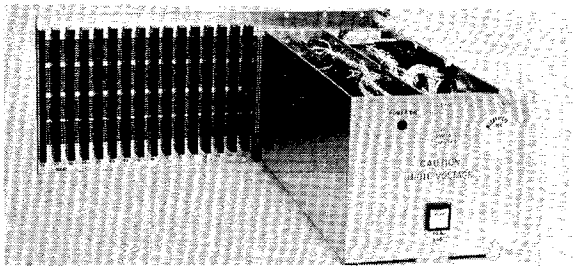
Elements of basic SUE (figure 2-1) are the card frame, Infibus, Infibus Controller, and power supply. The Infibus, together with the card frame constitute a chassis. When a self-contained SUE power supply and the Infibus Controller are mounted in the chassis, fifteen slots remain for mounting other system modules.

If the user supplies d-c power or externally mounts a SUE power supply, 23 slots are available for system modules.

In either configuration, easy plug-in expansion is achieved.



USER SUPPLIED DC POWER



SELF-CONTAINED DC POWER

Figure 2-1. Basic System

1110, 1111, 1112 PROCESSOR MODULES

With the addition of the SUE processor unit, the basic system becomes a powerful general-purpose computer. The processor is a two-card module, with the parallel arithmetic unit on one card and the control memory on the other. The processor contains its own internal data paths for register-to-register transfers to minimize the Infibus time required by the processor.

Four processors can be plugged into the same Infibus, providing a multiprocessor configuration. Physical location of a processor in the chassis determines its precedence over other processor units. All processors requesting service will access the Infibus before any one processor is able to request the Infibus again.

1240 AUTOLOAD MODULE

SUE 1240 Autoload module automatically loads memory from a specified input device. Four 64-word LSI Read Only Memories (ROM) can be installed on a SUE 1240. One of these four ROMs is selected by two manual switches mounted on the module. The open contact of these switches are available at the free edge connector for remote selection. SUE Autoload cards that have ROMs installed and programmed are given specific model numbers, 1241 through 1271. These SUE cards have been programmed to operate with the following input devices:

Paper Tape:	Teletypewriter Models 6710 and 6720 High Speed Paper Tape Readers, Models 6714 and 6715
Disc:	Model 6750
Card Reader:	Model 6736
Cassette Magnetic Tape Unit:	Models 6780 through 6783
Magnetic Tape Unit:	Model 6794
Multiple Devices:	Up to four of the above

3300 SERIES MEMORY

The SUE system designer has a choice of four memory modules, providing a wide range of system performance.

Model No.	Type	Cycle Time
3311	4K x 16 bit words core	850 ns ± 25 ns
3312	8K x 16 bit words core	950 ns ± 25 ns
3320	1K x 16 bit words ROM	250 ns
3330	1K x 16 bit words RAM	200 ns write 250 ns read

SUE memories can be intermixed: semiconductor and core can be implemented in the same system to provide the economy of core and the high speed of LSI/RAM or ROM. The 4K and 8K core memories can be mixed. For example, 12K of core can consist of three 4K modules or one 4K and one 8K module. These memory modules can be used in any combination up to 31K words of memory.

System performance can be enhanced by overlapped or interleaved memory modes of operation. These modes are standard features of the core memory modules, requiring only simple jumper changes on the cards. Cycle-overlap between blocks occurs in both the overlap and interleaved modes, approximately doubling the system throughput rate from 1.18 megawords per second to 2.36 megawords per second with the 850-nanosecond core memory. Overlap operation allows independent memory module addressing, whereas interleaved operation provides cycle overlap with one module recognizing only odd addresses and the other recognizing even addresses.

4500 SERIES DEVICE MODULES

SUE is the first computer system to offer general purpose strappable Input/Output controllers that interface to a wide variety of devices. The multi-application concept reduces the number of spares required for system back-up by allowing a few board types to replace many unique boards in the system. These are the parallel Input/Output controllers 4501, 4503, 4506, 4507, and the SUE 4502 Asynchronous Serial Input/Output Controller.

Both the parallel and serial controllers are single-card modules that plug directly into the Infibus. When two or more devices are requesting Infibus access simultaneously, the device controller closest to the Infibus controller receives access first. To change the precedence of devices on the Infibus, the controller cards need only be swapped in the chassis. This feature provides the flexibility to resolve system timing problems without wiring changes. Peripheral devices connect to their controllers via a cable connector located on the free edge of the card. Controllers that are strapped to a specific configuration are given 4600 series model numbers. Parallel controllers have been configured to control communications with card readers, card punches, line printers,

high-speed paper tape equipment, magnetic tape devices and computer-to-computer communications. One basic design on a single printed circuit board is used in all four models of the parallel controller. The difference in the 4501, 4503, 4506, and 4507 models is the assembly of different input/output components to accommodate the polarity of the peripheral device.

Logic polarities of the four models are as follows:

Model	Data In	Data Out	Command Lines Out
4501	High	High	Low
4503	Low	Low	Low
4506	Low	High	High
4507	High	Low	Low

Serial controller 4502 is designed to control communication with Model 33, 35, and 37 Teletypewriters, asynchronous data modems, CRT displays and other devices that are RS-232C compatible.

SUE 4590 Block Transfer Adapter (BTA) provides block transfer control using the inherent Direct Memory Access (DMA) of SUE. The result is that data can be transferred concurrently to and from memory with processor operations. The BTA is a single card module designed to convert a SUE character-oriented controller to a block transfer controller. The BTA inserts into the slot just to the right of the controller to be converted.

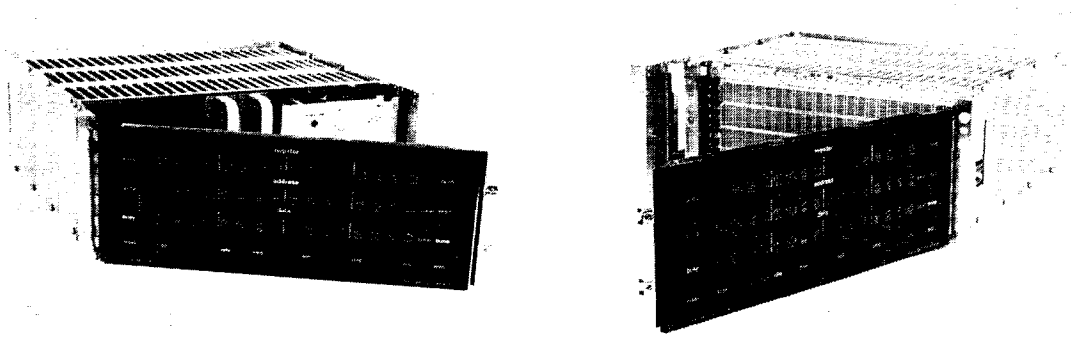
2200 Series Control Panels

A SUE system can be operated without a power distribution unit or control panel. Power on-and-off sensing circuits and software preserve memory contents during power down and provide an orderly turn-on and turn-off operation.

For applications that require a simple on-off switch panel, the SUE 2201 and 2202 power distribution units provide key-operated switches and a-c power outlets for system components. These 1-3/4 inch panels are mounted below the card chassis and extend the overall panel height to 8-3/4 inches. In systems with a fan pack assembly, SUE 7921 or 7922, these panels cover the front of the fan packs.

SUE control panels feature unique touch response switches for easy operation and high reliability. Indicators are light emitting diodes with low power consumption and long life characteristics. All panels can be mounted on the front or rear of the chassis -- a feature that allows front or rear insertion of system modules (figure 2-2). The panels may also be remotely located up to 20 feet from the chassis.

SUE control panels interface to the Infibus with the electronics contained on one or two circuit cards.



FRONT INSERTION

REAR INSERTION

Figure 2-2. Alternate Control Panel Positions

Two versions of the panel are available: SUE 2215 and 2220. Features of the panels are outlined in the following table, with a complete description in Chapter 11.

Model	Switches	Indicators	Circuit Cards
2215	23	17	1
2220	62	65	2

SYSTEM EXPANSION

Infibus Expansion

System modules can be used in sufficiently large numbers to exceed the expansion capability of one Infibus, another can be added as required. Infibus expansion (figure 2-3) is required when all slots within the first chassis are filled. Additional chassis are added and the Infibus is extended with the SUE 1827. The Infibus Driver/Receiver extends all signal lines on the Infibus.

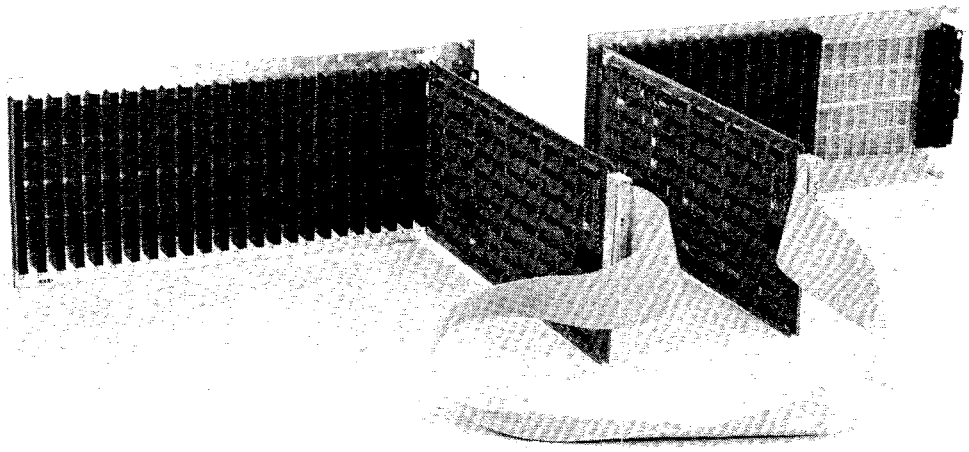


Figure 2-3. Extended Infibus

POWER EXPANSION

Two styles of power supplies are available with SUE: Models 5955 and 5951, can be mounted in the chassis, and the larger Model 5952, is mounted in its own 19-inch rack mountable chassis (figure 2-4).

The typical use of the 5955 or 5951 is in a SUE 7910 chassis that contains 16 slots for system modules. In a larger system these supplies can be mounted externally and cable connected to a 7911 chassis containing 24 slots, or a power supply can be mounted in each 7910 chassis in a multiple chassis system.

Twenty-four slot chassis, with more than a 38 ampere load on the +5 vdc supply or more than 3 core memory modules, require the 5952 supply. See Chapter 5 for a complete description of SUE power supplies.

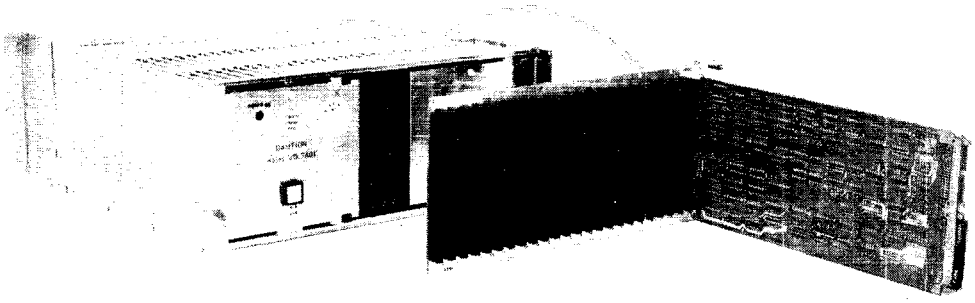


Figure 2-4. External Power Supply

SYSTEM CONFIGURATIONS

The simplest SUE configuration is the "Bikini-Mini" (figure 2-5).

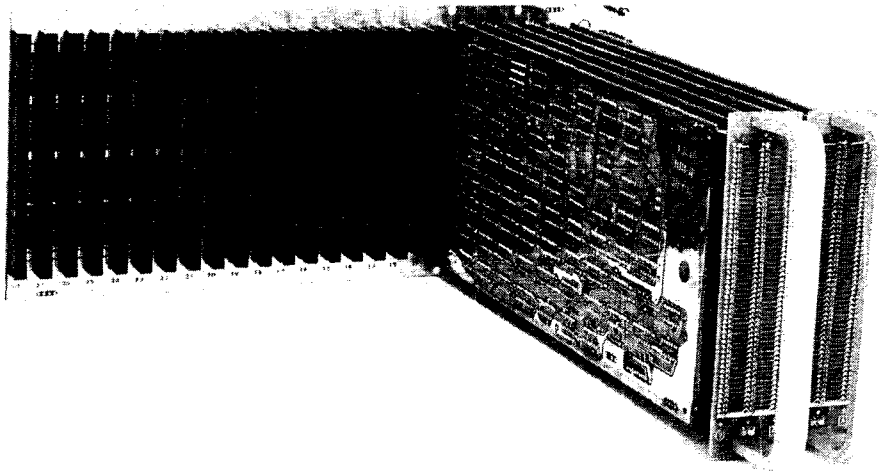


Figure 2-5. SUE as a "Bikini-Mini"

Figure 2-6 shows a small SUE computer with Teletypewriter.

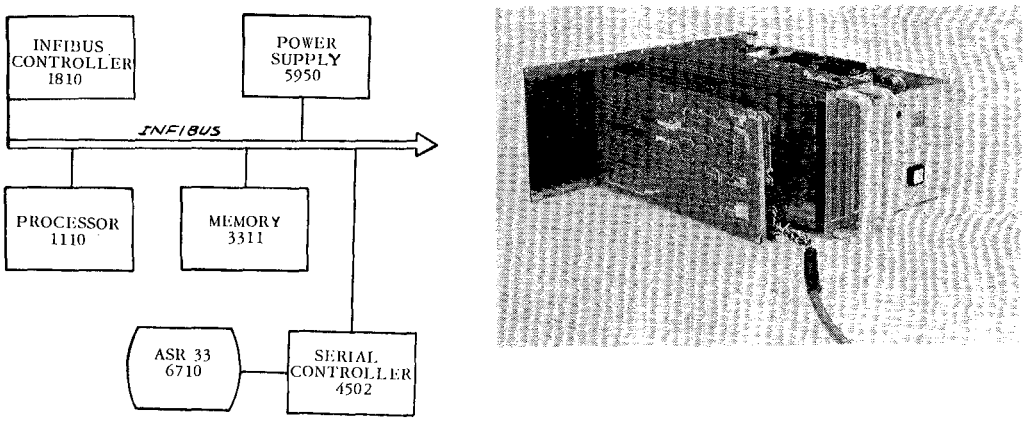


Figure 2-6. SUE as a Small System with Teletypewriter

Next a single 8K memory SUE system supporting both a low-speed Teletypewriter and a high speed paper tape reader is presented. In figure 2-7, arrows indicate low-speed program-controlled transfers interacting with the high-speed block transfers across the Infibus.

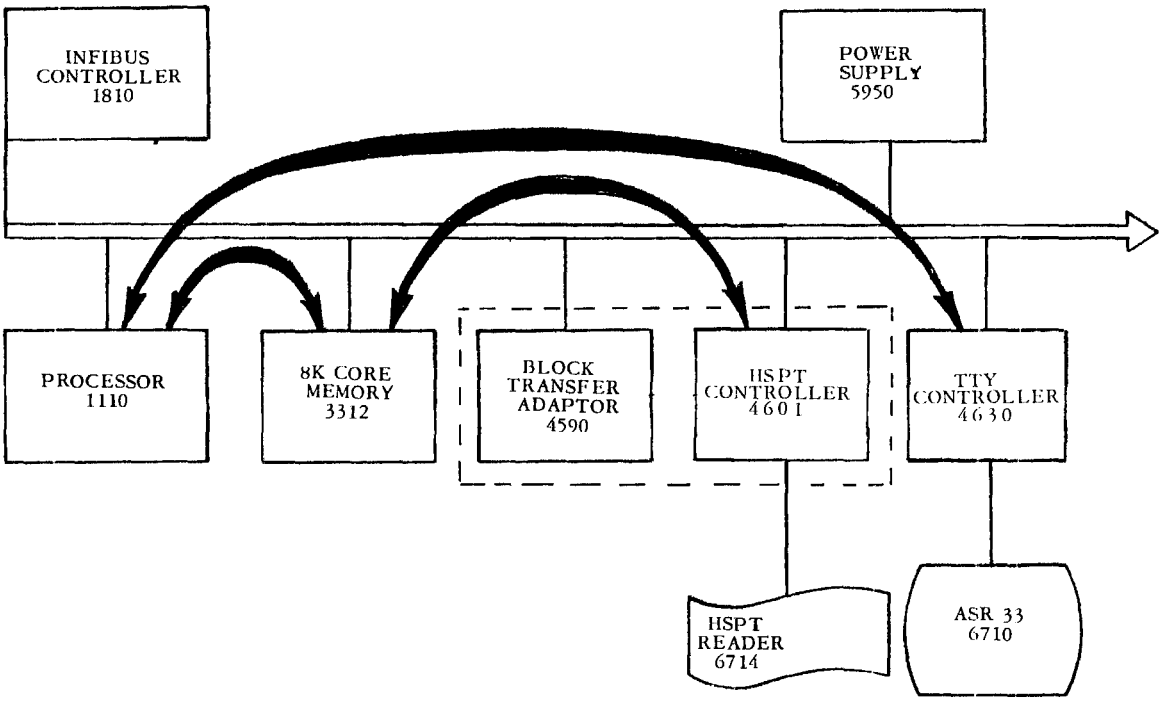


Figure 2-7. SUE Shares a Block of Memory

Expanding to multiple blocks of memory, SUE makes use of built-in interleaving, and overlapping. Figure 2-8 shows how the memory overlap mode of operation with dual high-speed devices can be achieved by assigning buffer areas in different memory modules. Each I/O transfer operates at memory-access rate, doubling the number of transfers per memory cycle.

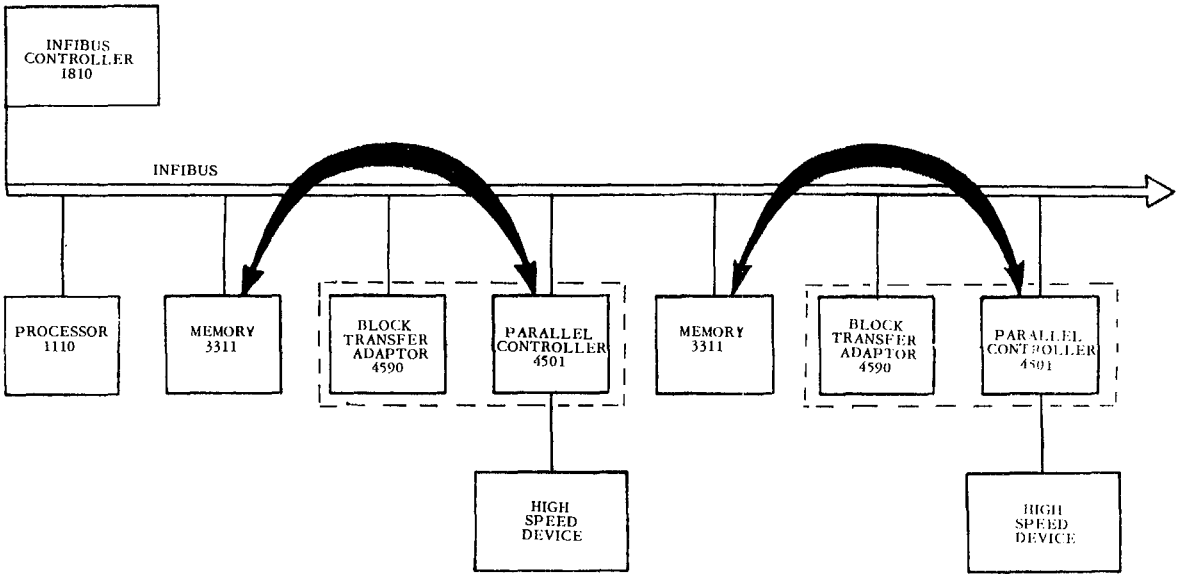


Figure 2-8. SUE Overlaps Memory

To further illustrate SUE expansion capability, 28K words of memory are added with a block transfer Serial Control Module and two block transfer Parallel Control Modules. There are 3 slots left for expansion (figure 2-9).

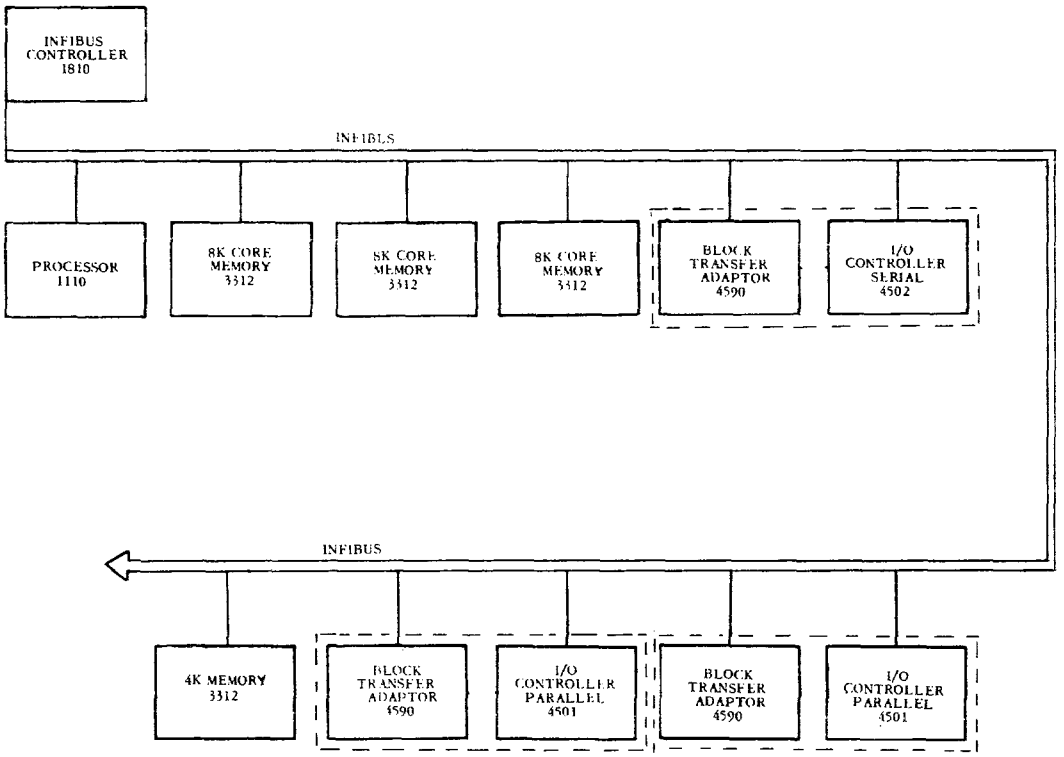


Figure 2-9. SUE as a 28K Memory System

SYSTEM THROUGHPUT

The wide bandwidth of the Infibus allows extremely rapid device operation and bus-access interleaving, resulting in exceptionally high throughput. With block-transfer control, I/O transfers are independent of instruction execution.

Transfer requests on the Infibus are under control of the Infibus Controller. The Infibus Controller resolves device-request priority and grants access to the bus. Queuing occurs during the transfer in process, eliminating any unused time on the Infibus. Device data transfers have the highest priority for Infibus access, system interrupts next highest, and processor requests the lowest priority. Therefore, device data-transfer requests delay the instruction execution of the processor if both request Infibus access at the same time. However, as shown in the remainder of this chapter, the wide bandwidth of the Infibus allows both excellent processor instruction rates and a very high cumulative data rate from devices. This feature makes SUE the highest performance minisystem available today. In a system with multiple overlapped 4K core memories, an accumulative system data transfer rate of 1, 180K words per second is possible with no decrease in processor instruction execution time. Higher data rates are achieved with semiconductor memories.

INFIBUS BANDWIDTH

The Infibus provides a common interface for system modules to transfer data at up to five megawords per second.* All Infibus transfers are asynchronous. The amount of time that signals from each device spend on the Infibus depends upon the source and target device responses, and not on a fixed clock interval. All SUE system modules are designed to minimize Infibus service cycle times. The processor modules contain their own internal data paths for inter-register transfers and the memory modules contain their own address and data registers to allow operation independent of the Infibus. All I/O controllers provide full address and data buffering, minimizing their Infibus cycle time.

For convenience in evaluating the system throughput, nominal transfer rates are used in this chapter. SUE takes advantage of average transfer rates, because it is truly asynchronous and does not have to provide a worst-case timing allowance. The majority of transfers will be to or from core memory. The maximum number of transfers is calculated by determining the longest period of Infibus transfer between the source and target devices. When core memory is the source or target it usually has the longest Infibus transfer time.

*When many devices are interfaced to a SUE, statistical considerations of queueing theory should be added to the following discussions of basic usable data rates.

MEMORY BANDWIDTH

SUE is offered with two basic types of program memories: semiconductor and core. The semiconductor memory has a fast access time with little recovery time required. The SUE semiconductor memory can be accessed at up to 4.0 megawords per second. SUE 4K core memory is an 850-nanosecond cycle, random-access system. Memory modules contain separate address and data registers that allow asynchronous operation.

An individual block of memory can accept requests at a 1.18-megaword per second rate. With two separate blocks of memory connected to the Infibus, several modes of operation can be configured. One configuration is arranged by having memories interlocked so that the memories can be accessed serially (figure 2-10A). That is, the restore portion of the cycle must be completed before the other memory block can be accessed. This mode is used when high memory-transfer rates are not required and system power requirements are to be minimized.

By removing the interlocking jumpers on the memories, two or more memories can be operated in an overlapped configuration (figure 2-10B). Each memory block can be accessed as soon as the currently-accessed block releases the Infibus. Therefore, the next block need not wait for the previous block to complete its restore cycle. Effective memory cycle time in the overlapped configuration is approximately one-half that of a single interlocked memory cycle, raising the transfer rate to 2.36 megawords per second.

By removing the interlocking jumpers and proper address recognition jumpering on the memory cards, memory interleaving can be performed (figure 2-10C). Memory interleaved operation is much like overlapped except the memory blocks alternate accessing of sequential word addresses automatically. Effective memory transfer rate is 2.36 megawords per second for interleaved memory.

DIRECT MEMORY/PROCESSOR THROUGHPUT

The relationship between Direct Data Transfers (DDT) for I/O devices to memory and memory-to-processor transfers is discussed in detail here using core memory as an example. Semiconductor memories allow greater Infibus bandwidth.

INTERLOCKED MEMORY

Interlocked memory results in a maximum of one transfer every 850-nanoseconds to or from a memory location. I/O devices require one memory access for every transfer and by system design have a higher priority than processors. Processor accesses to memory are sometimes two per instruction. The processor runs at maximum rate if no I/O device is active.

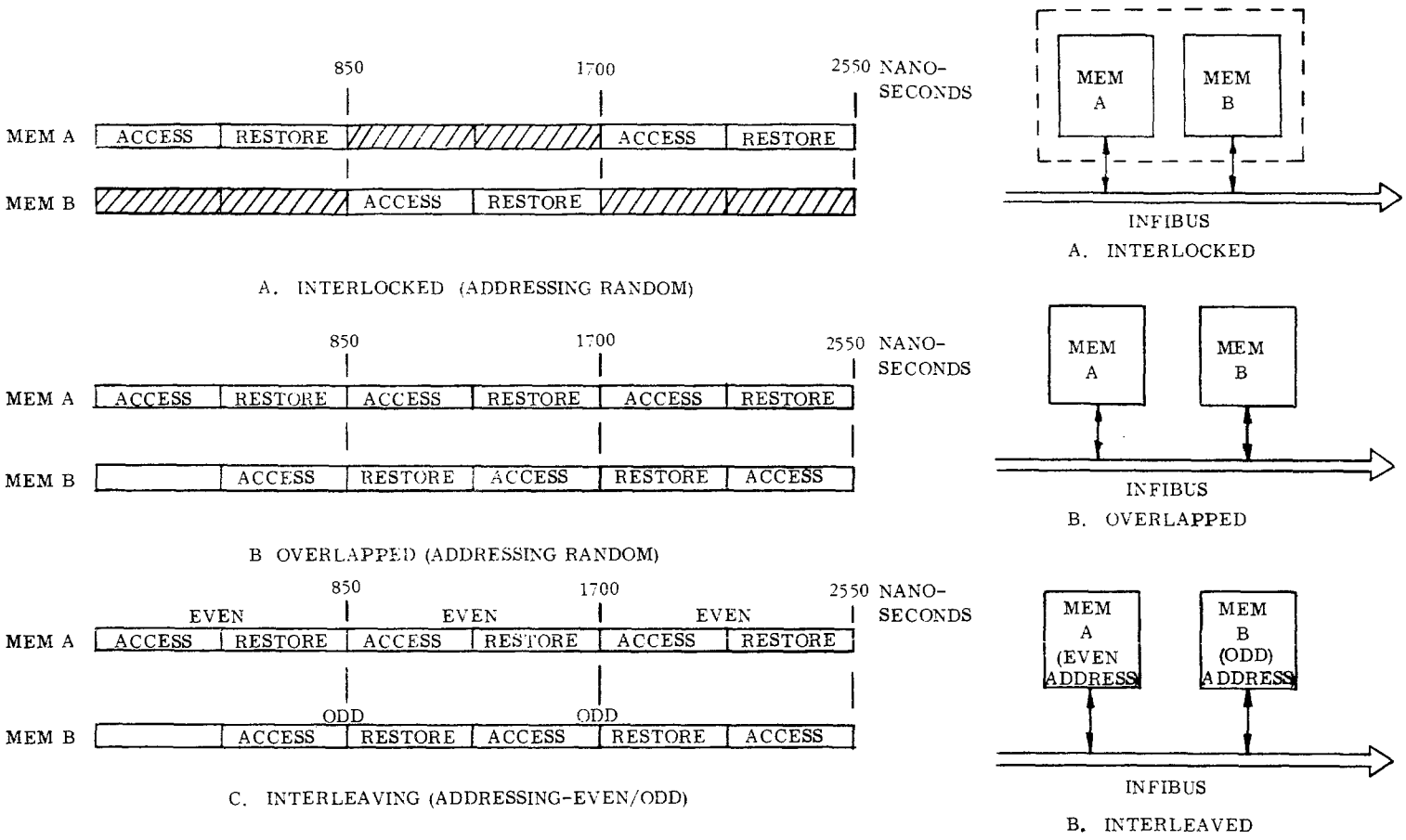


Figure 2-10. Core Memory Operation Comparisons

The processor rate is affected by both the number of I/O devices and the transfer rate of the I/O devices. When a single I/O device is on the Infibus, it is constrained by the following rules:

- o No transfer can be faster than the memory transfer time (850 nanoseconds).
- o The device can gain two successive bus accesses only when there is no contention with the processor.

Another way to consider the interaction of a single I/O device and the processor is to determine the repetition rate of the I/O device and correlate this to the percentage of Infibus time needed. For example, using a high-speed disc with a transfer rate of 50K words per second as the single I/O device, once every 20 microseconds the disc is queued up for an Infibus cycle. The cycle requires 0.85 microseconds because of the memory cycle time. Thus, this I/O device requires only a little over 4 percent of available Infibus time. By queuing Infibus requests in advance, the processor transfer rates are degraded by only 4 percent. The effect on the processor instruction execution can be even less because the memory fetches constitute only a percentage of the instruction decode and execution time.

The maximum single device I/O data rate into core memory is 590K words per second and the processor is degraded to about 50 percent of its maximum transfer rate. Data rates stated here result from the queuing technique that prohibits one device from locking up the Infibus. Systems with only one I/O device are rare, and, if I/O rates higher than 590K words per second are desirable they can be achieved with overlapped memory.

The rules listed for the single I/O device are also true for all multiple I/O devices. With interlocked memory, the maximum memory transfer rate that can be achieved is 1.18 million words per second.

For example, if two I/O devices are present in the system, the highest priority device runs at its full rate if that rate is not greater than 590K words per second. The lower priority I/O device and processor compete to fill the remaining memory transfers. The lower priority I/O device generally gains Infibus access for transfer ahead of the processor. The queuing decision for the next transfer is made at the beginning of the present transfer. This decision is based on the highest priority device that requests the Infibus at that time. Because I/O and processor requests for the Infibus are asynchronous, the processor alone can request Infibus access when the queuing decision is made. If the processor is not queued up for the next access, the I/O devices gets the next two Infibus accesses if their repetition rate is high enough.

If two I/O devices were capable of transfer rates at 590K words per second, the lower priority device would run at less than full capability, because the processor would be competing for the available 590K transfers per second.

When more than two I/O devices are present, probability of the processor achieving memory transfers is a function of the cumulative I/O rate of all devices. Note that lower priority I/O devices and the processor can be locked out if the cumulative I/O rate is excessive. Processor lockout generally occurs only for short periods because high-speed I/O devices are burst oriented.

OVERLAPPED MEMORY

Overlapped memory allows two blocks of memory to be accessed alternately. Each access requires only an average of 425 nanoseconds. If two 4K word memory blocks were used in this manner, one block would contain memory byte addresses 0000_{16}-$1FFF_{16}$ and the other block 2000_{16}-$3FFF_{16}$. Access to each of these memory blocks still requires 850 nanoseconds per block. The average access time of 425 nanoseconds occurs because one memory block releases the Infibus sooner so that transfers to other memory blocks can occur.

One way to take advantage of overlapped memory blocks is to use one block for processor instructions and the other block for I/O data transfers. As long as the single I/O device and processor transfers remain segregated and the I/O rate stays within 1.18 million words per second, the processor rate remains at maximum. Two conditions will tend to degrade the processor rate:

- Exceeding the 1.18 million words per second transfer rate on the I/O memory block
- Mixing I/O devices with the processor memory block or having multiple I/O devices on the I/O memory block

Exceeding the maximum transfer rate on the I/O memory block degrades the processor rate because the Infibus controller queues up the I/O device for access to the I/O memory block during the period of time the processor could be accessing its memory block. The result is that the cumulative I/O rate is held to 1.19 million words per second. The desired I/O rate is limited and the processor rate is degraded.

When it is desirable to increase the cumulative I/O rate above the 1.18 million words per second maximum, I/O device can be mixed into the processor memory block. The result is a further degradation of the processor

rate. The probability of the processor functioning is determined by the rules laid down for both overlapped and interlocked memories. The overlapped rules determine the degradation of the cumulative transfer rate on the processor memory block. The interlocked rules apply to the relationship between the I/O devices and the processor within the processor memory block. The above rules apply only if the I/O devices using the processor memory block have lower priorities than those using the I/O memory block.

If I/O rate is of primary concern, a modified configuration will give a slightly higher I/O rate but will cause more degradation of the processor rate. Place the I/O device with the second highest transfer rate on the processor memory block with the second highest interrupt priority. An extension of this is possible by alternating the I/O devices between the memory blocks (overlapped) as the I/O rates decrease. This will improve the cumulative I/O rate of the system but will degrade the processor rate much quicker. A system to run under the above configuration can be designed by following these rules:

- Decide what maximum cumulative I/O rate is desired (up to 2.36 million words per second)
- Determine the amount of I/O rate to be placed on the processor memory block by subtracting 1.18 million words/seconds from maximum cumulative I/O rate
- Place every other I/O device on the processor memory block until the calculated I/O rate is achieved. Assign access priorities in alternating decending order.

The above procedure insures that the I/O memory block is working at maximum capacity while the processor is transferring at an optimum rate under the provision that your cumulative I/O rate is given top priority.

Multiple I/O devices on the I/O memory block degrade the processor transfer rate even though the cumulative I/O rate does not exceed the capability of I/O memory block. The queuing decision for the I/O devices inhibits the maximum processor transfer rate into the processor memory block.

INTERLEAVED MEMORY

Interleaved memory permits the same improvements in memory access time that overlapped memory provides. When memories are interleaved, one block of memory contains all even-word addresses and the other block all odd-word addresses. This allows consecutive memory addresses to be accessed at a rate of one every 425 nanoseconds. The difference between overlapped and interleaved memories is that overlapped provides for different blocks of contiguous memory addresses to be accessed alternately, interleaved memory provides for a block of contiguous memory addresses to be accessed in odd-even sequence.

I/O devices and processors tend to access memory contiguously. For example, two 4K word blocks of memory in the interleaved mode provide 8K words of memory with all even-word addresses in one block and odd-word addresses in the other. If this configuration were used for processor instructions and the program executed straight through the memory addresses, the average memory access would be 425 nanoseconds. If the average instruction in one program required two memory cycles, the average instruction timing could be reduced by 850 nanoseconds. Interleaving program memory thus increases processor instruction rates.

When more than one device is accessing two interleaved blocks of memory, the maximum memory access rate is 637 nanoseconds. This is calculated from the possible states in which the memory blocks may be when access is desired. If the memory access requests are sufficient to insure that a request is queued up while another is taking place, the access can require either 425 nanoseconds or 850 nanoseconds. The access time depends on whether the access request is for the same memory block (850 nanoseconds) or the alternate (425 nanoseconds). Because these accesses (odd- or even-word addresses) are random in nature, they occur with equal probability, resulting in the maximum average time of 637 nanoseconds. Multiple devices requesting access to a set of interleaved memory blocks follow the guidelines established for interlocked memories with regard to access priorities.

Using the above interleaved memory blocks with one I/O device and processor the I/O device can transfer at rates up to 393K words per second without affecting the normal transfer rate of the processor. The processor functions at the same rate as if it had its own dedicated interlocked memory. Increasing the I/O device rate above 393K words per second degrades the processor rate. Multiple I/O devices accessing two interleaved memory blocks (same configuration as above) follow the interlocked memory access priorities. The maximum possible memory transfer rate is 1.57 million words per second. When multiple high-speed I/O devices are implemented the processor can be locked out entirely.

CONCLUSION

The memory transfer rates discussed above are suitable for most applications. If transfer rates above 2.36 million words per second are desired, they may be achieved in one of two ways.

Core memories operating on separate Infibus subsystems provides cumulative data rates limited only by system design. Intrasystem communication would be via the bus coupler.

Semiconductor memory modules on the Infibus increase the memory transfer rates to 2.87 million words per second for RAM memory and 3.33 million words per second for ROM memory. The separate Infibus subsystem approach can be applied to semiconductor memories, further enhancing transfer capability.

CHAPTER 3

INFIBUS AND INTERFACES

The previous chapter presented an overview of system throughput in terms of frequency. The first part of this chapter shows the "Nanosecond Handshaking" that produces such high performance. The second part of this chapter presents details for interface to the Infibus. In most cases, SUE universal controllers and Block Transfer Adaptors can handle all common peripherals. If a unique interface is required, the user can take advantage of several SUE modules specifically designed as aids to custom interfaces. Timing details of the Infibus are contained in Appendix H.

THE INFIBUS AND INFIBUS CONTROLLER

All communication among SUE system modules takes place via the Infibus and is monitored by the Infibus Controller. The 110 lines of the Infibus are common to all pluggable modules. The communication discipline described in this chapter is used by all modules to maximize system throughput. Infibus communication is asynchronous and provides full-word data transfers up to five megahertz.

Four types of communication use the Infibus:

- Data Access and Service (Direct Memory)
- Processor Access and Service (Instruction Stream)
- Interrupt Access and Service (Real Time Response)
- System Control

The communication capability of the Infibus and Infibus Controller permits Direct Data Transfers (DDT) to and from memory concurrent with the processor instruction stream.

Under block transfer control both high speed and low speed devices can asynchronously request Infibus access and obtain service for direct memory transfers without interrupting the processor instruction sequence.

Interrupts, on the other hand, are used to divert the instruction sequence to subroutines for

- Initialization and termination of block transfer control

- Execution of programmed data transfers by processor
- External real time response
- Internal self-interrupts

Data and processor access and service are discussed before interrupts to illustrate the extremely fast interaction of multiple devices and the processor on the Infibus. This presents an inside view of the interaction process that allows the high throughput of the Infibus.

PHYSICAL DESCRIPTION

The Infibus is a multilayer printed circuit board with provision for sixteen or twenty-four, 110-pin circuit card connectors. Each connector contains two rows of 55 pins each on 0.10-inch centers. The Infibus printed circuit board is 6.75 x 13.75 inches. System module cards are inserted into the connectors in a vertical plane, assuring maximum vertical airflow. These cards have beveled, edge-wipe connectors for easy insertion and extraction.

The chassis can contain a pluggable power supply. In this case there are sixteen connectors available for system modules. The power supply is inserted on the right side of the chassis. The adjacent system module is the Infibus Controller that allows other system modules to access the Infibus.

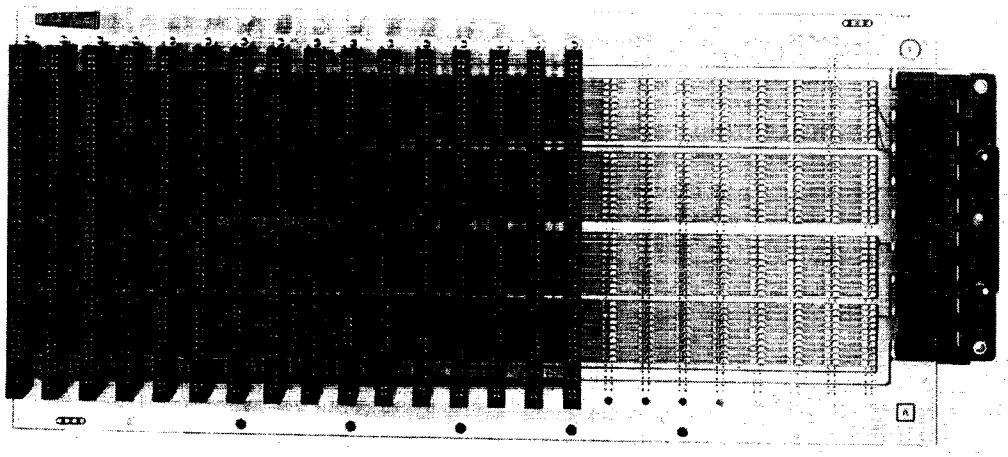


Figure 3-1. The SUE Infibus Showing Module and Power Supply Connectors

Subsequent system modules (right to left) must be the processor, or processors, followed by other modules such as I/O device controllers, and system expansion modules. Memory modules can be located in any slot positions within the chassis. The location of system modules determines precedence within their class, for Infibus access. Those modules closest to the Infibus Controller have the highest precedence. No empty slots can be left between modules.

The Infibus has two outside layers for signals and the two inside layers for system ground and power planes. Signals and lines are connected to common connector pins as follows:

	Pins
Address	16
Data	16
Infibus Control	10
Memory	5
Read/Write Control	3
Interrupt Sensing	8
Block Transfer	8
Control Panel	10
Power Fail	2
Ground	12
+5 vdc	8
+15 vdc	4
-15 vdc	4
Clock	1
Spare	3
TOTAL	110

Appendix I shows the Infibus pin assignments.

SYSTEM MODULE COMMUNICATION

Infibus communication occurs in two overlapped cycles: the bus select and bus service cycles. A system module that requires the Infibus to perform data transfer with another system module can request bus access while a bus service cycle is in operation. The Infibus Controller is the system module that selects the service request to be honored. The requesting system module (master) must acknowledge when selected for bus service; the system module being written into or read out (slave) completes its service cycle with a DONE as quickly as possible. The Infibus Controller monitors this as watchdog timer;

if the module requesting bus access does not acknowledge within one microsecond, the Infibus Controller removes the Select signal and processes the next bus access request. If the bus service time exceeds two microseconds, the Infibus Controller signals the master module to QUIT, freeing the bus for other access

This overlap of bus access request and bus service maximizes system data throughput via the Infibus. Optimally, this is 200 nanoseconds per service cycle. Therefore, the maximum system communication rate is five million 16-bit words per second.

The four types of Infibus communication use some separate and some common signals on the Infibus. The type of communication depends on the special functions the system module is to execute. Master modules may request bus access and, when granted, cause another system module to receive or transmit data. Examples of Master modules are Processors, Block Transfer Adapters, and Control Panels. Memory modules are always slave modules because they can never cause another module to execute an action.

All significant registers of system modules are addressable as are words in memory. Any system module can execute bus service cycles and operate in the slave mode. This includes the master modules as well as the I/O device controllers. The I/O device controllers signal the Processor that they are ready for action via the Infibus Interrupt Lines.

Several Infibus signals are provided for special system functions: These include system clock (25 megahertz), Master Interrupt Inhibit, Master Reset and External Attention Switches, Power Status, and a power source frequency signal (50 or 60 Hertz) from the power supply.

INFIBUS OPERATION

There are three methods for requesting service

- Service Request D (SRLD) indicates a device request for Direct Data Transfer.
- Service Request 1 through 4 (SRL-1-4) indicates a request to interrupt the processor.
- Service Request C (SRLC) indicates a processor request for access to the bus.

Simultaneous requests on two or more service request lines are resolved in the Infibus Controller by priority. The priority of the service request lines from the highest to the lowest are

Service Request Line	Priority
SRLD - Direct Data Transfers	1st
SRL4 - Interrupts	2nd
SRL3 - Interrupts	3rd
SRL2 - Interrupts	4th
SRL1 - Interrupts	5th
SRLC - Processor	6th

If several devices request service on the same service request level, the device physically closest to the Infibus Controller gains access. This is accomplished by trapping the Precedence Pulse. The Precedence Pulse is chained from one device to the next, with the device physically closest to the Infibus Controller receiving the pulse first. Once a processor has access, however, it cannot trap the Precedence Pulse on the next cycle, eliminating the possibility of a processor tying up the Infibus.

Interrupt Priorities

System modules are assigned interrupt levels according to function priority. Standard software uses the following interrupt level assignments:

Level Four (SRL4) is reserved for system fault and real-time-clock interrupts. This is the highest level. The Infibus Controller uses this level to indicate that it has detected a power failure, power restore, or a line-frequency interrupt.

Level Three (SRL3) is reserved for high-data-rate input-output devices such as discs, drums, magnetic tapes. These devices normally are controlled by a Block Transfer Adapter that communicates directly with a memory module in a block mode. The central processor is interrupted on SRL3 when the end of the block transfer occurs.

Level Two (SRL2) is used for slow-data-rate devices such as card and paper-tape readers and punches. I/O controllers use this level to signal the processor that a word or byte is ready for programmed transfer or block transfer completion.

Level One (SRL1) is shared by manual system interrupts. The operator-attention switch on the control panel causes a level-one interrupt.

DATA SERVICE REQUEST D (SRLD)

When the select line (SELD) is inactive, any one or more master devices can assert the service request SRLD (top of figure 3-4). If no master device is waiting for the bus, the Infibus Controller will assert the SELD line and the Precedence Chain (PCDA/B). SELD goes to all devices temporarily preventing any additional requests for the Infibus. The precedence pulse is propagated (chained) through all devices until it reaches the first master device requesting the Infibus. The requesting master device highest in the precedence chain will block further propagation of this signal. The receipt of the precedence and SELD indicates that this master device will be given the next bus cycle. The master device must assert Select Acknowledge (SACK) and remove its service request SRLD (figure 3-2 for service access lines).

The assertion of Select Acknowledge (SACK) is recognized by the Infibus Controller which removes the precedence (PCDA/B) and select (SELD) signals. The waiting master device now inhibits further queuing on the Infibus by means of its SACK signal.

The waiting master device monitors the strobe line (STRB) to determine when the Infibus has been released by the previous master and has become idle. When STRB is removed, the selected master device asserts address and control lines, and the data lines if the operation is to be a write. After a delay, to allow for address deskewing, the waiting master device asserts the strobe line, and removes SACK to allow the Infibus Controller to select another master device (figure 3-3 for service lines). All slave devices examine the address and control lines. Timing is derived from the strobe line.

The slave device addressed must respond with the service completion signal DONE. If the transfer is a write (RITE) to the addressed slave device, DONE indicates to the master that the data to be written into the slave has been taken from the lines. When the DONE signal from the slave is recognized the master removes strobe, address, data, and control. The write operation for this master has been completed and the Infibus is idle, unless a newly selected master asserts its address and control signals. Overlapped selection of a new master was enabled when the current master removed its select acknowledge (SACK).

In a read operation, the selected master asserts only address and control lines when it recognizes that strobe (STRB) has been removed by the previous master. It waits for address deskewing and then asserts its strobe for the addressed slave to place the data on the data lines along with a delayed DONE signal. When the master recognizes the DONE signal, it samples the data and removes its address, control and strobe. The read operation for this master has now been completed. The Infibus can be idle or a newly selected master can assert its address and control signals. Overlapped selection of a new master is enabled when the current master removes its Select Acknowledge (SACK).

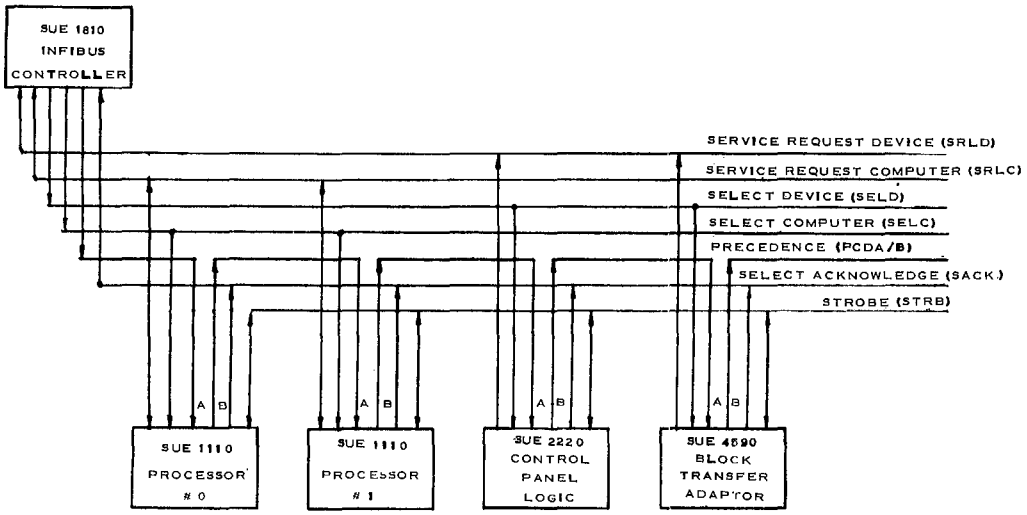


Figure 3-2. Infibus Access Signals

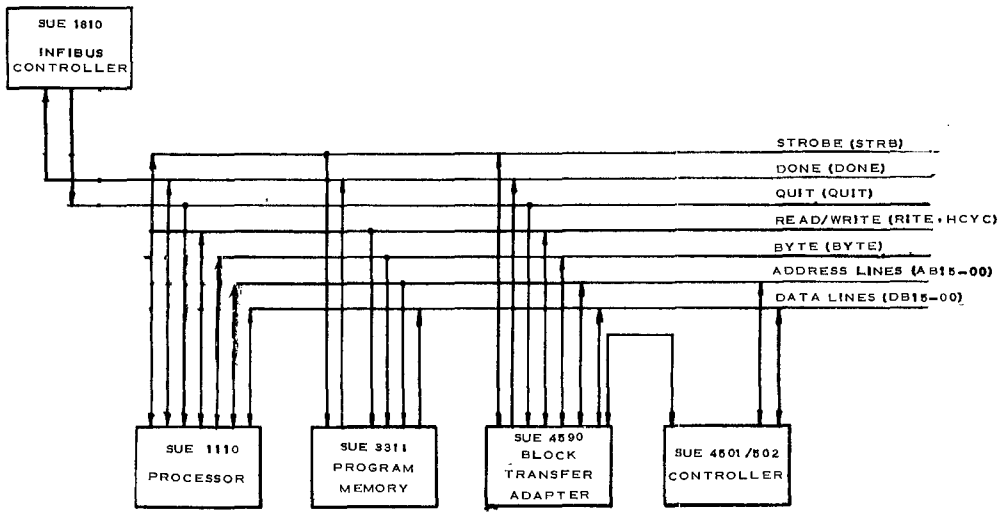
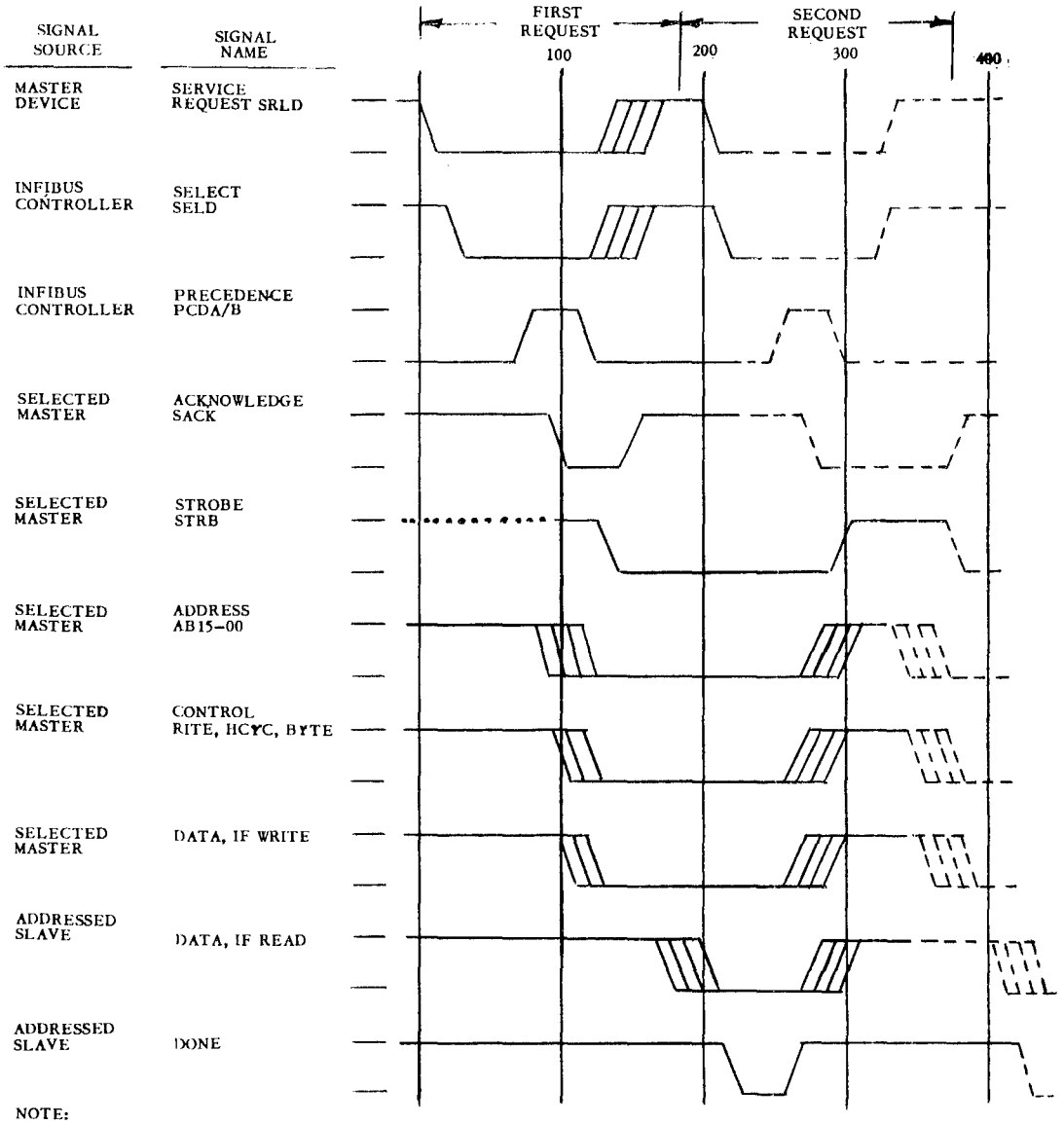


Figure 3-3. Infibus Service Lines



NOTE:

TIME IN NANOSECONDS

FIRST REQUEST AND SERVICE SIGNALS IN UNBROKEN LINES

SECOND REQUEST AND SERVICE SIGNALS IN DASHED LINES

PREVIOUS STROBE CONDITION IN DOTTED LINES

Figure 3-4. Overlapped Infibus Request and Service Timing

Figure 3-4 defines Infibus service overlap. The second request begins before the first request has completed the transfer. The second request transfer is shown as dashed lines.

Receipt of DONE by the master device indicates successful completion of the bus service. If DONE fails to occur within 2 microseconds, the Infibus Controller will assert the QUIT line, indicating a cycle abort. The occurance of DONE or QUIT causes the master device to remove its asserted lines, including the strobe. This indicates to all other devices that another transfer may begin. A cycle abort is stored by the master device as a status, indicating that the addressed slave device was either not in the system or was inoperable.

PROCESSOR SERVICE REQUEST (SRLC)

All Processor modules use Infibus Service Request and Select lines that are specifically dedicated for processor use. Processor bus access priority is assigned lower than that of system modules such as Block Transfer Adapters and Control Panels (figure 3-2).

Operation on the Infibus for processor access and service is the same as described above for device operation. Although four processors may share one Infibus, processors are designed not to monopolize the Infibus once access is granted. A processor may not request bus service until the service request signal, SRLC, is removed.

When in the master mode, processors request service by asserting SRLC. The Infibus Controller will respond with the select signal, SELC, and the precedence signal, PCDA, providing a service request (SRLD, SRL1, SRL2, SRL3, or SRL4) from any non-processor module is not present. The processor asserts the select acknowledge signal, SACK, and removes its SRLC. However, a second processor may keep SRLC active. The first processor will not request a second access until SRLC becomes inactive, which allows all processors access to the Infibus on sequential service cycles. The selected processor monitors the strobe line. When the strobe is removed, the processor asserts its data, address, and control signals as required by either a write or read operation and awaits a DONE signal from the slave module.

INTERRUPT SERVICE REQUEST (SRL1-4)

The Infibus provides a set of four shared-interrupt lines (SRL1-4) that are used by devices to signal the Infibus controller of an interrupt condition and request access to the Infibus so that the interrupting module can transmit its module address on the Infibus. This module address is stored by the central processor into a fixed memory location within the system executive space.

Corresponding to each of these four bus service requests for system interrupt are four select lines (SEL1-4) used by the Infibus Controller with the precedence signal to notify the interrupting device that it can have bus access. This bus access operation for interrupt function is similar to the bus access operation for device and processor access.

SUE computers have two classes of interrupts: system interrupts and Processor Self-Interrupts. System interrupts are generated by non-processor system modules and by external real-time signals.

System interrupts may be signaled on any of the four lines. Simultaneous requests are resolved on a priority basis. Service request line, SRLD, for bus access from direct data devices is given highest priority; the four interrupt Service Request Lines, SRL4-1, are next, and the processor request line, SRLC, is given lowest priority for bus access.

Once a system interrupt request is selected on any of the four levels, all four interrupt levels are masked automatically by the processor. If a specific interrupt level deserves priority over the current level, it is necessary for the current interrupt routine to enable that level.

Processor Self-Interrrupts

Processor self-interrupts do not use the interrupt lines of the Infibus. There are two kinds of processor self-interrupts: detection of an unimplemented instruction, and bus cycle abort. If a processor detects an unimplemented instruction, it will store the instruction, current status, and address of the instruction into the executive space and will "trap" to the vectored unimplemented instruction routine whose address was stored in executive space.

If a processor detects a QUIT signal from the Infibus Controller indicating that a processor bus service cycle was not completed within two microseconds, it will interrupt the instruction sequence and store the unaccepted address, current status and address of the aborted instruction, then vector to a bus cycle abort routine. The executive space allocation for the central processor self-interrupts is shown in Table 3-1.

System Interrupts

The Infibus Controller continuously monitors the status of the interrupt service request lines. If an interrupt service request occurs and system interrupts are not inhibited by a control panel switch, and the requested interrupt

Table 3-1. Self-Interrupt Executive Space

Unimplemented Instruction Abort	Unimplemented Instruction	Status	Address of the Unimplemented Instruction	Unimplemented Instruction Routine Vector
	0020*	0022	0024	0026
Bus Cycle Abort	Aborted Instruction Operand Address	Status	Address of the Aborted Instruction	Abort Routine Vector
	0028	002A	002C	002E

*Memory Executive Address

level is not masked by a programmable mask bit, the Infibus Controller signals the processor that it must interrupt its instruction execution sequence at the end of the current instruction.

At the end of the instruction, the Infibus Controller provides the processor with the number of the highest priority level interrupting. The Infibus Controller transmits the corresponding select (SEL1-4) and the precedence (PCDA) signal.

The interrupting device on the selected level with the highest precedence, as determined by its proximity to the Infibus Controller, asserts the acknowledge signal, SACK. The Infibus Controller removes the select line and services other Infibus access requests. The interrupting device now transmits its device number on the Infibus and asserts the strobe signal indicating the Infibus is busy and its module address is to be read by the processor.

The central processor receives the interrupt level number and module address, stores the module address, its current status and program counter into the executive space corresponding to the selected level and vectors to the interrupt routine. Executive space allocation for system interrupts is shown in Table 3-2.

This priority interrupt process avoids device polling because the module address of the interrupting device is automatically transmitted.

The interrupt response time is 5.58 microseconds (850-nanosecond core memory). This time includes all of the context switching--device number, program counter, status register, and interrupt vector--and the fetching of the first instruction of the interrupt service routine from core memory.

Table 3-2. System Interrupt Executive Space

LEVEL 4	MODULE ADDRESS 0018	STATUS 001A	CURRENT PROGRAM COUNTER 001C	SERVICE ROUTINE VECTOR 001E
LEVEL 3	MODULE ADDRESS 0010	STATUS 0012	CURRENT PROGRAM COUNTER 0014	SERVICE ROUTINE VECTOR 0016
LEVEL 2	MODULE ADDRESS 0008	STATUS 000A	CURRENT PROGRAM COUNTER 000C	SERVICE ROUTINE VECTOR 000E
LEVEL 1	MODULE ADDRESS 0000	STATUS 0002	CURRENT PROGRAM COUNTER 0004	SERVICE ROUTINE VECTOR 0006

Figures 3-4A and 3-4B define the context switching process and the return from an interrupt service routine to the interrupted program.

The return from interrupt instruction (RETN) transfers the contents of a memory word into the system status register and the contents of the next memory word into the program counter (PC). This one-word instruction restores system status, transfers program control to the return address of the calling function and restores the enable/disable conditions of the interrupt levels. RETN takes 4.26 microseconds.

The automatic context switching during the interrupt response makes the optional sharing of interrupt levels easy for the programmer. All addresses between $F000_{16}$ and $FEFF_{16}$ are reserved for register addresses and device identification. All system devices that can generate an interrupt are assigned a set of 16-bit addresses for registers. The device number (same address as the status register) is used by a computer interrupt program to identify the device with the highest precedence on shared interrupt lines.

INFIBUS SIGNAL DESCRIPTION

This description of a selected set of Infibus signals is an aid to understanding Infibus requests and service.

Four signals are used for direct data bus access:

SRLD — Service Request Line, Data.

SRLD has one load (the Infibus Controller) and as many sources as there are master device modules in the system. It is used by a device to request bus access

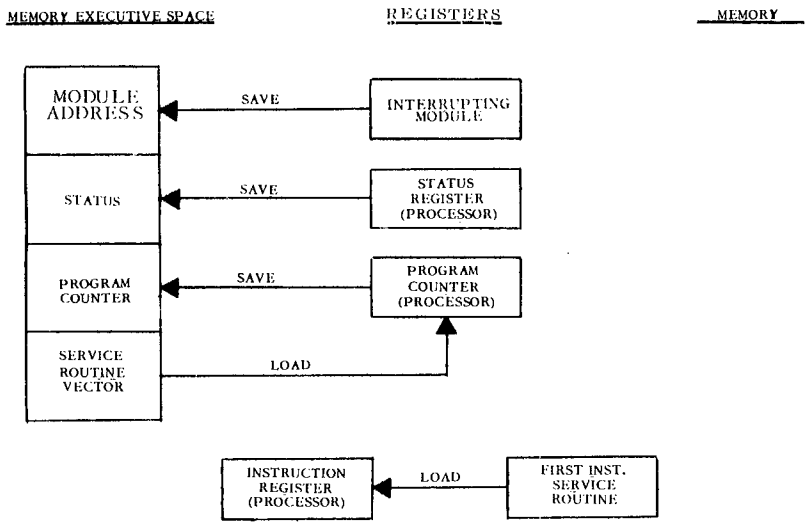


Figure 3-4A. Interrupt Response Sequence

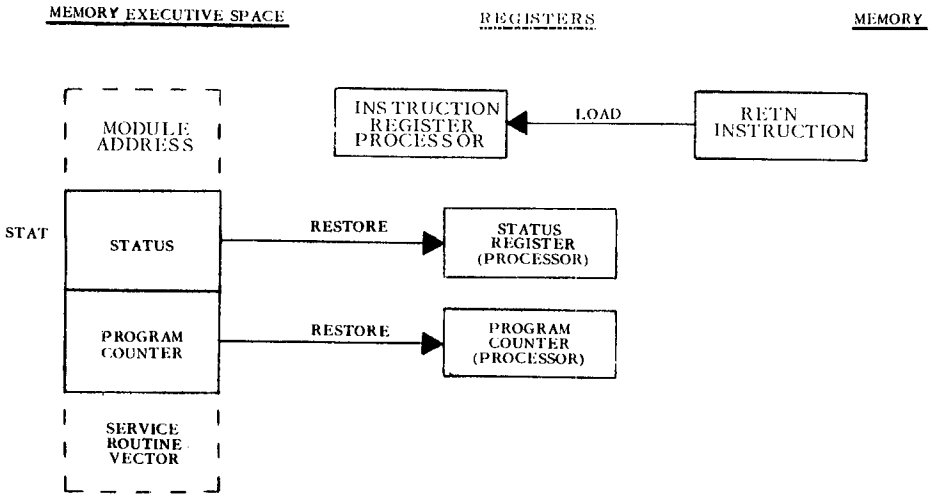


Figure 3-4B. Interrupt Recovery Sequence

SELD — Select Module For DDT.

SELD has one source (the Infibus Controller) and as many loads as there are master device modules in the system. It is used along with the precedence signal to select the device for bus access.

PCDA/B — Precedence.

PCDA/B has one source (the Infibus Controller) and is propagated right to left by each device module if the device is not requesting bus access on the selected level.

SACK — Select Acknowledge.

SACK has one load (the Infibus Controller) and as many sources as there are master device modules in the system. It is used by the device to signal recognition to the Infibus Controller that it has been selected for the next bus service cycle.

Three signals are used to monitor cycle timing (STRB, DONE, QUIT).

STRB — Strobe.

STRB is bidirectional. It can have both a source and a load in every system module. When asserted it signals that the Infibus is busy in a service cycle, and when removed, indicates Infibus availability to a selected device.

DONE — Done.

DONE is bidirectional. It can have both a source and a load in every system module. It is used by the slave module to signal that the bus service cycle is completed.

QUIT — Quit.

This signal has one source, the Infibus Controller, and as many loads as there are master modules. It is used to signal the master module that the slave module did not signal DONE within two microseconds of the start of the cycle. A non-processor module records this as a cycle-abort status and sends an interrupt signal to the processor. A processor module initiates a self-interrupt sequence.

Three control signals (RITE, HCYC, BYTE) are used to command register or memory read and write operations. A set of 16 lines is used for address transmission and another set of 16 lines for data transmission.

RITE, HCYC, BYTE Control Signals.

These signals are bidirectional and are used by the selected master module to command a read or write function within the slave module. RITE when asserted specifies a write function. HCYC when asserted specifies a half cycle memory function. BYTE when asserted specifies a byte only transfer, bit DB00-07. Combination functions of these three signals are in Table 3-3.

Table 3-3. Control Signal Combinations

Write RITE	Half Cycle HCYC	Byte BYTE	Function
1	1	1	Write Only, Byte*
1	1	0	Write Only, Word*
1	0	1	Clear-Write, Byte
1	0	0	Clear-Write, Word
0	1	1	Read Clear, Byte
0	1	0	Read Clear, Word
0	0	1	Read and Restore, Byte
0	0	0	Read and Restore, Word

* A Read Clear must occur first on core memory.

When registers are addressed, a write control signal causes the slave module to load the addressed register from the Infibus. A read control signal causes the slave module to transmit the contents of the addressed register on the Infibus.

AB15-AB00 Address Lines.

These lines are bidirectional. They may have a source at any master module that can request bus service. Each line will have a load at every slave module with an addressable function. AB15 is the most significant address line and AB00 is the least significant. AB00, when asserted, specifies the right-most byte (odd numbered); when removed specifies the left-most byte (even numbered) of a 16-bit word.

DDB15-DB00 Data Lines.

These lines are bidirectional. They may have a source and a load at all addressable system modules. All data

transfers between system modules utilize these lines. DB15 is the most significant data line; DB00 is the least significant. Single-byte transfers use lines DB07-DB00 only, the right most byte.

Processor signals:

SRLC — Service Request Line, Computer.

SRLC has one load (the Infibus Controller) and as many sources as there are processors on the Infibus. This signal is used by processors to request bus access.

SELC — Select Computer.

SELC has one source (the Infibus Controller) and as many loads as there are processors on the Infibus. (Up to four are allowed.) The signal is used to select the highest precedence processor that is requesting access.

Interrupt signals:

SRL1, SRL2, SRL3, SRL4 Service Request Lines, Interrupt.

These lines have sources from each system module that can generate a system interrupt. Each line has one load in the Infibus Controller.

SEL1, SEL2, SEL3, SEL4 Select, Interrupt.

Each line has one source, the Infibus Controller, and as many loads as there are system modules that can generate a system interrupt on a defined level.

System Control signals:

There are eleven lines in the Infibus that are used for special system functions, which are related to switch functions that control system reset, autoload and power line interrupts.

REPB — Reset Push Button.

REPB is asserted by Control Panel module logic. It is sensed by the Infibus Controller which then transmits a Master Reset signal, MRES.

MRES — Master Reset.

MRES is asserted by the Infibus Controller when it detects the Panel Reset Push Button is pressed, power initiation (recovery) or power failure signals. It is used by all system modules to reset to an initial status condition. When MRES is activated due to a power failure status (PWST), the signal is delayed 2.2 milliseconds after detection.

PFIN — Power Failure Interrupt Inhibit.

PFIN is asserted by special switch located behind the Control Panel. It is used by the Infibus Controller to inhibit generation of an interrupt service request on level four when the power status line, PWST, indicates power failure.

PWST — Power Status.

PWST is asserted by the power supply to indicate regulated power is available. Power supply logic will remove this signal 3.3 milliseconds before loss of regulation due to power failure. Only the Infibus Controller monitors this signal. If the function is not inhibited by PFIN, the Infibus Controller may initiate a level-four interrupt.

LFIN — Line Frequency Interrupt Inhibit.

LFIN is asserted by a special switch located behind the Control Panel. It is used by the Infibus Controller to inhibit generation of an interrupt service request on level four when the line-frequency signal, LFRQ, is received from the power supply.

LFRQ — Line Frequency.

LFRQ is asserted by the power supply once each line frequency cycle (normally 60 cycles per second). Only the Infibus Controller monitors this line. If the function is not inhibited by LFIN, the Infibus Controller will initiate a level four interrupt.

PRAL — Power Restart Auto Load Enable.

PRAL is asserted by a switch located behind the Control Panel. It allows Auto Load to occur automatically upon power initialization or recovery.

PRIN — Power Restart Interrupt Inhibit.

PRIN is asserted by a switch located behind the Control Panel. It is used by the Infibus Controller to inhibit generation of a level-four interrupt whenever PWST indicates power on, upon recover.

ATLD — Auto Load.

ATLD is a pulse generated by the Control Panel logic when the Auto Load switch is pressed or at power recovery or initiation if enabled by PRAL. The Auto Load Module is initiated by this pulse.

MINH — Master Interrupt Inhibit.

MINH is asserted by Control Panel logic when the Master Interrupt Inhibit switch is pressed. The signal, which is monitored by the Infibus Controller, will inhibit all system interrupts. Processor self-interrupts are never inhibited.

EXAT — External Attention.

Assertion of this line will cause the Infibus Controller to initiate a level-one interrupt.

Block Transfer Adapter Lines.

A set of four Infibus lines is used to allow communication between only immediately adjacent modules. These lines are for special communication between a BTA and its slave I/O Controller, and may be used for signals between any two cards in a module.

BT1A, BT1B — Bus Cycle Request.

Each signal is asserted by the slave I/O Controller to signal the BTA that it desires a bus cycle.

BT2A, BT2B — BTA Response.

Each signal is asserted by the BTA to signal the slave I/O Controller that it should either receive data on the next DONE signal when a write control is active, or that it should place data on the Data Lines when a read control is active.

BT3A, BT3B — Controller Error.

Each signal is asserted by the slave I/O Controller to indicate an error condition to the BTA.

BT4A, BT4B Interrupt Request.

Function - Each signal is asserted by the BTA to request a system interrupt. The signal remains negated until the BTA block-length register counts down to zero, the controller error, BT3B, is asserted, or an abort condition occurs during a block transfer. When the BTA is installed but not being used for direct data transfers, bus cycle request, BT1B, is sent back to the slave I/O controller on this line.

Clock and Run Lines:

CLK1 Clock.

CLK1 is generated on the Infibus Controller by a 25-megahertz oscillator. It represents the system clock and can be used by other system modules.

RUNN Run.

RUNN is generated by all processors. It is used by the Control Panel to light an indicator displaying that any processor is in the RUN mode.

Memory Option Lines:

These lines are used when either of the memory option modules Memory Protect or Memory Parity are used. Use of the Memory Parity module also requires use of the optional 4K by 18 memory modules.

KEY0, KEY1 Memory Protect Override.

These lines are asserted by the central processor to allow the CPU to override the Memory Protect function.

HOLD Memory Cycle Inhibit.

HOLD is asserted by the Memory Protect module (when present) until it determines that a memory write is permissible. This line is also used by memory modules that are jumpered to operate in an interlock mode.

PBLO, PBHI Parity Bits.

These lines are asserted by the Memory Parity module (if present) on every memory module write operation to generate an odd parity bit. PBLO is the parity bit for the right byte, DB07-00; PBHI is the parity bit for the left byte, DB15-08. On every memory read operation these lines are checked for correct parity. If a parity error is detected the Memory Parity module will assert a level-four interrupt request.

THE INTERFACES

There are two types of SUE Universal I/O Controllers:

SUE 4501 Parallel Controller (paper tape, line printers, card readers or punches). There are four versions of this controller to allow selection of device interface signal levels.

SUE 4502 Serial Controller (teletypes, RS 232-C devices).

SUE 4590 Block Transfer Adapter (converts 4501 and 4502 to block Controllers).

For block transfer control the SUE 4590 plugs adjacent to an I/O controller for block transfer direct to memory without impairing the processor instruction stream.

SUE provides several custom interface modules for special interface requirements:

Custom Infibus Interface (4550)

The 4550 module provides all the Infibus interface logic necessary for proper timing and drive to interface directly with the Infibus. In addition to the interface logic mounting space is provided for up to 50 16-pin DIP chips for further customized logic. The user implements device logic and connects to the standard Infibus logic.

Universal Logic Board (7980)

The 7980 (ULB) is a universal logic board designed to hold up to 98 14-pin DIP chips. The chips can be socket mounted or soldered directly to the board. Solder pads are provided for staking wire-wrap pins or soldering wire directly to interconnect chips. Standard Infibus interface logic diagrams are provided. The ULB has 110 pins for device interfacing to the back edge connector.

SPECIAL INFIBUS INTERFACE LOGIC

The following section describes in detail the design cirteria for Infibus interfacing. This information is needed only when the user must design a special-device interface.

Custom Interface Techniques

To assure that overall SUE system performance is not impaired a number of design techniques and components are recommended.

A special module designed to be inserted into the Infibus must have the following logical blocks:

Infibus Line Drivers and Receivers
Bus Access and Control Logic
Address Recognition Logic

If the module can generate system interrupt requests it must include Interrupt logic.

If the module has several sources of data that can be transmitted on the data lines it may include a data line multiplexer.

Special interface modules handle data, control, and status information in the same way as the standard controller. A data register is used to buffer transmitted data, a control register is used to receive and hold command bits and a status register is used to hold status bits that indicate the state of the module. In the SUE computer these registers should be directly addressable by processors or other master modules. Special addresses are assigned to these registers so that they can be read or written under program control.

A block diagram of a module with these logical blocks is shown in figure 3-5.

Some special modules may require additional logical blocks. For instance, if direct memory access is desired the special module should contain an address register and a block length counter. These registers also should be addressable for programmed initialization. As an alternative the special module can be designed to operate with the Block Transfer Adapter. This module contains an address and block-length counter and provides direct memory access.

Infibus Line Drivers and Receivers

Most Infibus lines are bidirectional and can be driven by more than one source and received by many devices. These lines are driven by open collector drivers and are asserted at a low logic level. When not activated they are pulled high (+5v) by line termination resistors. Receivers of these lines must present a high impedence to minimize loading.

The SUE product line includes a special MSI component used for Infibus *line driver receivers.* This is a 16-pin dual-in-line package that contains four

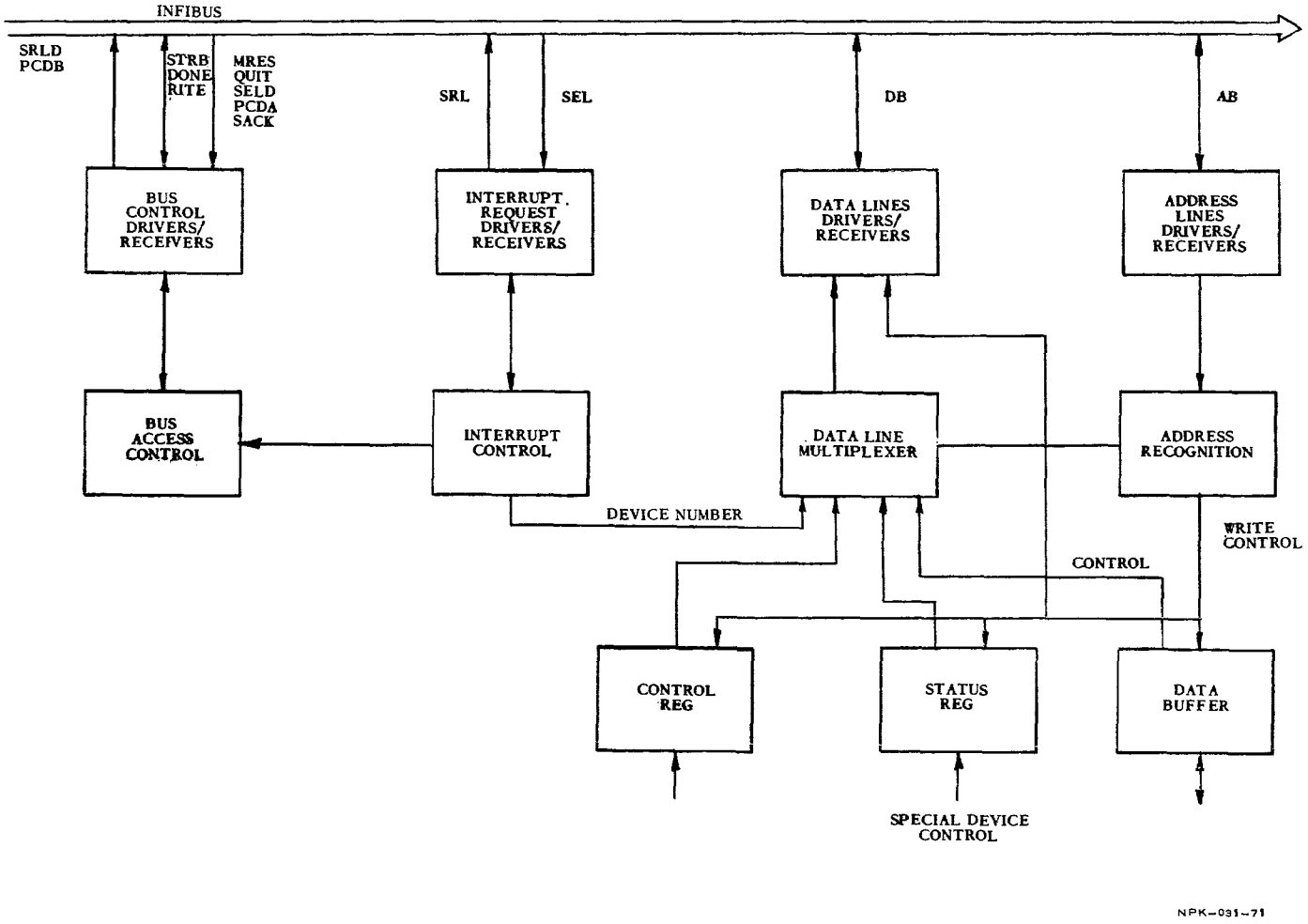


Figure 3-5. Infibus Interface Block Diagram

driver/receiver circuits. Use of this component minimizes printed circuit board space. If this component is not used equivalent specifications regarding logic levels, current sink, loading, capacitance and propagation delays should be satisfied.

Address Recognition Logic

Every module on the Infibus must be able to operate in the slave mode and recognize when it is being addressed. A block of at least eight sixteen-bit addresses is reserved and assigned to the module. Jumper straps on the middle eight bits should be provided to make this block of addresses selectable. This is shown in figure 3-6. The four least-significant bits of the address select the internal registers (bit position 0 is not decoded). The logic should verify that the most significant hexadecimal digit is F. A system module examines the address lines when the strobe signal is asserted on the Infibus. High-speed logic elements should be used to minimize propagation delays. When a module detects its address it will examine the RITE line to determine if a read or write operation is required. If the RITE line is not asserted a register read operation is commanded. The module gates the contents of the selected register onto the Infibus data lines. If the RITE line is asserted the data on the Infibus is written into the selected register. In both cases the module signals DONE to indicate completion of the data transfer.

Module Registers

Normally a special module will contain three addressable registers: a control, status, and data register.

A control register is loaded by a processor instruction to start or stop its functions. This register is not usually more than five or six bits in size. Typical bit assignments may be

Bit 0 = 1 start, 0 stop
Bit 1 = 1 output mode, 0 input mode
Bit 2 = 1 allow interrupts, 0 inhibit interrupts
Bits 3 and up, special command bits to devices

The status register indicates to a processor that the module and device are ready and indicates any detected errors in device operation. It is usually not more than three or four bits in size. Typical bit assignment may be

Status Register:

Bit 0 = 1 data ready, 0 data not ready
Bit 1 = 1 device not ready, 0 device ready
Bit 2 = 1 overrun, 0 no overrun
Bit 3 and up, special

Writing into the status register is generally used as a programmable means of clearing the module.

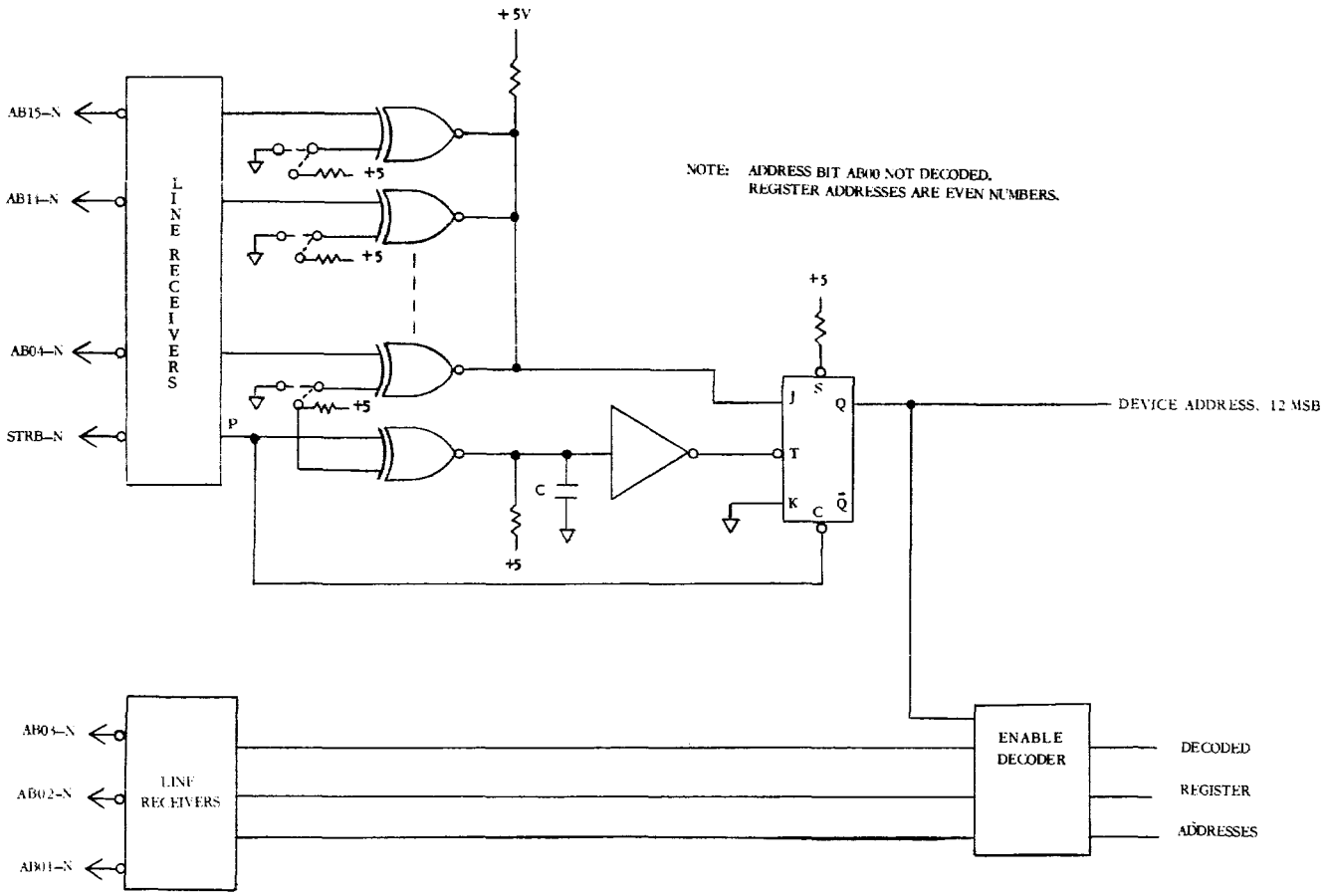


Figure 3-6. Address Recognition

The data register is an 8-bit or 16-bit buffer register for formatting data during input or for holding data to be output. More than one buffer register may be provided if desired.

Interrupt Logic

A special module can be designed to generate an interrupt Service Request signal as described in the Infibus discussion. Any of the four levels can be used; however, it is recommended that only levels two or three be used and that levels one and four be reserved for special system functions. It is recommended that the module be designed to allow or inhibit interrupts under program control. This can be done by designating one of the control register bits. When set the register bit will allow the interrupt request.

The module's interrupt logic requests a bus cycle by asserting a service request line (SRL1-SRL4). Upon being granted access by the Infibus Controller, the module places its device number on the data lines of the Infibus, and after a delay, asserts the strobe line. Data line bit 0 determines whether the interrupt is for input (= 0) or output (= 1).

The device number of a module is the lowest address of the block of addresses assigned to the device. This number should correspond to the address of the status register of the module. This makes programming convenient for interrupt operations when several devices are sharing a common interrupt level.

Bus Access and Control Logic

The design requirements for bus access and bus communication were described previously with the Infibus. The precedence chain signal, PCDA, deserves special attention. PCDA must be propagated serially through each module. This must be accomplished with the minimum possible delay to maintain the Infibus operational speed.

The recommended driver/receiver is a Schottky AND gate, type SN74S11N or equivalent. This is shown in figure 3-7.

Data Line Multiplexer

The Data Line Multiplexer offers a convenient method for gating several sources of data onto the Infibus data lines. Four-to-one or two-to-one multiplex components are available for this purpose. Typical components are types SN74153N or SN74157N or equivalents.

Figure 3-8 represents a logic diagram of a single component, providing two, four-to-one gating functions on a single component.

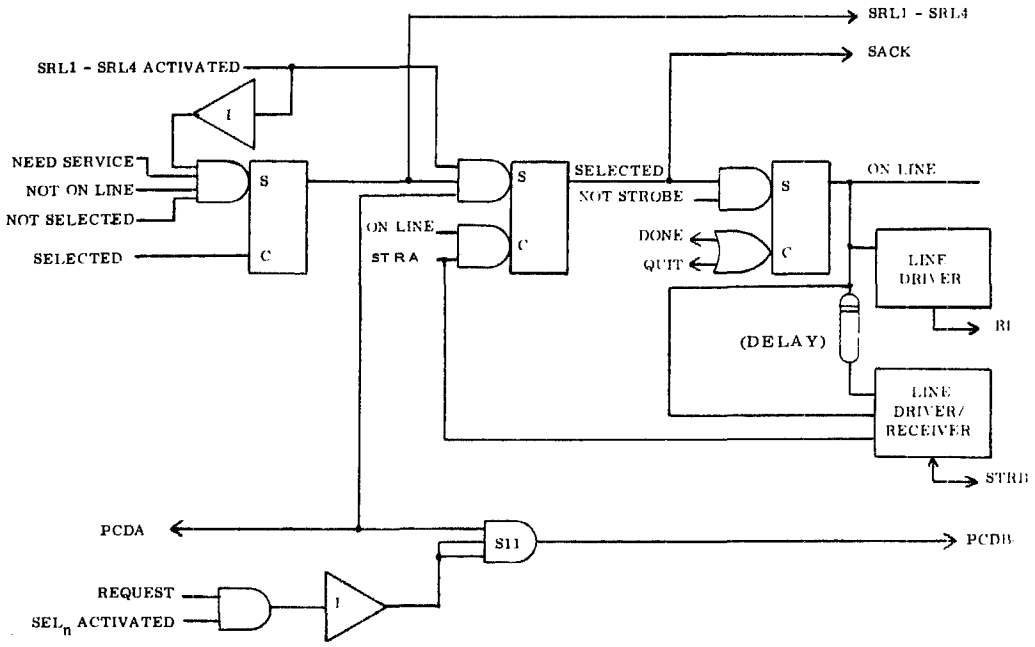


Figure 3-7. Interrupt Logic

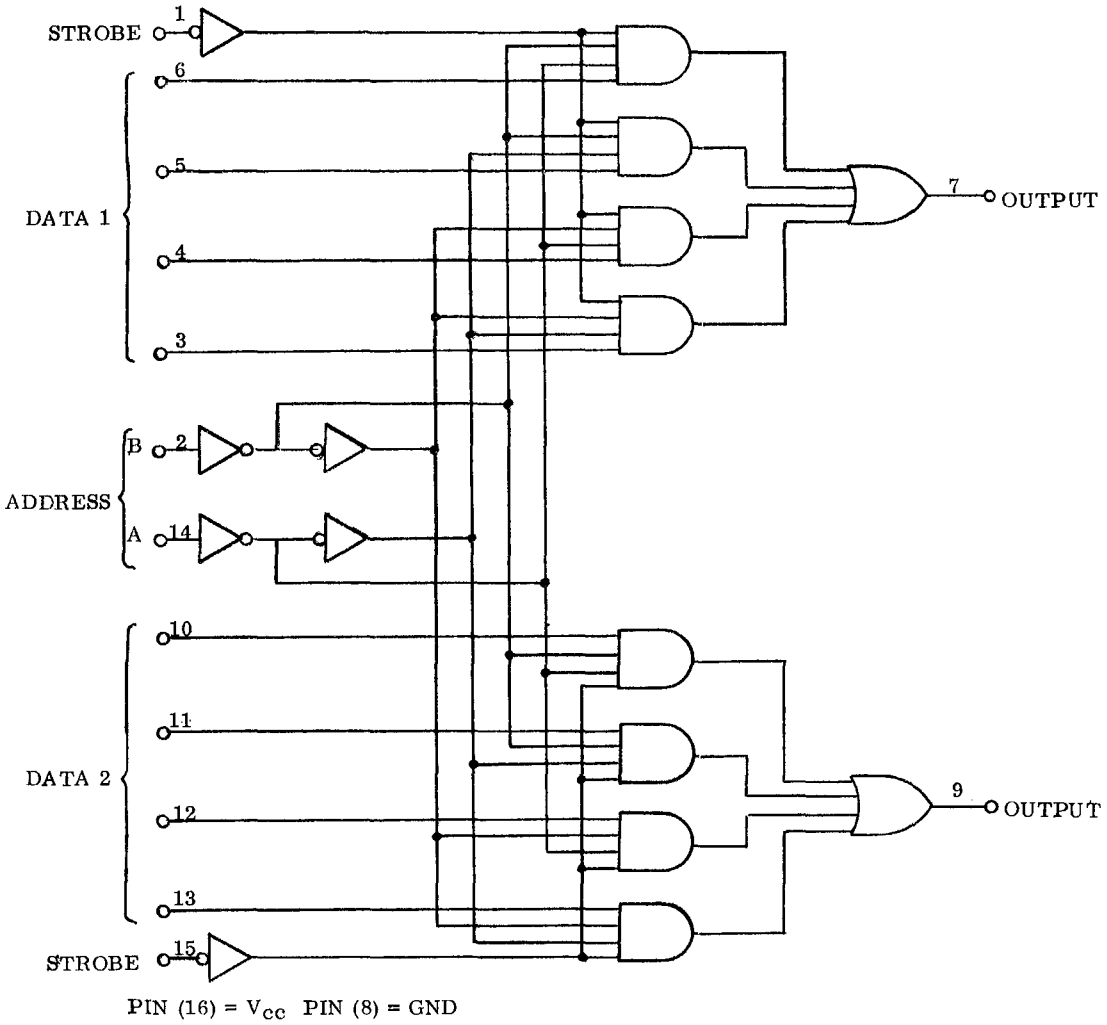


Figure 3-8. Multiplex Logic

CHAPTER 4

SUE PROCESSORS

SUE PROCESSOR OPTIONS

System user engineering, when applied to the processor, means that the user obtains precisely the amount of computer power he needs for his application and has the option of adding processor power in the future by merely plugging more processors (or another type) into the system.

SUE has a family of processors, the Standard, Business and Scientific. Deciding which of three available processors is best for a given task is quite simple since they are all identical except for the size and content of the ROM control store and all contain the same basic instruction set.

SUE processors are microprogrammed with a 36-bit by 256 or 512 word bipolar Read-Only-Memory (ROM) control store. The 60-nanosecond ROM contains the firmware which uniquely defines a SUE instruction set. Instruction sets are changed or extended for special applications by reprogramming the control store and changing the 11 ROM components.

The Standard processor is the Model 1110. Its control store contains the basic instruction set and is used in general purpose applications. The 1110 processor has a 64 mnemonic basic instruction set and is upward compatible with both the 1111 Business processor and the 1112 Scientific processor.

Business applications, or those that require decimal arithmetic instructions, are enhanced by using the Model 1111 processor. The 1111 business processor firmware includes the 1110 instruction set and nine memory to memory decimal arithmetic and character string manipulation instructions.

SUE Model 1112 processor has 36 special instructions for scientific applications, in addition to the 1110 Standard Instruction Set. Special instructions include bit manipulation, double length fast shifts and single precision multiply and divide.

Discussions in this handbook are based on the Model 1110 processor because its instruction set is common to all three. The 1111 and 1112 processor instruction sets are described in Appendix J.

SUE 1110 PROCESSOR FEATURES

16-bit data word

16-bit parallel arithmetic-register unit that processes two register operands in 160 nanoseconds

7 general registers available as accumulators, index registers, address pointers, stack pointers; and one program counter register.

543 instructions

- 492 general register
- 26 branch
- 8 shift
- 17 control

Addressing

- Source and target
- Byte and word
- 64K bytes
- Relative and absolute
- Direct
  - Indexed
  - Indexed, auto-increment
  - Indexed, auto-decrement
- Indirect through index
  - Indirect through index, auto-increment
  - Indirect through index, auto-decrement
- Multilevel indirect

Fast bit programmable shifts, multiple positions

12 branch conditions: either true or false

Look-a-head instruction fetch

All general register save/restore instruction

Loop complete status indicator

Arithmetic and logical instructions, nondestructive compare and test

Generalized push-down, pop-up stack processing with no restrictions on register usage or size of stack

Automatic testing of every arithmetic, logical, and move instruction for zero, negative, and odd without a separate test instruction.

Auto-increment and auto-decrement in both word and byte modes

Concurrent processing with direct memory transfers

Priority interrupts, 4 levels standard with unlimited sharing on levels

Reentrant interrupt handling

Automatic save and restore of program counter and status combined with interrupt vectors

5.58-microsecond response (850 nanosecond core memory)

4.26-microsecond return (850 nanosecond core memory)

Unimplemented instruction trapping

Selective interrupt inhibiting

Multiprocessor configuration capability

PHYSICAL CHARACTERISTICS

The 1110 processor unit is packaged on two standard SUE printed circuit cards. These cards are 6-1/4 by 13-1/2 inches and are multilayer. Internal layers are ground and power planes providing a module that is insensitive to electrical noise. This module is assembled with the most advanced semiconductor (TTL) components that are available from two or more sources. Over 90 percent of the logic gates are on MSI or LSI dual-in-line packages.

One Processor card contains the arithmetic-register unit, the other card contains the ROM control memory. A small plug-on card interconnects the two cards on their free edge.

PROCESSOR ORGANIZATION

The arithmetic unit processes two 16-bit operands in parallel. It contains a Register File of twelve 16-bit registers and six additional registers. They are organized around a high speed, parallel arithmetic-logic-unit, ALU, that logically

processes two 16-bit operands within 130 nanoseconds, or arithmetically within 160 nanoseconds. Eight of the registers in the file are addressable by programmed instructions. Seven are used as accumulators or index registers and are called General Registers. The eighth register is used for the Program Counter.

The SUE Processor (figure 4-1) is designed to operate independent of other system modules and to minimize its utilization of the Infibus. It contains its own internal buses for inter-register transfers. The processor handles data that is received from system memory modules or directly from Input/Output controllers. It does this under control of stored programs or externally provided 16-or 32-bit instructions. Automatic priority interrupt response saves and restores the Program Conter and Status register for any of six levels in executive space in memory. The interrupting device number is also saved prior to the processor's automatic jump to the prestored interrupt vector location of the interrupting level. Response time is 5.58 microseconds and recovery is 4.26 microseconds (response times with 850-nanosecond core memory).

Infibus Interface

The processor can operate in either the master or slave mode. Therefore, the Infibus interface logic contains bus access and control logic as well as address recognition logic. The processor may request a bus cycle from the Infibus Controller and may command a read or write operation from any other system module that can operate in the slave mode.

When the processor is halted, all of the processor's internal registers can be addressed by other system modules that operate in a master mode. Therefore the Infibus interface logic must recognize when one of these registers is being addressed and respond by receiving or transmitting the selected register's contents. Addressable processor registers include the seven General Registers, Program Counter, Status, Instruction and two firmware registers.

The Infibus interface includes three 16-bit registers. These are the Address (A), Receive (R), and Transmit (T) registers. The A register holds the address that is placed on the Infibus when the processor is addressing a system module. The R register receives data from the Infibus, and the T register holds data for transmission on the Infibus. All three of these registers may be gated to the ALU for logical or arithmetic combinations with the selected register of the Register File.

Arithmetic-Register Unit

This unit consists of the ALU, the Register File and a Multiplex Unit, MUX. The MUX is used to select an operand from one of three registers or a 16-bit input separated into four 4-bit fields. Each field is individually enabled or disabled as an input to the ALU under microcode control.

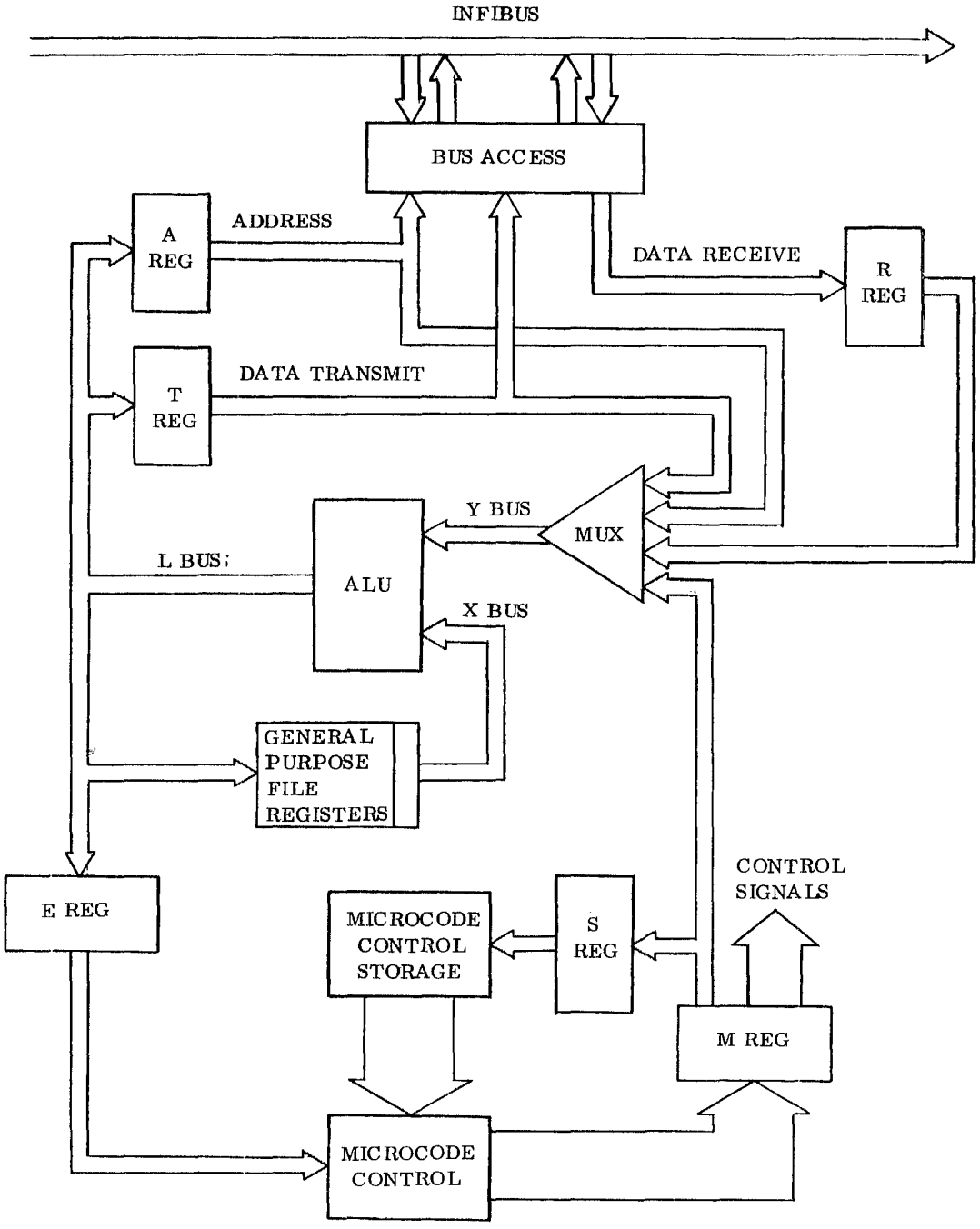


Figure 4-1. SUE Processor Unit Organization

The ALU responds to a command from the microcode word. These commands specify one of sixteen logical functions or one of sixteen arithmetic functions to be performed by the ALU.

The functions are performed on the two ALU inputs, X and Y, and an arithmetic carry signal C, that adds to the least-significant-bit position.

The output of the ALU is via the L Bus that provides the path to write the results of the microstep into the processor's A, T and E registers or Register File.

The Register File consists of twelve similar 16-bit registers. It is constructed from twelve single MSI 4 x 4 flip-flop arrays. One of the twelve registers is selected by the microcode for input to the ALU via its X input. The twelve registers consist of the Program Counter, seven General registers, Status register, Instruction register and two registers: File A and File B that are used internally for microprogrammed execution of instructions.

ROM Microcode Control

This portion of the processor consists of a 256-word by 36-bit ROM Control Storage. It is constructed from nine, bipolar, LSI, ROM components, organized in 256 x 4 bits each.

The microcode control uses three registers. The E register holds the 16- bit programmed instruction that was received from a Program Memory or other external source. Fields of this register specify the instruction code, addressing mode, General register, Index register, and occasionally a literal operand. The S register is an eight-bit counter (expandable to 10 bits) that sequences microsteps. It addresses the Control Storage and is under control of the microcode. The M register holds the 36-bit microcode that specifies action of the current microstep as well as control of the next sequential step.

Microcode Word

The SUE Processor uses a wide polyphase and semihorizontal microcode word of 36-bits. The word format contains 13 fields for specification of control functions within the processor and selection of operands. This large word size for microprogram control allows a number of useful functions to be specified in one microstep. This capability minimizes the number of microsteps that are needed to execute a single instruction. The result is high speed instruction execution with the flexability of microprogram control.

Another feature of the microprogram control is its ability to access the next microstep while in process of executing the current step. This look-ahead feature further minimizes the time for instruction execution. Microsteps are

normally sequenced at a 7.69 megahertz rate, 130 nanoseconds per step. On those microsteps that require an arithmetic function in the ALU, a step is given 160 nanoseconds to assure worst-case carry propagation.

The processor operates asynchronously with other system modules. It is given microcode ability to wait within a microstep and to test for an external signal that signifies completion of an asynchronous event, such as a memory read operation. Therefore the instruction execution rate can be increased with higher speed memory modules.

Microcode Format

The fields of the microcode word are related to the organization and functions of the processor. A variety of special condition test and skip or branch microsteps are provided to allow conditional coding to minimize microsteps.

The microcode word format represents a three address micro-instruction. The X and Y fields each specify an operand, and the W field specifies the destination for the result of the operation. The S field selects the micro-instruction class. The T, A and C fields specify the operation code of the arithmetic-logic-unit. The M, L_2 and L_1 fields are used for either branch or jump addresses within the microstore, for literal operands, or special commands for housekeeping functions. The D, F and Z fields are used to provide preparation for the next micro-instruction.

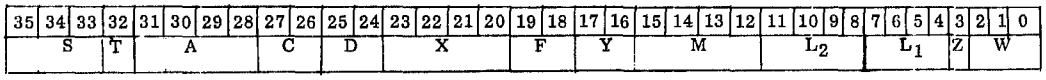


Figure 4-2. Microcode Field Format

The functions of these fields are:

S Field, Microcode Type, specifies the type of microcommand. The type may be a normal sequential step, a special command, a branch or jump command.

T, A Fields, ALU Code, selects one of 16 logical functions if T = 0 or one of sixteen arithmetic functions if T = 1, to be performed by the ALU.

C Field, Carry Control, specifies the options of adding the Carry In and the source of Carry In to the ALU and also controls setting of Carry and Overflow flip-flops.

D Field, Do Next Order Control, selects the option of skipping or not skipping the next micro-order if the ALU's output is or is not all ones or if the jump condition is true or false.

X Field, Register File, selects one of 12 registers in the Register File for ALU input and/ or to receive the output of the ALU.

F Field, Next Order X Field, selects the source of the selection bits for the next access to the Register File.

Y Field, Y Input to ALU, selects the R, A or T register or the Literal Fields L_2, L_1 (under control of the M field) of the microcode for input to the ALU.

M Field, Multifunction, operates with other fields for different functions. When Y = 3 it controls mapping of the literal L_2 and L_1 fields to the ALU's Y input. When S = 2 it is part of the branch address. When S = 4 or 5 it specifies conditional jump codes. When S = 6 or 7 it specifies a bit position in the T register to be tested.

L_2 Field, Literal Field 2, when Y = 3 is a four-bit literal that may be gated to the ALU's Y input. When S = 1 it is a special command. For other values of S (other than 0 or 1) it represents the most significant bits of Control Storage branch or jump addresses.

L_1 Field, Literal Field 1, has two functions when Z = 0. When Y = 3 it is a four-bit literal that may be gated to the ALU's Y input. It may also represent the least significant bits of branch or jump addresses. When the Z field is a one, L_1 specifies bits to be used to generate a special literal source.

Z Field, Special Literal Enable, is used to specify special interpretation of the L_2 and L_1 fields to select a field from the E register or other sources for generation of special literals.

W Field, Write ALU's Output, selects the Register File specified in the X field and/or one of the A, T, E registers or loop counter to receive the ALU's output. For W = 5, 6, or 7 both the Register File and the A register, T register, or E register are respectively selected to receive the ALU's output.

CHAPTER 5

MECHANICAL AND ELECTRICAL DATA

SYSTEM FLEXIBILITY

SUE hardware provides the maximum in system simplicity, flexibility and modularity.

System modules such as an internal power supply, processor unit, Infibus Controller, memory modules and I/O controllers are accommodated in the basic 16-slot chassis.

Further expansion is provided by the expansion chassis that contains an additional 16 or 24 slots. Increased power consumption of an expanded system can be accommodated by the heavy duty external power supply. A 24-slot chassis can be used for the basic system with external power supply.

SUE Infibus operation is completely asynchronous and allows system expansion. Extension of the Infibus to an additional chassis is accomplished by a plug-in Infibus Extender module. The Infibus Extender provides signal synchronization, cable termination, and amplified logic drive necessary for system expansion.

Most system modules may occupy any slot position in the Infibus. One restriction is that empty slots are not permitted between modules. Module precedence is determined by customer preference and is established by physical order on the Infibus. System modules may be added, deleted, or exchanged with minimum effort.

Peripheral device controller modules are connected to the device via free-edge (opposite end from the Infibus) connectors and standard I/O cables. This greatly simplifies system integration and allows convenient configuration change.

SYSTEM CONFIGURATION ELEMENTS

SUE is available in a table-top, rack-mounted or desk console system configuration. Rack mounting is provided by standard system cabinets or customer-furnished EIA standard 19-inch cabinet racks.

Chassis Assemblies

Two SUE basic chassis are available. Model 7910 chassis contains an INFIBUS with 16 card slots and the Model 7911 contains an Infibus with 24 slots. These two basic chassis may be mixed to achieve the desired system expansion.

7910 Chassis Assembly

Approximate Weight: empty chassis: 15 pounds

Cooling - attachable 1-3/4-inch-high fan pack assembly mounts to underside of basic chassis.

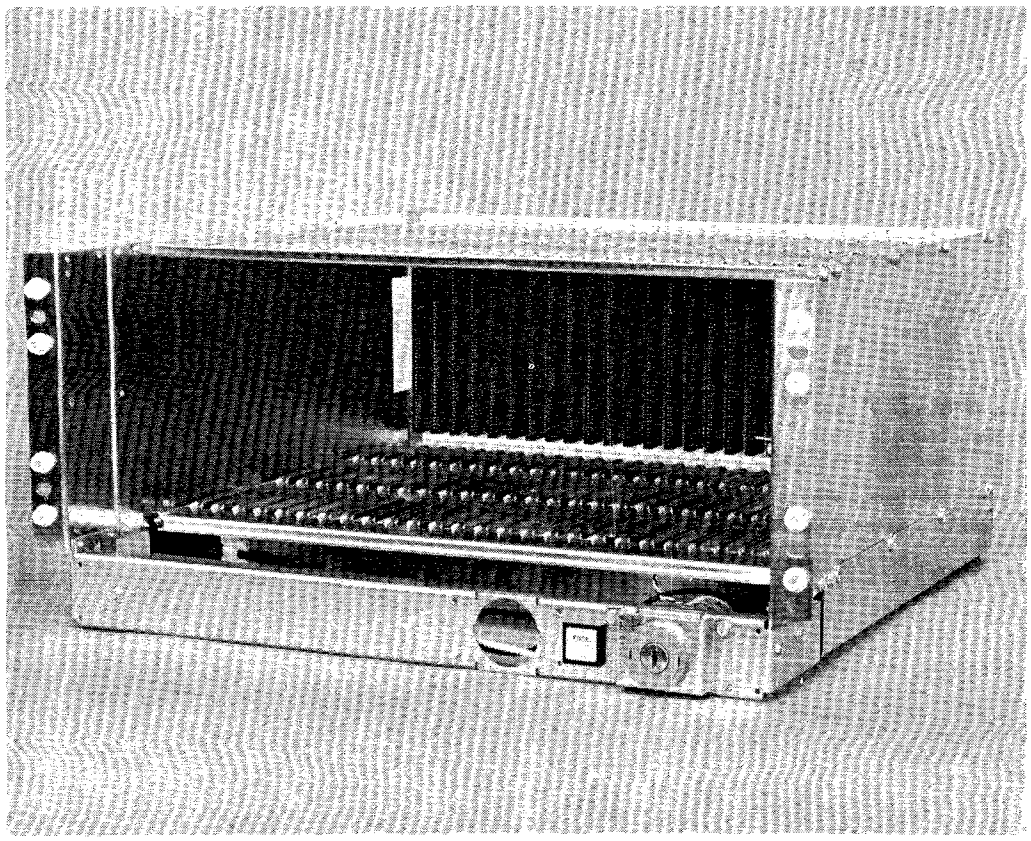


Figure 5-1. Basic SUE Chassis, Model 7910, including 2201 power distribution option

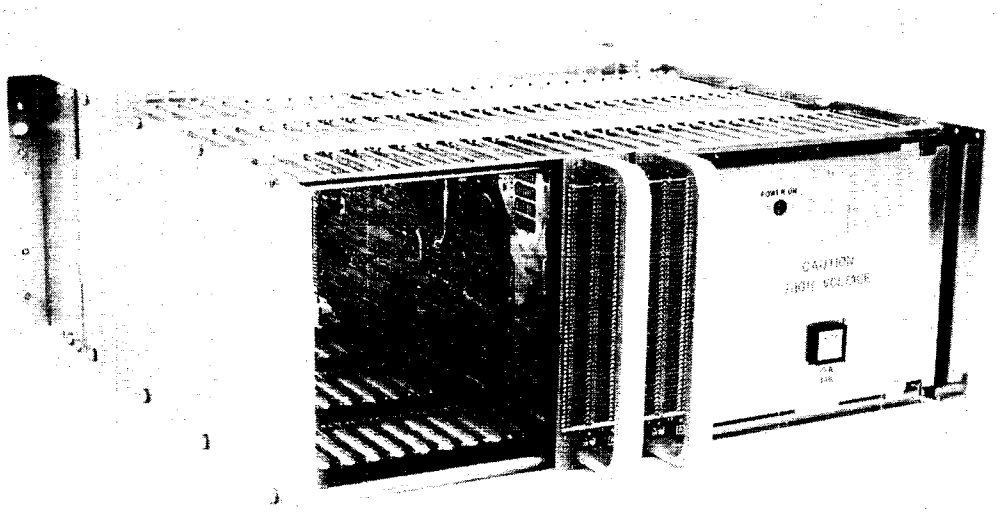


Figure 5-2. A Typical SUE with Processor Unit, 3311 4K Memory and Internal Power Supply Installed in a 16-Slot Chassis

7911 Chassis Assembly. Model 7911 chassis provides 24 card slots and incorporates all the features shown in Figure 5-1. The 7911 must use external power supply model 5952.

The power supply interconnect is installed on the right side of the Infibus, leaving 24 slots.

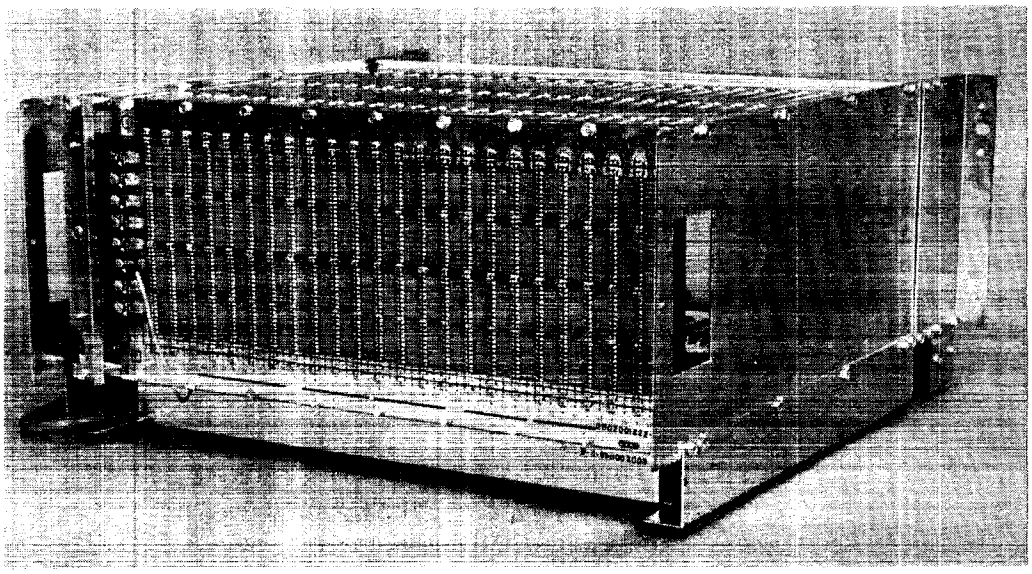


Figure 5-3. SUE Expansion Chassis Model 7911

Power Supplies

Two styles of power supplies are available for SUE. Models 5951 and 5955 can be mounted in the chassis and the larger Model 5952 rack mounts externally to the system chassis.

5951, 5955 Power Supplies. The internal power supply is normally mounted in the 7910 chassis (figure 5-4). This power supply can also be connected to a 7911 chassis with a 7720 cable and externally mounted by the user as dictated by module space and d-c current requirements.

Approximate Size: 6.25 inches high by 7.0 inches wide by 14.25 inches deep

DC Current Specs:		+15 vdc	+5 vdc	-15 vdc
	5951	7 amperes	38 amperes	3 amperes
	5955	5 amperes	18 amperes	1 ampere
Regulation:		±1%	±2%	±1%

AC Power Specs: 105 to 125 vac @ 9 amperes maximum, 47 to 63 Hz

Temperature:

Operation	0°C to +55°C
Storage	-40°C to +75°C

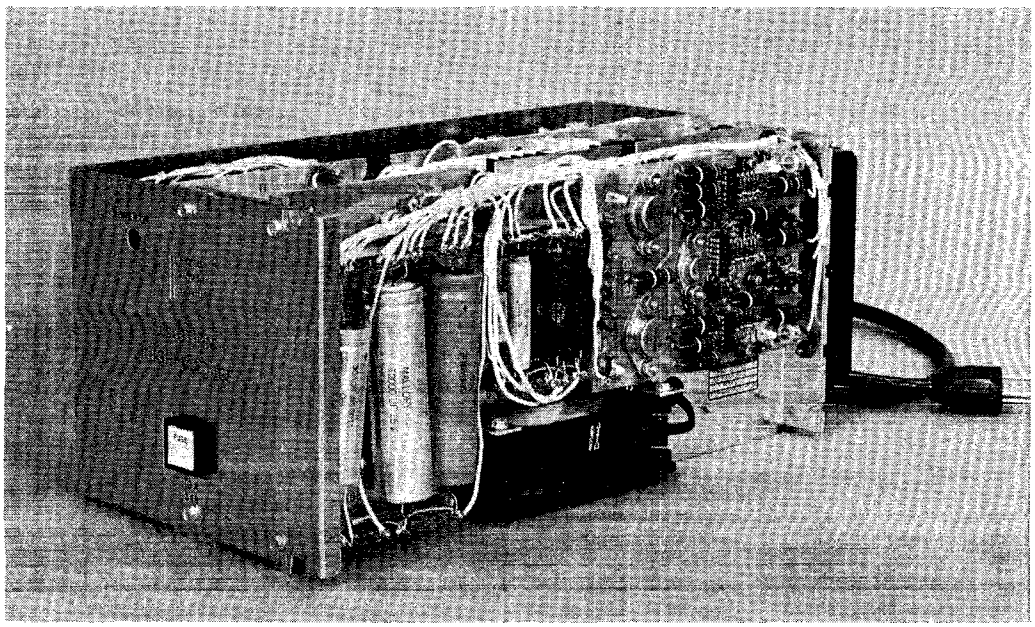


Figure 5-4. Internal Power Supply

5952 Power Supply. The external power supply is used primarily with the 7911 chassis when system d-c power requirements exceed the capacity of the basic 5950 supply (figure 5-5).

Approximate Size:	7 inches high by 19 inches wide by 18 inches deep
Weight	50 pounds
Cabling:	Power interface bus cable to basic SUE chassis (7910 or 7911)
DC Power Specifications:	+15.0 vdc @ 25 amperes, ±1% regulation +5.0 vdc @ 50 amperes, ±2% regulation -15.0 vdc @ 6 amperes, ±1% regulation
AC Power Specifications:	105 to 125 vac @ 20 amperes maximum, 47 to 63 Hz
Temperature-Operation:	0^{o}C to $+55^{o}$C
Storage:	-40^{o}C to $+75^{o}$C

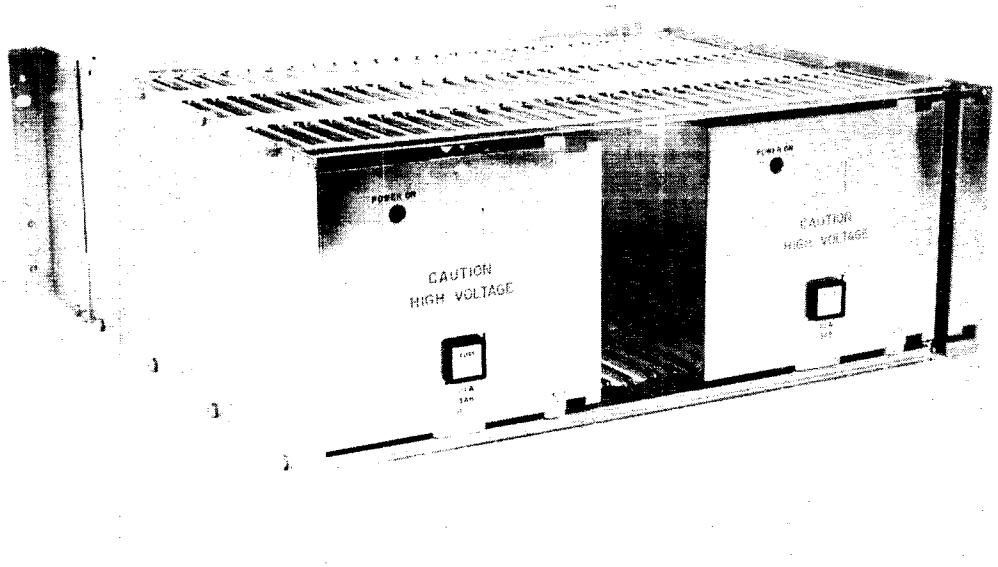


Figure 5-5. External Power Supply (5952)

User Supplied DC Power. The system user may supply power in place of SUE power supplies provided the following specifications are met:

Voltages	+15.0 vdc	+5.0 vdc	-15.0 vdc
Output Currents (Max)	25 amperes	50 amperes	6 amperes
Regulation Bands	±1%	±2%	±1%
P-P Ripple (Max)	60 mv	50 mv	60 mv
Max. Transients	0.4v over/ undershoot for 10%, 100% -10% pulsed load	0.2v over/ undershoot for 80%, 100% -80% pulsed load	0.4 over/ undershoot for 10%, 100% -10% pulsed load
Max. Recovery Times	5 milliseconds	5 milliseconds	5 milliseconds
Overvoltage Protection	+19.0 v	+7.0 v	-19.0 v

MECHANICAL CONFIGURATIONS

Following are several mechanical configurations of SUE chassis described above. These are illustrated to suggest a few of the possible system arrangements (figure 5-6 through 5-12).

Minimum System

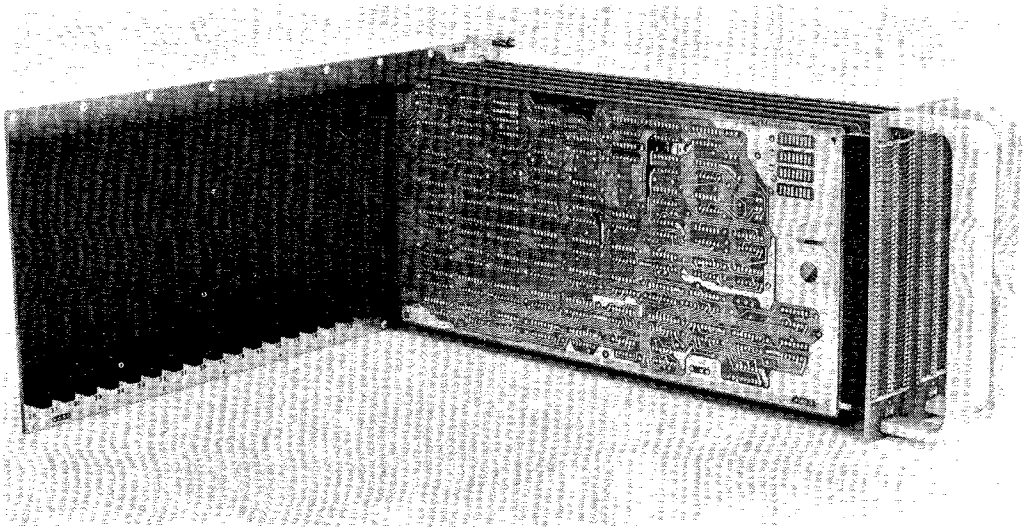


Figure 5-6. SUE As A "Bikini-Mini"

Mixed Chassis

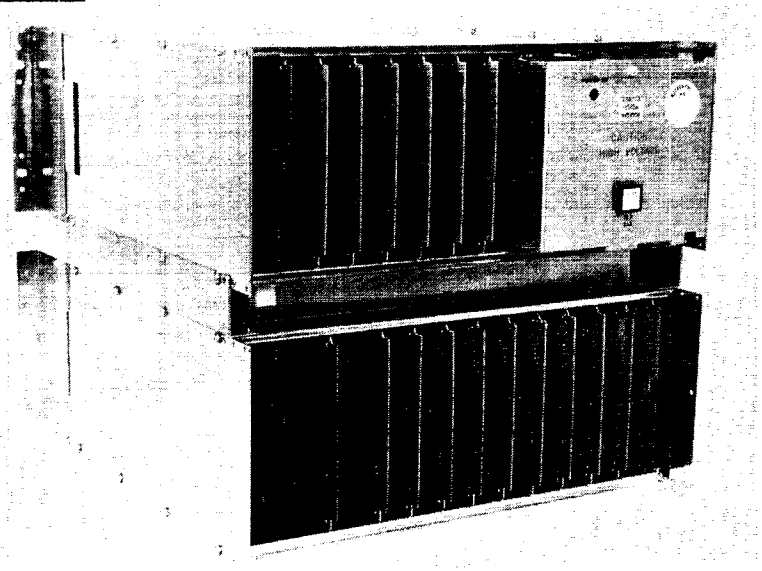


Figure 5-7. Combined 7910 and 7911 Basic SUE Chassis with Internal Power Supply

Multiple 7911 Chassis

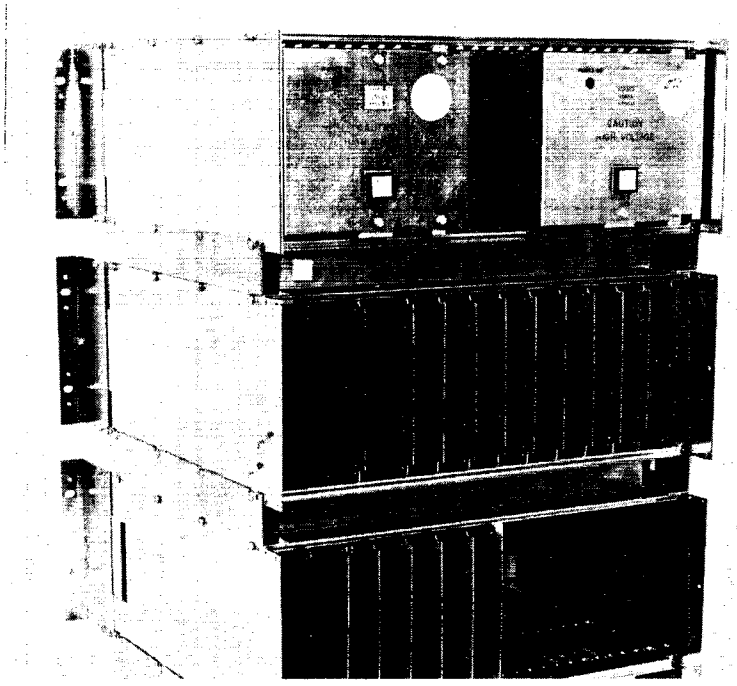


Figure 5-8. Two 7911 Chassis with 5952 Power Supply

Note: Placement of cards is for illustration only.

Standard SUE

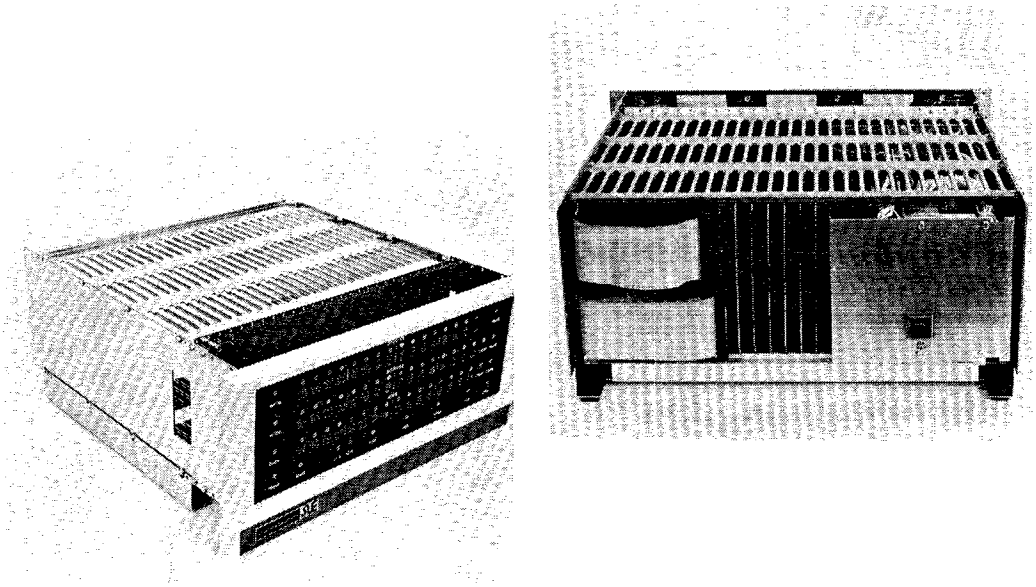


Figure 5-9. SUE Computer Chassis

Expanded SUE System

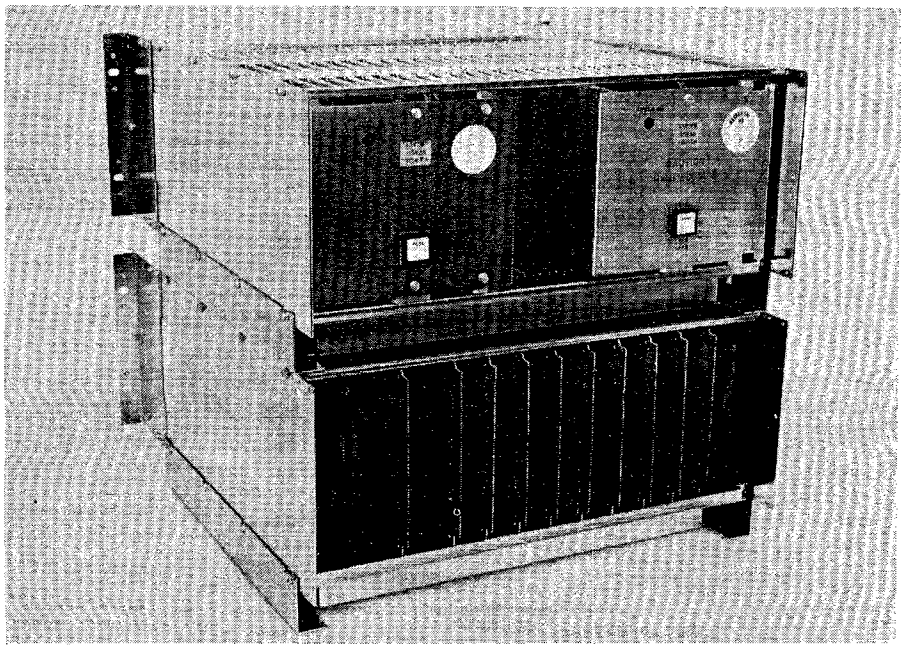


Figure 5-10. Basic SUE System Using 7911 Basic Chassis and 5952 Power Supply

Note: Placement of cards is for illustration only.

Table Top Cabinet

The system can be installed in a table-top cabinet (figure 5-11).

Approximate size: 11-1/2-inches high by 20-inches wide by 22-inches deep

Approximate weight: 25 pounds

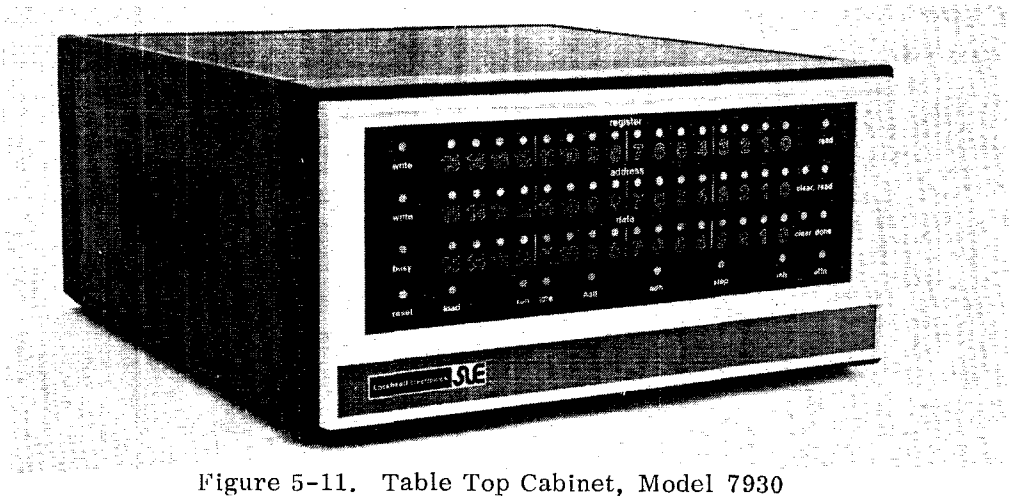


Figure 5-11. Table Top Cabinet, Model 7930

System Equipment Cabinets

Three individual chassis may be mounted in the SUE Free-standing equipment cabinet.

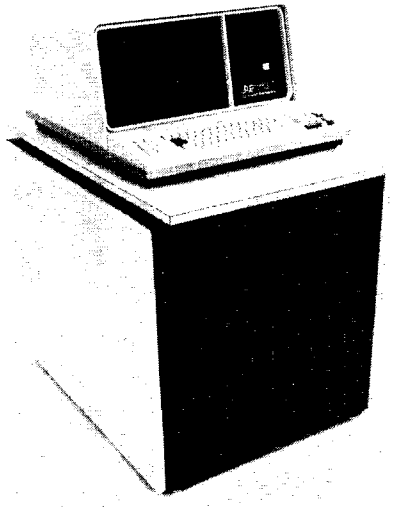


Figure 5-12. Free-Standing System Equipment Cabinet, Model 7932 lo-boy style with optional CRT

CUSTOM SYSTEM MODULES

Infibus Interface. Interface to the Infibus is via a 110-pin connector. Electronic handshake and logic level interface were described in Chapter 3. Each device requests bus access from the Infibus Controller. When permitted the device can communicate directly with any other device on the bus. The requesting device must present to the bus the proper sequence of control signals and address of the target device. In this fashion any I/O device or a processor unit can have direct access to memory or to other devices.

Special interface design can be implemented on Universal Logic Board, Model 7980. This module is furnished as a drilled printed circuit board with universal spacing for up to 96 14-pin DIP chips. A kit of 1,000 wire wrap pins is available.

To simplify special interface design the user can select the common Infibus interface system module, Model 4550, consisting of Infibus interface circuits and a standardized and simplified customer interface including only basic control, data and addressing lines. The 4550 module provides mounting space for up to 50 DIP chips for further modification of the simplified interface to meet external device requirements. Wirewrap pins are available.

Most applications requiring custom interface to parallel data devices can use custom configured standard parallel I/O controller Models 4501, 4503, 4506, 4507.

ENVIRONMENTAL REQUIREMENTS

Two fan pack assemblies, Models 7921 and 7922 are available to provide system module cooling. The fan packs are attachable assemblies that mount to the underside of the 7910 or 7911 chassis and add 1.75 inches to the height of the chassis. Model 7921 provides 420 CFM airflow and Model 7922 quiet fan assembly provides 300 CFM. Normal installation requires one fan pack for each 7910 or 7911 chassis. User furnished airflow must maintain a minimum of 420 CFM at air exit temperature of 132°F (55°C) or below.

Ambient temperature of the installation can vary between 32°F and 132°F (0°C and 55°C) with no degradation of computer operation. A recommended ambient room temperature between 65°F and 85°F (18°C and 29°C) will insure prolonged system life.

Temperature range for shipping and storing of the system is between -40°F and 158°F (-20°C and 70°C). Prolonged exposure of cabinets and hardware to extreme humidity should be avoided.

AC POWER REQUIREMENTS AND CONTROL

A source of 115 vac (± 10 vac), 60 Hz (47 to 63 Hz) single-phase power is required. Systems with Model 5951 or 5955 power supplies require a standard 15 ampere circuit for operation.

Total power requirements for this type of system are the sum of the peripheral device loads plus the 9 amperes of the power supply. A 15 ampere, 3 prong U-ground power cord is supplied on the rear of Model 5951 or 5955 for connection to either a Model 2201 or 2202 switched power distribution panel or a switched power source. Model 5952 power supplies have two power cords; one is connected to a 20 ampere power source and has 20 ampere, 125 vac, 3-wire grounding, straight blade plug and the other is a power switch cord with a 3 prong U-ground power cord which is normally connected to a Model 2201 or 2202.

SUE peripheral devices and other system equipment may be operated from either switched or unswitched power as appropriate to system performance and protection.

CHAPTER 6

ADDRESSING MODES

The SUE computer has a variety of addressing modes that allow efficient use of time and space in coding programs. Handling both word-oriented and byte-oriented data arrays is provided by a full complement of word/byte operations. SUE instructions are capable of directly addressing the full 64K ($FFFF_{16}$) byte addresses. The upper 2K byte addresses are reserved for registers of system modules, leaving 62K bytes of memory for general use. Word addresses are the even-numbered byte addresses . SUE data is fetched from memory on a 16-bit word basis that requires only one memory cycle. In byte operations, the left byte is used when the byte address is even and the right byte is used when the address is odd.

TERMS AND ABBREVIATIONS

The following terms and abbreviations are used throughout this chapter:

A = Address Expression or Direct Address
AR = Accumulator Register
B = Addressing Mode
CA = Computed Address Expression
D = Relative Displacement Factor
K = Constant Expression
L = Literal Data
OP = Operation
PC = Program Counter
R = Register Designator
XR = Index Register
\- = Auto-Decrement
\+ = Auto-Increment
* = Indirect
( ) = Indexed

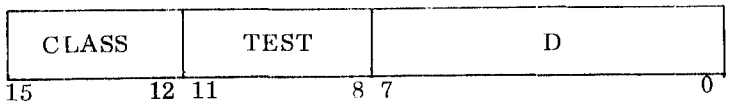
ADDRESSING MODES

Most data in a program or a system is structured in some way - in a table, a stack, a table of addresses, or in a small set of program variables. SUE has specifically designed addressing modes to handle these common data structures. For example: tables require indexing with auto-increment or auto-decrement, stacks require auto-increment/auto-decrement registers used as stack pointers, and tables of addresses require indirect address-indexed with auto-increment or auto-decrement.

Relative and Absolute Addressing

Instruction storage requirements have been optimized by providing single word instructions wherever possible. For example, branch instructions use a relative displacement technique that allows a branch within ±127 words from its current location.

Branch Format



CLASS = Branch Class
TEST = Conditional Branch Test Indicators
D = Relative Displacement -128_{10} to $+127_{10}$ Words

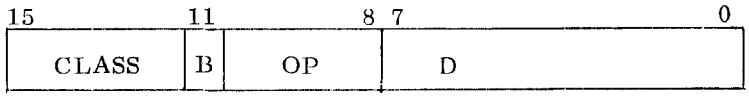
Coding example:

```
BEQT    A
```

The Branch-If-Equal True instruction (BEQT) branches to location A if the equal flag is set; otherwise, program control falls through to the next sequential instruction. The assembler subtracts the current program counter (PC) from the address A, divides the difference by two (for a word calculation), and places the resulting displacement into the D field. If the displacement exceeds a ±127 words from the PC, the assembler or link loader will flag the instruction to notify the programmer of an error.

Branching outside of the displacement range is achieved by using a JUMP instruction. Some instructions in the control group can use either absolute addressing for the first 256 words of memory or addressing relative to the program counter.

Control Format



CLASS = Control Class (0)
B = Relative or Absolute Mode Designator
OP = Control Operation
D = Relative Displacement (-128_{10} to 127_{10} Words) or Absolute Address (first 256 words)

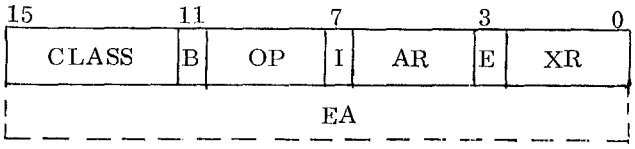
Coding Example:

```
RETN  CA
```

The Return from Interrupt instruction (RETN) transfers the contents of memory word CA into the CPU status register and the contents of CA+2 into the program counter. CA could be the absolute address of the status word in the interrupt storage area of lower memory, or any relative memory location containing the status word. The 256-word area in lower memory provides a system executive storage that the control instructions use (load/store status, load/store general purpose registers, and return from interrupt. Refer to Control Instructions, Chapter 7, and Interrupt Structure, Chapter 3, for futher examples of their use.

GENERAL REGISTER ADDRESSING

General Register Format



CLASS	=	General Register Group (1 through 7)
B	=	Word/Byte Mode
OP	=	Register Operation
I	=	Indirect Address Indicator
AR	=	Accumulator Register Designator
E	=	Extended Address Indicator
XR	=	Index Register Designator
EA	=	Extended Address

OPERATION MNEMONICS (OP):

MOVx	Move
SUBx	Subtract
ADDx	Add
ANDx	Logical Product
IORx	Logical Sum
EORx	Logical Difference
CMPx	Compare
TSTx	Test

Where x = "W" for Word Mode and "B" for Byte Mode

The following addressing modes are associated with the general register classes. The listed addressing mode operands are defined as:

A	Direct Address
A(R)	Direct Address - Indexed
A(R+)	Direct Address - Indexed with Auto-Increment
A(-R)	Direct Address - Indexed with Auto-Decrement
R	Register
(R)	Indexed
(R+)	Indexed with Auto-Increment
(-R)	Indexed with Auto-Decrement
*A	Indirect Address
*A(R)	Indirect Address - Indexed
*A(R+)	Indirect Address - Indexed with Auto-Increment
*A(-R)	Indirect Address - Indexed with Auto-Decrement
=K	Immediate Constant
=K(R)	Immediate Constant plus the Contents of the Register
*(R)	Indirect through Index
*(R+)	Indirect through Index with Auto-Increment
*(-R)	Indirect through Index with Auto-Decrement

General purpose registers (R1-R7) can be used as index registers. Register R0 is reserved for the program counter (PC) and is not used for normal register operations.

Immediate Constant (Single-Word Instruction)

The immediate constant is in the data-to-register class of instructions. It provides the ability to specify a 0_{10} to 15_{10} decimal constant (0_{16}-F_{16}) in a one-word instruction.

```
OP    =K, AR    Data-to-Register
```

The immediate constant, K, operates on the register specified by AR and the result is stored in AR.

Coding Example:

```
ADDW  =6, R4
```

The immediate constant 6 is added to the contents of register R4 with the result stored in R4.

Immediate Constant (Extended Instruction)

OP =K, AR Data-to-Register

In this case K represents a 16-bit signed constant (0_{16}-$FFFF_{16}$) that operates on the accumulator AR.

Coding Example:

SUBW =852, R3

SUBW subtracts the 16-bit constant 852 from the contents of register R3. The result replaces the contents of register R3.

The SUE assembler checks the constant K for greater-than 15 and automatically generates the single or extended instruction.

Register-to-Register Operations (Single-Word Instructions)

The register-to-register operations (Move, Subtract, Add, etc.) operate on the high-speed general purpose registers internal to the CPU without making memory access to fetch operands.

OP RA, RB Register-to-Register

The contents of the register RA operate on the contents of register RB. The result is stored in RB.

Coding Example:

ADDW R3, R5

The contents of register R3 are added to the contents of R5 with the result stored in R5.

Direct Address Memory Reference (Extended Mode-Double Word Instructions)

OP	A, AR	Memory-to-Register
OP	AR, A	Register-to-Memory

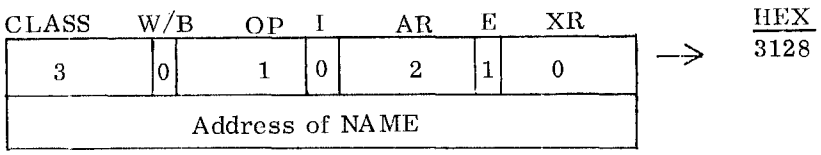
A is the direct address of the memory operand and AR is the accumulator designator. The first operand operates on the second with the result stored in the second operand.

The assembler generates the address A in the word following the instruction and sets the extended bit (E) to indicate that this is an extended address instruction.

Coding Example:

```
SUBW    R2, NAME
```

The instruction subtracts the contents of register 2 from the memory cell NAME. The following code is generated:



The field XR is zero because the direct address A is not indexed. The subtract operation code is 1 and the register-to-memory class is 3.

(R) Indexed

One of the most important coding tools to the programmer is indexing. It provides the offsets and increments necessary for efficient table handling. Index registers may also be used to house a full 16-bit memory address that may be commonly used by many instructions.

(R) Indexed - the entire memory address resides in the specified index register R.

```
OP      (XR), AR      Memory-to-Register
OP      AR, (XR)      Register-to-Memory
```

This example indicates how memory-to-register and register-to-memory operations are achieved in a one-word instruction.

Coding Example:

```
MOVW    (R2), R1
```

This instruction moves the contents of the location whose address is in R2 into the general register R1.

A(R) Direct Address - Indexed

An example of indexing used in table handling operations is: A(R) - Direct Address-Indexed. The contents of the specified register R is algebraically added to the extended address A. The results become the effective memory address.

```
OP    A(XR),AR    Memory-to-Register
OP    AR,A(XR)    Register-to-Memory
```

In these examples, the index register, XR, contains the offset or increment into an ordered data structure. The data can be operated upon in the accumulator AR or in the memory location A plus the contents of XR.

Coding Example:

```
ADDW    R4,TABLE(R7)
```

Assume that register R4 contains the value 100, R7 contains the increment 4 and TABLE resides at location 150.

```
Loc 150     Table       DATA    650
    152         +2      DATA    100
    154         +4      DATA    200 becomes 300
    156         +6      DATA     50
     .           .
     .           .
     .           .
     .           .
     .           .
```

The effective memory address is calculated by adding the contents of R7 to the address of TABLE. 150 + 4 = 154. The contents of R4 is added to the contents of location 154. 200 + 100 = 300. The result (300) replaces the 200 value in location 154.

*A Indirect Addressing

Indirect addressing is denoted by the asterisk (*) symbol in front of the memory reference operand A. An indirect address is one that points to a location containing the effective operand address.

```
Loc 110                MOVW      *A,AR
   .                     .
   .                     .
   .                     .
Loc 260        A       ADDR      350
   .                     .
   .                     .
   .                     .
Loc 350                DATA      1
```

The MOVW instruction moves the contents of the memory word 350, as indirectly addressed by A, to the specified accumulator AR. This operation moves the value 1 into register AR.

Multilevel Indirect

The multilevel indirect capability is available in the word mode only. The normally even-numbered address of a word is made an odd number to indicate another level of indirect addressing. This technique disallows multilevel indirect with byte operations because byte addresses may be odd-numbered addresses. The multilevel indirect feature can be used with all indirect addressing modes of SUE. In each mode the memory word address generated is tested for its least significant bit set; if it is, the address is used as a pointer to the effective address. The process is continued until a fetched address is an even number and it becomes the final effective address of the memory operand. After fourteen levels of indirect address chaining, the processor will issue an unimplemented instruction interrupt.

```
Loc 100                        MOVW     R3,*NAME
                                 .        .
                                 .        .
                                 .        .
Loc 200     NAME               ADDR     211
    202                +2      ADDR     213
    204                +4      ADDR     257
                                 .        .
                                 .        .
                                 .        .
Loc 210                        ADDR     386
Loc 386                        DATA       0
```

The MOVW instruction moves the contents of R3 into memory location 386. The memory operand *NAME is the indirect address 211. Whenever an indirectly accessed address is odd, that is, the least significant bit is set,

it is also an indirect address. The least significant bit is disregarded which makes the effective word address, 210. Memory word 210 is accessed and found to be an even number of 386. The even number terminates the search and 386 becomes the effective address of the memory operand.

*A(XR) Indirect Address - Indexed

OP	*A(XR), AR	Memory-to-Register
OP	AR, *A(XR)	Register-to-Memory

These memory reference instructions allow operations on a selected accumulator AR or on a selected array addressed by the contents of the effective address.

```
Loc 110                 MOVW    =2, R3
    112                 ANDW    R1, *A(R3)
     .                    .
     .                    .
Loc 260       A         ADDR    350
    262                 ADDR    400
    264                 ADDR    450
    266                 ADDR    500
     .                    .
     .                    .
     .                    .
Loc 400                 DATA    100
```

The indirect address is calculated by adding the contents of R3 to the memory address A. 260 + 2 = 262. The content of location 262 is the computed address of the memory operand *A(R3). The content of register R1 is logically ANDed to the contents of location 400. The logical product is moved to location 400.

If R3 is now incremented by 2, the operation would indirectly address the word at location 450, since the third word of table A is used as the effective address.

*(XR) Indirect - Indexed

Indirect-indexed provides the programmer the ability to address memory indirectly through the index register XR. XR contains the address of the memory word containing the effective address of the operand.

(1)	OP	*(XR), AR	Memory-to-Register
(2)	OP	AR, *(XR)	Register-to-Memory

Example (1) indirectly accesses the contents of a memory word to operate on a specified accumulator register AR. Example (2) allows the contents of the specified accumulator AR to operate on a memory word indirectly addressed through the index register XR.

```
CMPW        R5, *(R6)
```

The content of the location specified by R6 is used as the address of the memory operand. The content of R5 is compared to the effective operand.

Auto-Decrement and Auto-Increment Options

To provide convenient stack processing and table searching, the index register may be automatically decremented or incremented whenever a word or byte of data is removed from or appended to a data structure.

Coding Examples:

```
(1)     MOVB    TABLE (-R2), R3
(2)     MOVW    R3, TABLE(R2+)
```

Example (1) shows the move of a byte from the buffer TABLE to register R3. Example (2) illustrates the storing of a word from register R3 to the buffer TABLE as indexed by R2. The auto-decrement and auto-increment of register R2 provide the table address handling along with the data fetch or store in one instruction. R2 is pre-decremented by 1 for a byte operation or post-incremented by 2 for a word operation. These options can be applied to any of the preceding modes which use an index register.

For example:

```
TEST    DATA    H)FFFF
NAME    DATA    H)8AC0
        DATA    H)60F0
        DATA    H)FFFF
        DATA    H)24A0
        DATA    H)11FF
BEGIN   MOVW    =8, R2          Initialize loop count.
        MOVW    TEST, R3        Put test word into R3.
AGAIN   CMPW    R3, NAME(-R2)   Compare test word with
                                  name indexed by R2
                                  after decrement.
        BEQT    OUT             If equal, branch to OUT.
        BLPF    AGAIN           If Loop Complete flag
                                  is false (not set)
                                  continue loop, if not
OUT     HALT                    HALT
```

This routine compares the first four words of NAME with TEST. It exits to OUT if it finds a word equal to TEST or halts if it does not find one. The Loop Complete Test BLPF tests the last general purpose register that was auto-incremented or auto-decremented, for a zero condition. In this example R2 is the loop count register.

After the registers are initialized by starting at location BEGIN, the first location to be compared with R3 will be NAME + (8-2) (location containing H)24A0). Register R2 is predecremented to 6 before the comparison is made. The value in R3 (-1_{10}) is less than the value at NAME +6 (937_{10}) so both the G and E status bits will be reset and BEQT will not branch. Since R2 did not decrement to zero, the BLPF instruction will branch to location AGAIN. The value in R3 is compared with the contents of location NAME +4 in this iteration (R2 is predecremented to 4) and the E status bit is set. The BEQT instruction will branch to location OUT and SUE will halt.

CHAPTER 7

INSTRUCTION SET

This chapter defines the instruction set of the SUE processor. The description includes instruction mnemonic, format, timing, status affected and operation. The timings are basic and require addition of addressing mode times from Appendix A.

NOTATIONS

The following notations will be used in this chapter:

A	direct address
*A	indirect address
Abs.	absolute address
AR	accumulator register
B	address mode
EA	extended address
OP	operation
R	general purpose register
REL	relative address
(S)	source operand
(R)	target operand
XR	index register
.XXX.	logical operation
[ ]	the contents of
=	literal data to follow
+	auto-increment
-	auto-decrement
⟶	becomes

STATUS INDICATORS

SUE 1110 Processor has a 16-bit status indicator register. Status indicators may be affected by execution of General Register and Shift instructions. This is indicated by their symbol in Instruction Descriptions. The status indicators may also be set or reset with special Control instructions.

The status bit position within the Status register, symbol, name, and description are as follows:

Bit	Symbol	Name and Description
0	E	Equal - In a compare operation, the source operand equals the target operand.
1	G	Greater-Than - In a compare operation, the source operand is greater than the target operand.
2	V	Overflow - Set during Add, Subtract, or Arithmetic Left Shift if the Carry out of bit 15 is different than the Carry in to bit 15. If the set condition is not caused, V remains unchanged.
3	C	Carry - Receives the Carry out of bit 15 during an Add, Subtract, Arithmetic Left Shift, or Left Linked Shift. Reset during an Arithmetic Right Shift. Receives bit 0 shifted out from a Right Linked Shift.
4	F1	Flags 1, 2, or 3 - Programmable flag bits.
5	F2	
6	F3	
7	LP	Loop Complete - Set if content of register selected by XR field equals ZERO at the completion of an Autoincrement or Autodecrement instruction. Reset if content of XR is NOT ZERO.
8	O	Odd - For all General Register instructions except Compare, the Odd indicator receives the least significant bit of the result.
9	Z	Zero - For all General Register instructions except Compare, set if the result is ZERO and reset if NOT ZERO.
10	N	Negative - Receives the most significant bit of the result of any General Register instruction except Compare.
11	A	Active - Indicates that the processor is executing instructions. A is set unless the processor is quiescent.
12	L1	Interrupt Mask - Bits L1 through L4 correspond to system interrupt levels 1 through 4. When any bit is set or reset, respectively, the Bus Controller is requested to ignore or allow interrupt requests for the corresponding level.
13	L2	
14	L3	
15	L4	

INSTRUCTION SET ORGANIZATION

The SUE 1110 processor has 64 instruction mnemonics in four basic groups:

Control (12)

General Register (18)

Branch (26)

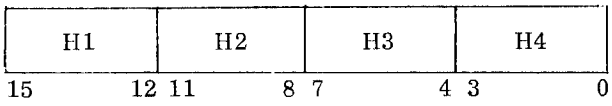
Shift (8)

Each group has functions designated by class, and each class has associated operations. For example, the General Register group has seven classes with eight general operations common to all classes.

The Control group includes load/store of multiple general purpose registers, load/store of status conditions and interrupt control instructions. The General Register group performs the arithmetic, logical, compare, test and data handling operations. The Branch group includes the unconditional and conditional branch functions. The Shift group provides 1-to 15-bit shift capabilities in eight different modes.

Care has been taken in the SUE design to provide an instruction format that is easy to use and understand. One of the design goals was to facilitate the encoding and decoding of the machine language code.

General Format



Field	Usage
H1	Class designation
H2	Operation designation
H3	Accumulator or G. P. register designation
H4	Index or G. P. register designation

SUE instruction fields do not overlap the four hex digits H1, H2, H3 and H4. Those fields that are subsets of a hex digit are right justified with the high-order bit used to represent the less common condition.

CLASSES

The instruction set is divided into sixteen classes indicating the type of instruction function. The SUE 1110 processor has eleven defined classes and three reserved classes, leaving two undefined. Undefined classes are available for user definition to handle special requirements.

Class Field	Group	Name/Status
0	Control	Control
1	General Register	Register to Memory, Auto-Decrement
2	General Register	Register to Memory, Auto-Increment
3	General Register	Register to Memory
4	General Register	Register to Register, Jump, Jump to Subroutine, and Data to Register
5	General Register	Memory to Register, Auto-Decrement
6	General Register	Memory to Register, Auto-Increment
7	General Register	Memory to Register
8	Branch	Branch if False or No Operation
9	Branch	Branch if True or Unconditional
10 (A)	Shift	Shift
11 (B)	Extended	Reserved
12 (C)	Extended	Reserved
13 (D)	Extended	Reserved
14 (E)	Special	Undefined
15 (F)	Special	Undefined

The class concept provides the programmer an immediate indication of instruction function. This function designation is especially useful in interpreting machine language representations.

OPERATIONS/CONDITIONS

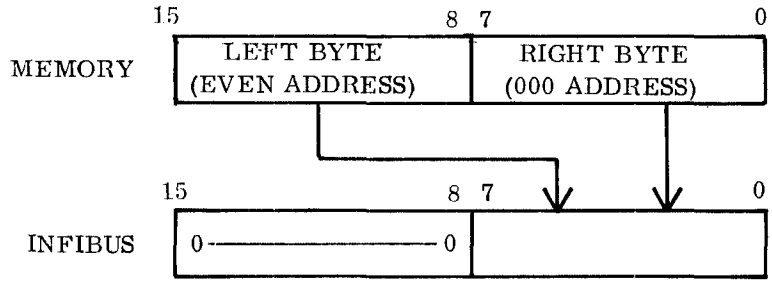
Each class of instruction has a set of operations or conditions associated with it. As each class is discussed a full explanation is given for the operation field.

ADDRESSING

The addressing mode key points will be discussed here. A detailed explanation of addressing modes was given in Chapter 6.

Byte Operations

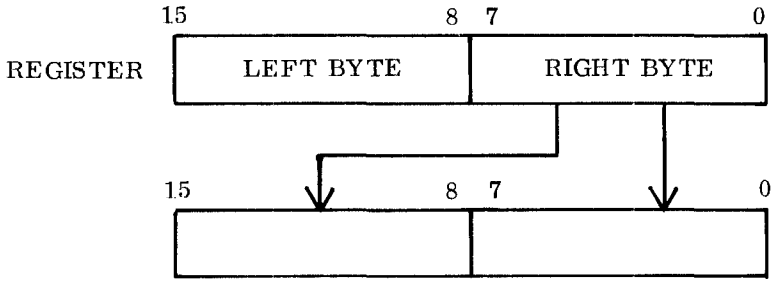
Memory-to-Register Instructions:



The operand (left or right byte) is made into a 16-bit word with the 8 most significant bits (Bits 15-8) equal to zero. This 16-bit word then operates on the designated register as specified by the OP code of the instruction.

For example: If a byte is moved to a register with a MOVB instruction, bits 15-8 of the target register are all zeros.

Register-to-Memory Instructions:



The source operand is always bits 7-0 of the designated register. The target is the left or right byte of the target cell or register as selected by bit 0 of the effective address.

Byte operations are valid for all instructions in the general register group except Class 4 (Register-to-Register and Data-to-Register). All other addressing modes are valid for byte instructions except indirect addressing.

Input/Output Instructions

There are no dedicated I/O instructions. The upper 2K address of SUE computers are reserved for device addresses, control words, status words, etc. Input/Output is performed by MOVE instructions and status checking by TEST instructions.

GENERAL PURPOSE REGISTERS

The SUE assembler has reserved names for the general purpose registers; the assembler will always recognize these names as register designators whenever they are used in an operand field.

NAME	USAGE
R0 or PC	Contains Program Counter
R1 through R7	General Purpose Register

Register R0 is used by the CPU as the program counter register. A running program may address R0 as an accumulator in the same way that it addresses R1 through R7. Care should be exercised in the use of R0, since any modification will change the sequence of instruction fetches from memory.

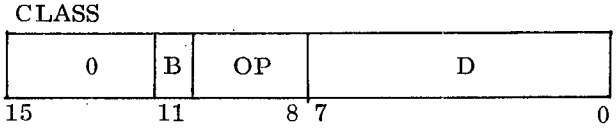
Registers R1 through R7 are high-speed hardware registers that may be used as accumulators, index registers, address pointers or stack pointers.

INSTRUCTION DEFINITIONS

Instructions are defined by class. Instruction format, addressing, and explanation of application are found under each instruction group.

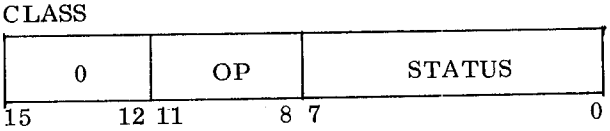
CONTROL GROUP INSTRUCTIONS

Memory Reference Format



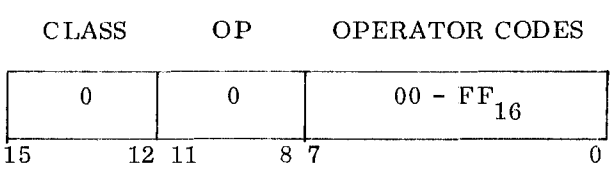
Class = Control Class (0)
B = Relative (1) or Absolute (0) designator
OP = Operation
D = Relative Displacement (-128_{10} to $+127_{10}$ words)
or Absolute Address (first 256_{10} Words of Memory)

Special Function Format



Class = Control Class (0)
OP = Operation
Status = Bits differ for each instruction.

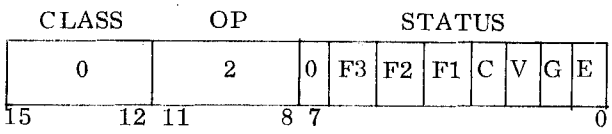
HALT



1.14μs

HALT stops the CPU from accessing instructions, inhibits all interrupts, and allows any Direct Data Transfers (DDT) in progress to continue until normal end-of-operation. The illumination of the HALT light on the operator's panel notifies the operator of the HALT. Execution resumes at the instruction following the HALT by pressing the run switch on the SUE control panel. Operator codes can be used to code different halts in a program.

RSTS Reset Status Indicators



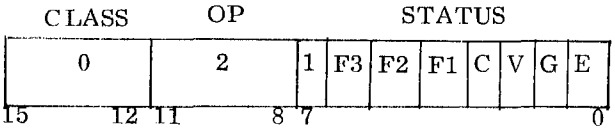
1.59μs

Status:

E	=	Equal
G	=	Greater Than
V	=	Overflow
C	=	Carry
F1-F3	=	Programmable Flags

For each "one" bit set in the status field the corresponding bit in the status register of the CPU is cleared to a zero. If the status field is zero the operation is effectively a no-operation.

SETS Set Status Indicators

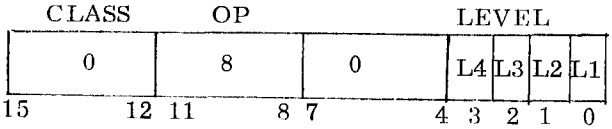


1.72μs

Status: Same as RSTS Status

For each "one" bit set in the status field the correponding bit in the CPU status register is set to a one. All zeros in the status field are effectively a no-operation. Note that bit 7 distinguishes between RSTS and SETS.

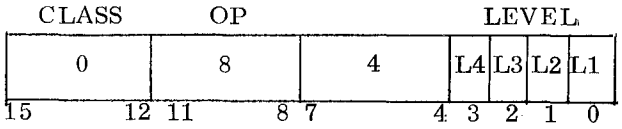
ENBL Enable Interrupts



1.85μs

Bits L1-L4 correspond to bits 12-15 of the Status Register. Each "zero" in the LEVEL field has no effect. Each "one" bit in the LEVEL field resets the corresponding status bit.

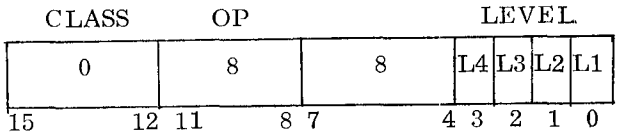
ENBW Enable and Wait



2.80 + μs

ENBW performs the same operation as ENBL and also puts the CPU into a wait state pending the next enabled interrupt.

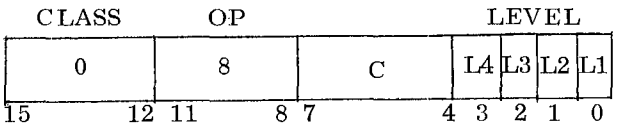
DSBL Disable Interrupts



1.98μs

Bits L1-L4 correspond to bits 12-15 of the status register. Each "zero" in the LEVEL field has no effect, each "one" bit in the LEVEL field sets the corresponding status bit. However, if a device attempts to interrupt to an inhibited level the interrupt will not be lost but will occur when that level is enabled.

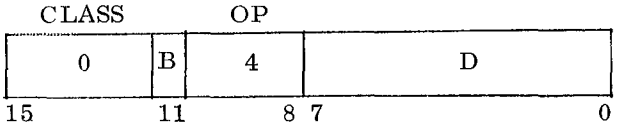
DSBW Disable and Wait



2.80 + μs

Performs the same operation as DSBL and also places the CPU into a wait state pending the next enabled interrupt. It is possible to disable all interrupts and enter a wait state. This will result in a Level 5 interrupt.

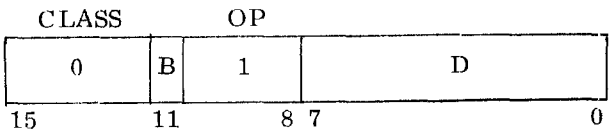
RETN Return from Interrupt



Rel-4.58μs*
Abs-4.26μs*

Transfers the content of a memory word located at D, into the system status register and the content of the next memory word to the Program Counter (PC). This one word instruction restores system status and transfers program control to the Return address of the calling function.

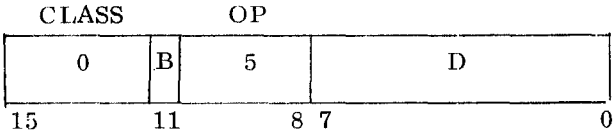
STSM Status to Memory



Rel-2.46μs*
Abs-2.14μs*

Transfers the CPU status register to the memory word addressed by D. (See Note below.)

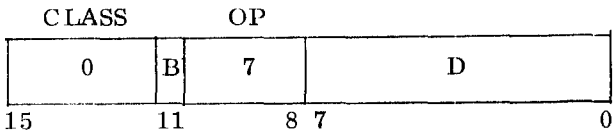
MSTS Memory to Status



Rel-2.79μ*
Abs-2.47μs*

Transfers the content of a memory word addressed by D to the CPUs status register. (See Note below.)

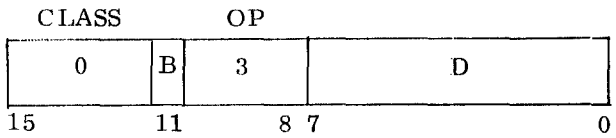
MREG Memory to Registers



Rel-8.25μs*
Abs-7.93μs*

The seven consecutive memory locations beginning with the effective address (D) are placed in general purpose registers R1 through R7 respectively. MREG and REGM are very useful in stack processing.

REGM Registers to Memory



Rel-7.56μs*
Abs-7.24μs*

* The operand fetch timings are included in these times.

The seven general purpose registers (R1-R7) are stored in the seven consecutive memory locations addressed by D.

Note: These instructions are used to initialize and preserve bits 10-0 of the status register in an interrupt response routine.

GENERAL REGISTER GROUP

The General register group of instructions, as defined for the SUE 1110 processor, uses seven of the sixteen classes:

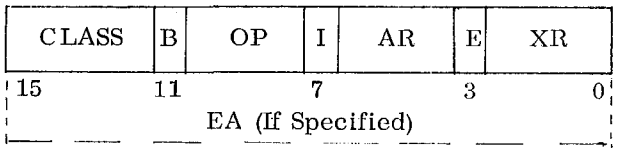
CLASS NO.	CLASS NAME
1	Register to Memory, Auto-decrement
2	Register to Memory, Auto-increment
3	Register to Memory
4	Data to Register, Register to Register, Jump and Jump to Subroutine
5	Memory to Register, Auto-decrement
6	Memory to Register, Auto-increment
7	Memory to Register

The General Register classes are double operand instructions and are represented in assembly language as:

Label Operation Operand 1, Operand 2.

Where Operand 1 is the source designation and Operand 2 is the target designation.

GENERAL FORMAT



CLASS = General Register (1, 2, 3, 4, 5, 6, 7)

B = Byte (1) or Word (0) Mode Instruction

OP = Register Operation

I = Indirect (1) or Direct (0) Address Indicator

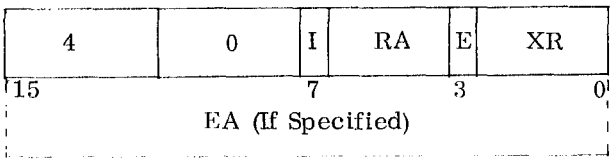
AR = Accumulator Register (target) in Memory-to-Register Classes, (source) in Register-to-Memory Classes

E = Single (0) or Double-Word (1) Instruction

XR = Index Register

EA = Extended Address

JUMP OR JUMP TO SUBROUTINE FORMAT



CLASS = General Register (4)

I = Indirect (1) or Direct (0) Address Indicator

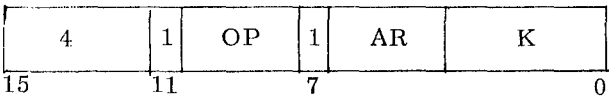
RA = Return Address Storage Register (JSBR only)

E = Single (0) or Double-Word(1) Instruction

XR = Index Register

EA = Extended Address

IMMEDIATE DATA-TO-REGISTER FORMAT



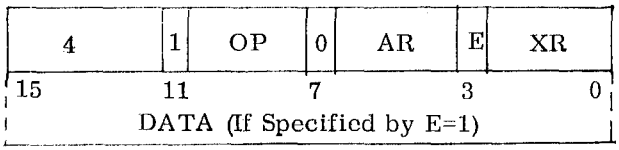
CLASS = General Register (4)

OP = Register Operation

AR = Accumulator Register (target)

K = Immediate Data ($0 \leq K \leq 15$)

GENERAL DATA OR REGISTER-TO-REGISTER FORMAT



CLASS = General Register (4)
OP = Register Operation
AR = Accumulator Register (target)
E = 0 for Register-to-Register Instructions
= 1 for Data-to-Register and Indexed Data-to-Register Instructions
XR = Index Register or Source Register
DATA = 16-bit Data Word (source)
(Byte instructions not allowed)

REGISTER OPERATIONS

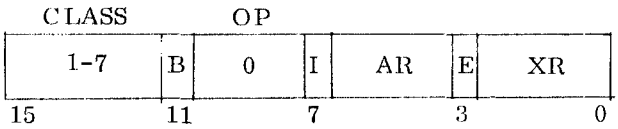
OP		Description	Status Affected (See Table 7-1)
0	MOVe	Transfer the source operand to the target operand. [S] ⟶ [T]	N, Z, O
1	SUBtract	Subtract the source operand from the target operand and store in the target operand. -[S] + [T] ⟶ [T]	C, V, N, Z, O
2	ADD	Form the sum of the source and target operands and store in the target operand. [S] + [T] ⟶ [T]	C, V, N, Z, O
3	AND	Form the logical product of the source and target operands and store in the target operand. [S] .AND. [T] ⟶ [T]	N, Z, O
4	Inclusive OR	Form the logical sum of the source and target operands and store in the target operand. [S] .IOR. [T] ⟶ [T]	N, Z, O
5	Exclusive OR	Form the logical difference of the source and target operands and store in the target operand. [S] .EOR. [T] ⟶ [T]	N, Z, O

OP		Description	Status Affected (See Table 7-1)
6	CoMPare	Subtract the source operand from the target operand. The register content and memory content are not affected.	G, E
		IF [S] = [T]	G Reset, E Set
		IF [S] < [T]	G Reset, E Reset
		IF [S] > [T]	G Set, E Reset
7	TeST	Form the logical product of the source and target operands. The register and memory content are not affected.	N, Z, O
		IF [S] .AND. [T] = All bits zero. Z set.	
		IF [S] .AND. [T] = High order bit set. N set.	
		IF [S] .AND. [T] = Low order bit set. O set.	

GENERAL REGISTER ADDRESSING MODES

Source	Target	Code
Register	Register (word instruction only)	R, R
Register	Indexed	R, (R)
Register	Indexed, Auto Increment	R, (R+)
Register	Indexed, Auto Decrement	R, (-R)
Register	Direct	R, A
Register	Direct, Indexed	R, A(R)
Register	Direct, Indexed with Auto Increment	R, A(R+)
Register	Direct, Indexed with Auto Decrement	R, A(-R)
Register	Indirect ADDR	R, *A
Register	Indirect ADDR, Indexed	R, *A(R)
Register	Indirect ADDR, Indexed with Auto Increment	R, *A(R+)
Register	Indirect, Indexed with Auto Decrement	R, *A(-R)
Register	Indirect-Indexed	R, *(R)
Register	Indirect-Indexed with Auto Increment	R, *(R+)
Register	Indirect-Indexed with Auto Decrement	R, *(-R)
Data	Register (word instructions only)	=0 to15, R
Data	Register (word instructions only)	=K, R
Data-Indexed	Register (word instructions only)	=K(R), R
Indexed	Register	(R), R
Indexed Auto Increment	Register	(R+), R
Indexed, Auto Decrement	Register	(-R), R
Direct	Register	A, R
Direct-Indexed	Register	A(R), R
Direct-Indexed, Auto Increment	Register	A(R+), R
Direct-Indexed, Auto Decrement	Register	A(-R), R
Indirect Address	Register	*A, R
Indirect Address-Indexed	Register	*A(R), R
Indirect Address-Indexed, Auto Increment	Register	*A(R+), R
Indirect Address-Indexed, Auto Decrement	Register	*A(-R), R
Indirect-Indexed	Register	*(R), R
Indirect-Indexed with Auto Increment	Register	*(R+), R
Indirect-Indexed with Auto Decrement	Register	*(-R), R

MOVB, MOVW Move byte, Move word



2.50μs*

The MOV instruction is used to transfer data between system registers, devices, and between registers and memory locations.

Operation: Moves [S] → [T]

Status Affected:
C - Unaffected
V - Unaffected
N - Set if Source is negative; if not, reset
Z - Set if Source = zero; if not, reset
O - Set if Source is odd; if not, reset

Description: Moves the content of the source operand to the target location and the content of the source is unchanged.

General purpose registers can be loaded with the content of memory locations by:

```
MOVW    A, R1
```

The content of memory address A replaces the content of register R1.

Registers can be loaded with data constants using the data to register instruction:

```
MOVW    #K, R1
```

The constant K is moved to register R1.

Operands may be pushed onto a stack by

```
MOVB        R1, (R3+)
```

The address of the stacking table in register R3 is incremented after the data is pushed onto the stack. Data may be popped off the stack by:

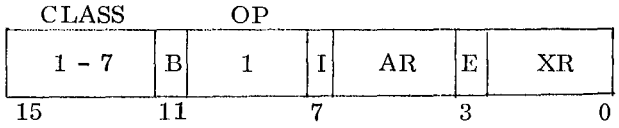
```
MOVB        (-R3), R1
```

The address in register R3 is automatically decremented before data is popped off the stack.

Register-to-Register moves are achieved by:

```
MOVW        R3, R2
```

SUBB, SUBW Subtract byte, Subtract word



2.79μs *

Operation: $-[S] + [T] \rightarrow [T]$

Status:
- C - Set if there was a carry from the most significant bit of the result; else reset
- V - Latest add or subtract operation produced a result outside of the range -2^{15} to $+2^{15}-1$
- N - Set if result is negative; if not, reset
- Z - Set if result = 0; if not, reset
- O - Set if result is odd; if not, reset

Description: Subtracts the source operand from the target operand and replaces the content of the target operand with the result. The content of the source is not affected by the operation. Arithmetic operations in SUE are in two's complement form. In the case of a subtract operation, the two's complement of the source is added to the target.

The subtract instruction can be used in the same manner as the examples illustrated for the MOVx instruction.

ADDB, ADDW Add byte, Add word

CLASS		OP					
1 - 7	B	2	I	AR	E	XR	2.79μs *

Operation: [S] + [T] → [T]

Status:
- C - Set if there was a carry from the most significant bit of result; if not reset
- V - Last add or subtract operation produced a result outside of the range -2^{15} to $+2^{15}-1$.
- N - Set if result is negative; if not, reset
- Z - Set if result = 0; if not, reset
- O - Set if result is odd; if not, reset

Description: Adds the source operand to the target operand and stores the results at the target address. The contents of the source are unaffected.

The add instruction, like subtract, is very useful for multiple increments (decrements) of address pointers or counters in memory or in registers. For example:

```
ADDW    =K, R2
```

adds the constant K to the register R2. All of the operand examples given for the MOVX instruction are possible using the Add instruction.

ANDB, ANDW And byte, And word

CLASS		OP					
1 - 7	B	3	I	AR	E	XR	2.50μs
15	11		7		3	0	

Operation: [S] .AND. [T] → [T]

Status: C - Unaffected

V - Unaffected

N - Set if result is negative; if not, reset

Z - Set if result is zero; if not, reset

O - Set if result is odd; if not, reset

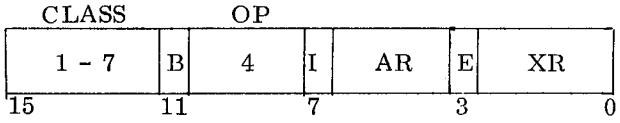
Description: The AND instruction forms the logical product of each bit in the target with the corresponding bit in the source. The result replaces the content of the target, the source is unaffected.

The AND instruction is useful for isolating data fields to be compared against test masks.

```
ANDW        MASK, R2
```

The memory location MASK is logically ANDed to the register R2. A branch on conditional status would be used to test the results of the mask operation.

IORB, IORW Inclusive Or byte, Inclusive Or word



2.50μs *

Operation: [S] .IOR. [T] → [T]

Status: C - Unaffected

V - Unaffected

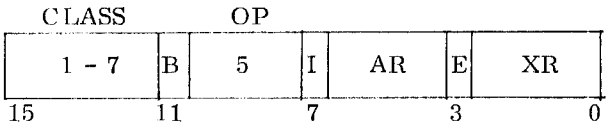
N - Set if result is negative; if not, reset

Z - Set if result = 0; if not, reset

O - Set if result is odd; if not, reset

Description: The Inclusive OR (IOR) sets each bit position in the target if either the source or original target bit position (or both) was set. The source is unaffected.

EORB, EORW Exclusive-OR byte, Exclusive-OR word



2.50μs*

Operation: [S] .EOR. [T] → [T]

C - Unaffected

V - Unaffected

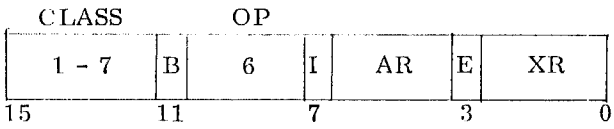
N - Set if result is negative; if not, reset

Z - Set if result is zero; if not, reset

O - Set if result is odd; if not, reset

Description: The exclusive OR instruction compares each bit of the source to the content of the target. The bits that do not compare result in a set bit, all others are reset. The result replaces the content of the target operand, the source is unaffected. The exclusive OR instruction is useful for data comparison or one's complement of words or portions of words.

CMPB, CMPW Compare byte, Compare word



2.69μs*

Operation: -[S] + [T]

Status: C, V, N, Z, O are unaffected

[S] = [T] Sets the Equal (E) and resets the Greater-Than (G) indicator

[S] > [T] Sets the Greater Than (G) indicator and resets the (E)

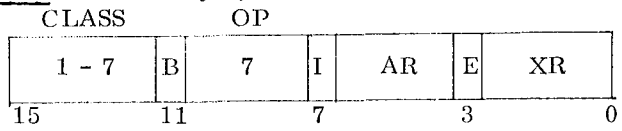
[S] < [T] Resets the (G) and (E) indicators

Description: The compare instruction performs an arithmetic compare by subtracting the source from the target and indicates equal-to, less-than or greater-than result via the (G) and (E) status bits. The content of the source and target are unaffected by the operation.

The compare instruction is useful in searching for known data. For example, a table termination character might be an FF in Hex. As each character is moved into a register for output it can be checked against the termination character:

```
CMPB    TERM, R3
BEQT    DONE
```

TSTB, TSTW Test byte, Test word



2.50μs*

Operation: [S] .AND. [T]

Status:

C - Unaffected

V - Unaffected

N - Set if result is negative; if not, reset (Checks bit 15)

Z - Set if result is zero; if not, reset

O - Set if reset is odd; if not, reset (Checks bit 0)

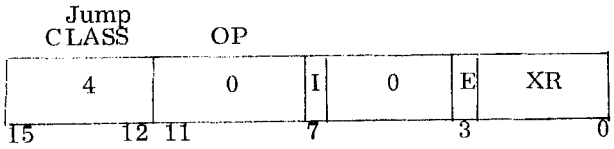
Description: The source operand is logically ANDed to the target operand. If all the bits in the source that correspond to bits set in the target mask are reset, then the result is zero and the Z status bit is set for conditional branching. If any of the tested bits in the source are set, then the result is non-zero and the Z status bit is reset. The source and target operands are unaffected by the test instruction.

The TST operation is designed to test for set status conditions in I/O operations or other masking type operations. For example, device status can be checked for abort conditions using the test instruction coupled with the branch-if-zero instruction:

```
TSTW    R2, ABORT
BZET    OK
```

The memory word ABORT contains the bits corresponding to the device abort conditions such as out-of-service, device failure, timing error, etc. Current device status in R2 is masked by the content of ABORT. The result sets the appropriate Z, N and O status indicators for conditional branching. If all the abort conditions were reset, meaning the conditions did not exist, then the Branch-if-Zero-True (BZET) instruction would transfer control to location OK. If any of the conditions did exist control would pass to the next sequential location following BZET.

JUMP



1.87 μs*

Operation: A ⟶ PC

Status: Unaffected

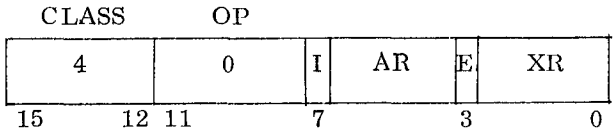
Description: The computed address replaces the content of the program counter; the instruction at this address is the next instruction to be executed.

JUMP is a word instruction and requires a word address as its operand. JUMP is capable of six addressing modes:

Direct	A
Indexed	(R)
Direct-Indexed	A(R)
Indirect Address	*A
Indirect-Indexed	*(R)
Indirect Address-Indexed	*A(R)

JUMP can be used to transfer control anywhere within the 31K words of program area.

JSBR Jump to Subroutine



1.87μs*

Operation: [PC] + 2→AR if E = 0 and [PC]+4→AR if E = 1; A→[PC]

Status: Unaffected

Description: The specified AR points to the instruction after the JSBR; the computed address is placed into the PC.

JSBR is a word instruction and requires a word address as its operand. JSBR has the same addressing capability as JUMP.

When a JSBR call passes parameters to a subroutine, the return address must be incremented past the call parameters to achieve the correct return address.

```
P       JSBR    SUBR,R1
P+2     (Part of JSBR Instruction)
P+4     DATA    100             Parameter 1
P+6     ADDR    1000            Parameter 2
P+8     Normal Return
          .
          .
          .
SUBR    Start of subroutine
          .
          .
          .
        JUMP 4 (R1)             Return instruction
```

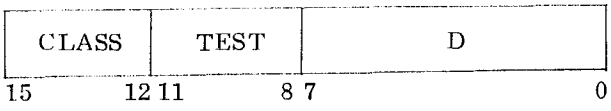
JUMP 4(R1) adds the content of the extended address, which in this case is a 4, to the content of register R1 and stores the result in the PC. Thus, the return is effectively completed in one instruction.

BRANCH GROUP INSTRUCTIONS

The Branch Group includes:

Branch Unconditional
Branch Conditional
No Operation

BRANCH FORMAT



CLASS = Branch Class (8 or 9)

TEST = Condition to be tested

D = Relative Displacement -128_{10} to $+127_{10}$ words

Branch instructions are word instructions and the displacement must always represent word addresses. The displacement is a two's complement binary number.

Branch Unconditional provides a means of transferring program control within a limited range (-128_{10} to $+127_{10}$ words) of the Program Counter (PC) by a one-word instruction.

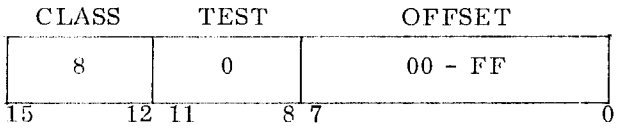
Branch Conditional provides the ability to test 12 conditions. Each condition can be tested to produce either a branch or a fall-through to the next instruction on either state (TRUE or FALSE) of the condition. Condition status is determined by testing the CPU status indicators and programmable flags.

No Operation is an instruction which merely passes control to the next sequential instruction. The displacement field is not decoded and can be used as a programming tool.

Table 7-1. Branch Conditions

CONDITION	MEANING (TRUE CONDITION)
UNCONDITIONAL	The branch is made unconditionally.
EQ - EQUAL	The last compare operation found the two operands to be equal to each other.
GT - GREATER THAN	The last compare operation found the source operand greater than the target operand.
OV - OVERFLOW	Overflow status bit is set during an Add, Subtract, Left Logical Linked, or Arithmetic Left Shift instruction if the carry out of bit 15 differs from the carry into bit 15.
CY - CARRY	Carry status bit copies the carry out of bit 15 during an Add, Subtract, Arithmetic Left Shift, or Left Linked Shift. CY is reset by an Arithmetic Right Shift. CY copies the least significant bit (bit 0) shifted out by a Right Linked shift.
F1 - FLAG 1 F2 - FLAG 2 F3 - FLAG 3	These are three programmable flags that can be set or reset by a "set or reset status indicator" instruction.
LP - LOOP COMPLETE	LOOP is set if the result of the latest auto-increment or auto-decrement on an index register equals 0; otherwise LOOP is reset.
OD - ODD	Result of the last general operation (except compare) was an odd number (LSB=1).
ZE - ZERO	Result of the last general operation (except compare) was all bits=0.
NG - NEGATIVE	Result of the last general operation (except compare) was a negative number (MSB=1).
LT - LESS THAN	Last compare operation found the source operand less than the target operand. (Both EQ and GT are tested)

NOPR No Operation



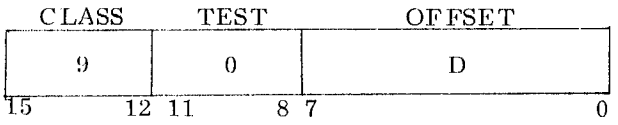
17.8μs

Operation: No-Operation is performed; control passes to the next sequential instruction.

Status: Unaffected.

Description: No Operation passes control to the next sequential instruction without decoding the OFFSET field.

BRUN Branch Unconditional



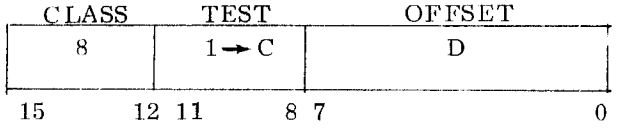
2.72μs

Operation: 2D+[PC]→[PC]

Status: Unaffected

Description: Branch Unconditional changes the sequence of instruction execution by placing a new effective address in the program counter.

BxxF Branch on conditions false

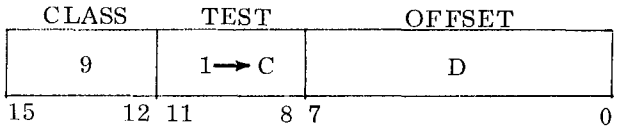


Fall-thru - 1.78μs
Branch - 2.72μs

Operation: 2D+[PC]→[PC]

Description: Branch conditions (BxxF listed in table 7-2) are tested: if false the effective address is placed into the PC; if not the next sequential instruction is executed.

BxxT Branch on conditions true



Fall-thru 1.78μs
Branch 2.72μs

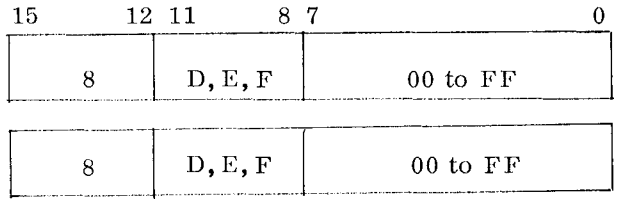
Operation: 2D+ [PC]→[PC]

Description: Branch conditions (BxxT listed in Table 7-2) are tested: if true the effective address is placed in the PC: if not the next sequential instruction is executed.

Table 7-2. Branch Conditions

FALSE	TRUE	TEST	CONDITION	STATUS BIT
BEQF	BEQT	1	Equal	0
BGTF	BGTT	2	Greater-Than	1
BOVF	BOVT	3	Overflow	2
BCYF	BCYT	4	Carry	3
BF1F	BF1T	5	Flag 1	4
BF2F	BF2T	6	Flag 2	5
BF3F	BF3T	7	Flag 3	6
BLPF	BLPT	8	Loop Complete	7
BODF	BODT	9	Odd	8
BZEF	BZET	A	Zero	9
BNGF	BNGT	B	Negative	10
BLTF	BLTT	C	Less-Than	0,1

The test conditions D, E and F are undefined and if data is executed of the form



they will cause a trap to interrupt level 5. Refer to Chapter 3 for level 5 definitions.

Branch instructions use a relative displacement addressing technique that allows a branch 128 words back or 127 words ahead of the branch instruction. Instructions reside in memory at even-numbered addresses, therefore, the branch instruction operand address is always even-numbered.

The SUE assembler performs the address arithmetic and assembles the correct offset into the instruction format. The assembler checks whether the offset is within the displacement range and error flags those outside the range. The assembly language forms are:

```
        BxxT    A
or:     BxxF    +6
```

A or +6 is the address or word displacement of the branch location, xx indicates the branch condition, and T and F are the TRUE or FALSE states of the condition to be branched upon. The instruction is read as:

Branch - if condition is True or
Branch - if condition is False;

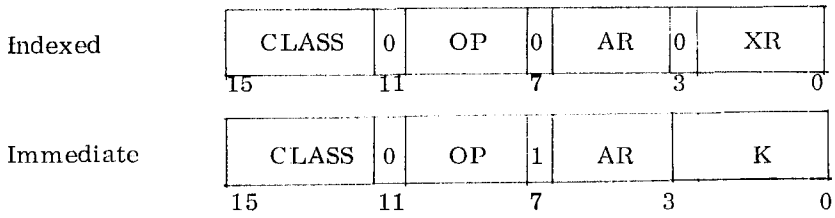
if not fall-through to the next sequential instruction.

The Branch instruction group contains 26 instructions. The two basic mnemonics (BxxT and BxxF) multiplied by the twelve TEST conditions account for 24. NOPR and BRUN make up the remainder.

SHIFT GROUP INSTRUCTIONS

The Shift Group provides shifts of 0 to 15 bits in one instruction. The shift instructions include arithmetic, logical and circular operations.

Shift Formats



CLASS = Shift Class (A)

Bit 7 = 0 = Count is indexed
= 1 = Count is immediate
AR = Register to be shifted
XR = Register containing count in its four least significant bits
K = Number of bit positions shifted

OP	Function
0	Single Left Arithmetic Open
1	Single Left Logical Linked
2	Single Left Logical Open
3	Single Left Logical Closed
4	Single Right Arithmetic Open
5	Single Right Logical Linked
6	Single Right Logical Open
7	Single Right Logical Closed

The shift count K can be contained in the rightmost 4 bits of the shift instruction:

```
SRLO    AR, K
```

or is specified as the least significant four bits in a general purpose register XR.

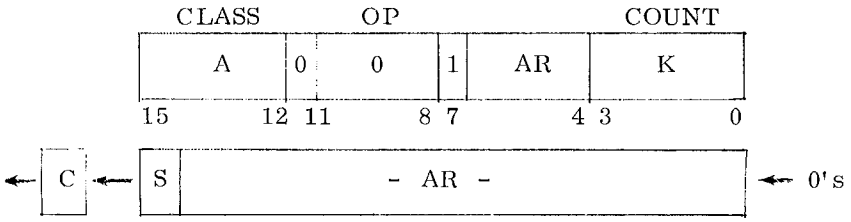
```
SRLO    AR, XR
```

AR is the register designated to be shifted.

There are eight shift instructions in the Shift class. Only the immediate format is illustrated here for simplicity.

Timing: The timing for all shift instructions is $2.76 + (0.26 \times \text{shift count})\ \mu s$.

SLAO - Single Left Arithmetic Open



Status:

C - Contains the last bit shifted out of bit 15

V - Set when bit shifted out of bit 15 differs from the new sign bit shifted into bit 15.

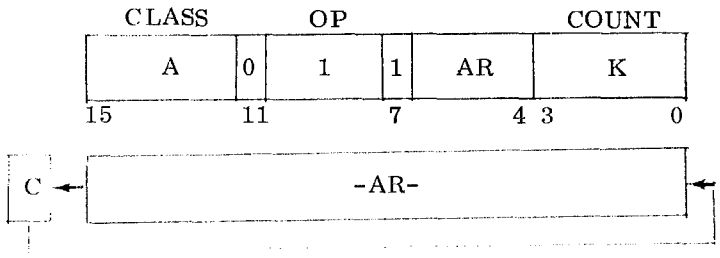
N - Set if the result is negative; if not, reset

Z - Set if the result equals zero; if not, reset

O - Reset

Description: Shifts bits 0 through 15 of AR left the number of places specified by the count K. Bits shifted out of bit 15 pass through the carry status indicator. Zeros are shifted into bit position 0.

SLLL Single Left Logical Linked



Status:

C - Set if last bit out of bit 15 was set; if not reset

V - Unaffected

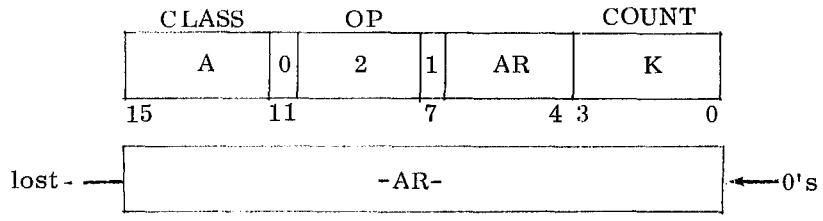
N - Set if high order bit of result set; if not, reset

Z - Set if all bits of the result equal zero; if not, reset

O - Set if low order bit of result set; if not, reset

Description: Shifts all bits of register AR left the number of places specified by the count K. Bits shifted out of bit 15 pass through bit C of the status register. Bits out of C pass into bit 0. In effect, there is a 17-bit rotate with the C bit inserted.

SLLO Single Left Logical Open



Status:

C - Unaffected

V - Unaffected

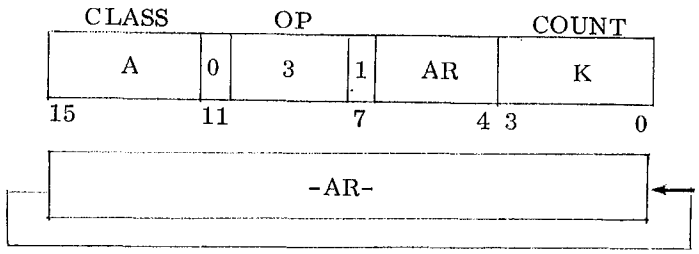
N - Set if high order bit set: if not reset

Z - Set if result is zero: if not reset

O - Reset

Description: Shifts all bits of AR left the number of places specified by the count K. Bits shifted out of bit 15 are lost. Zeros are shifted into bit 0.

SLLC Single Left Logical Closed



Status:

C - Unaffected

V - Unaffected

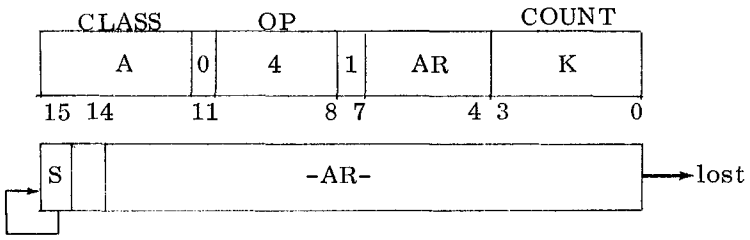
N - Set if high order bit set; if not, reset

Z - Set if all bits equal zero; if not, reset

O - Set if low order bit set; if not, reset

Description: All bits of the register AR are shifted left the number of places specified by the shift count K. Bits shifted out of bit 15 are shifted into bit 0.

SRAO Single Right Arithmetic Open



Status:

C - Reset

V - Unaffected

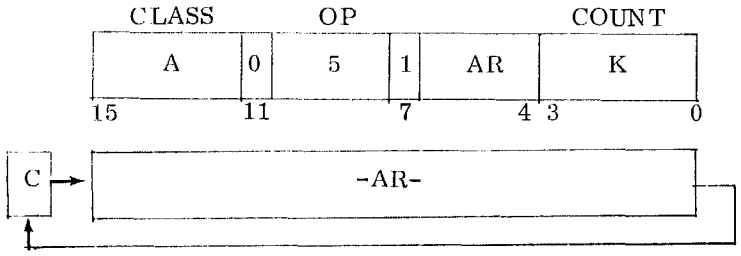
N - Set if result is negative; if not, reset

Z - Set if result is zero; if not, reset

O - Reset

Description: Shifts all bits of AR right the number of places specified by the count K. Bits shifted out of bit 0 are lost. Bit 15 is propagated.

SRLL Single Right Logical Linked



Status:

C - Set if last bit out of bit 0 was set; if not, reset

V - Unaffected

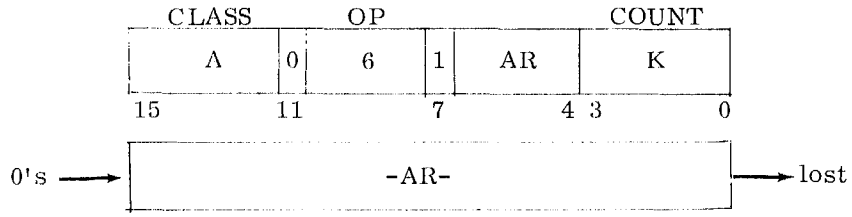
N - Set if high order bit of result is set; if not, reset

Z - Set if result is zero; if not, reset

O - Set if low order bit of result is set; if not, reset

Description: Shifts all bits of register AR right the number of places specified by count K. Bits shifted out of bit 0 pass through bit C of the status register. Bits out of C pass into Bit 15.

SRLO Single Right Logical Open



Status:

C - Unaffected

V - Unaffected

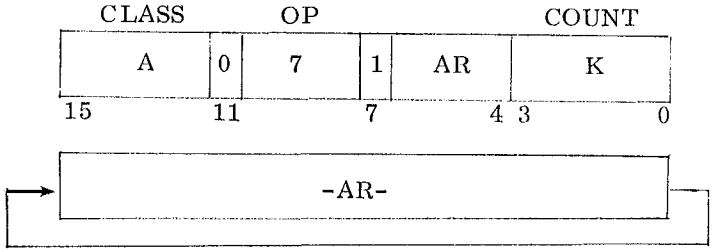
N - Reset

Z - Set if result is zero; if not, reset

O - Set if low order bit set; if not, reset

Description: Shifts all bits of AR right the number of places specified by the count K. Bits shifted out of bit 0 are lost. Zeros are shifted into bit 15.

SRLC Single Right Logical Closed



Status:

C - Unaffected

V - Unaffected

N - Set if high order bit of result set; if not, reset

Z - Set if all bits of result equal zero; if not, reset

O - Set if low order bit of result set; if not, reset

Description: Shifts all bits of AR right the number of places specified by the count K. Bits shifted out of bit 0 are shifted into bit 15.

INSTRUCTION TIMING

All timings are listed throughout this chapter alongside the instruction formats. Timings marked with an asterisk are for memory reference instructions, and must be added to the operand fetch timings listed in Appendix A. All operand fetch timings are for 850-nanosecond core memories.

CHAPTER 8

PROGRAMMING EXAMPLES

Programming simplicity is a feature of the SUE computer. Examples of this range from the simplest tasks of packing a buffer to the more complicated tasks of re-entrant techniques using stack processing. The instruction set features:

- Auto-increment and auto-decrement in both word and byte modes for easy table or stack processing.
- Loop complete status indicator for end-of-loop testing with a separate test instruction.
- Compare and test of either arithmetic or logical operations without changing source or destination operands.
- Relatively few easy to understand mnemonics such as MOVW, ADDW, SUBW, CMPW, EORW, ANDW, IORW, and TSTW shorten the learning period for the assembly language.
- Automatic testing and setting of Negative, Zero and Odd status for arithmetic, logical and move instructions.
- Minimum software overhead for subroutine entry and direct/indirect argument transfer.
- Generalized push-down, pop-up stack processing with no hardware restrictions on register usage or size of stack.
- Ease of interrupt handling.
- Ability to store/restore all general purpose registers with one instruction.

The programming examples in this chapter are guidelines. To be executable, definitions of variables, constants, and subroutines are usually required.

BYTE-ORIENTED BUFFER HANDLING

The buffer handling routine in this section reads and stored N characters, byte-by-byte, into BTABLE.

BTABLE+0	Byte 0
+1	Byte 1
+2	Byte 2
+3	Byte 3
+4	Byte 4
	.
	.
	.
	.
N-2	Byte N-2
N-1	Byte N-1

Odd addresses in a memory operand access the right byte and even addresses access the left byte. This feature causes alphabetic strings to be stored in left to right orientation as they are placed in memory, e.g., the string DEFINITION is placed in BTABLE;

BTABLE+0	D
+1	E
+2	F
+3	I
+4	N
+5	I
+6	T
+7	I
+8	O
+9	N

Placing the characters into BTABLE is facilitated by the auto-increment/decrement feature of the SUE Computer. In the instruction

MOVB R3, BTABLE(R2+)

the index R2 is incremented by one after each byte is stored in its proper location in BTABLE. This sets up the proper address for the next byte to be stored. The following illustrates how an N character buffer can be packed as it is being input:

```
        MOVW    =0, R2              Initialize index register R2 to zero.
LOOP    JSBR    FETCHB, R7          Subroutine inputs byte into R3.
        MOVB    R3, BTABLE(R2+)     Place byte in BTABLE indexed by
                                      R2 and increment R2 by 1.
        CMPW    =N, R2              Check for end of buffer.
        BEQF    LOOP                Branch to LOOP if equal is not set.
        HALT                        Halt when complete
```

WORD-ORIENTED BUFFER HANDLING

This table illustrates the packing of four words in a memory buffer:

WTABLE+0	Word 0
+2	Word 1
+4	Word 2
+6	Word 3

The auto-increment/decrement feature of the SUE Computer will increment/decrement the index by 2 for word mode operations. For use with the Loop Complete Indicator, the index must be made to go to Zero after the last word is stored. Following is one way of accomplishing this when the buffer must be packed in the order WTABLE+0, +2, ...+6

```
        MOVW    =-8, R3                 Initialize index register to the
                                          negative of the buffer size.
LOOP    JSBR    FETCHW, R7              Subroutine inputs word into R1.
        MOVW    R1, WTABLE+8(R3+)       Move first word into WTABLE+0,
                                          the next into WTABLE+2, etc.
        BLPF    LOOP                    Branch to LOOP until the index
                                          register R3 equals 0.
```

SUBROUTINE ENTRY AND EXIT

The examples that follow indicate how to program subroutines for the SUE Computer. If the register Rn (n = 1 to 7), which is used to store the return pointer, is not needed by the subroutine, the return can be accomplished by the one-word instruction.

```
JUMP    (Rn)
```

1. This subroutine, to test the current character for alphabetic or colon, illustrates the commonly used yes-no discriminator.

Calling Sequence:

```
        JSBR    ALFTST, R7      Jump to subroutine ALFTST store
                                  return in register 7.
        BRUN    NO              Return here if character is not
                                  colon or alphabetic.
        BRUN    YES             Return here if character is a
                                  colon or alphabetic.
        .
        .
        .
```

Subroutine:

```
ALFTST   MOVB    CHARAC,R1   Move char to R1 for test.
         CMPW    =':',R1     Compare Colon to R1.
         BEQT    ALFYES      Branch, if equal, to Yes Exit.
         CMPW    ='Z',R1     Check if highest alphabetic code
                               is less than character.
         BLTT    ALFNO       If yes, branch to No Exit.
         CMPW    ='A',R1     Check if lowest alphabetic code is
                               greater than character.
         BGTT    ALFNO       Branch, if greater than, to No
                               Exit.
ALFYES   ADDW    =2,R7       Update return to yes return in
                               calling sequence.
ALFNO    JUMP    (R7)        Jump to the address stored in
                               general register R7.
```

2. Subroutine to calculate and return the sum of N numbers via argument addresses. The argument pointer, R4, is automatically incremented after each argument is fetched. R4 points to the return address after all arguments are fetched.

Calling Sequence:

```
   JSBR   CALC,R4     Jump to Subroutine CALC, store
                        return in register 4.
   DATA   N           Number of arguments to be
                        summed.
   ADDR   ARG1        Address of first number to be
                        summed.
   ADDR   ARG2        Address of second number to be
                        summed.
    .      .            .
    .      .            .
    .      .            .
   ADDR   ARGN        Address of Nth number to be
                        summed.
   ADDR   CALSUM      Address of calculated sum.
   XXXX               Return from subroutine CALC
    .
    .
    .
```

Subroutine: (Size = 7 words, all instructions are single-word instructions.)

```
CALC    MOVW    =0, R3          Zero Accumulator register.
        MOVW    (R4+), R2       Move Number of arguments to R2.
LOOP    ADDW    *(R4+), R3      Add argument to accumulator.
        SUBW    =1, R2          Decrement counter.
        BZEF    LOOP            Branch to Loop if counter not zero.
        MOVW    R3, *(R4+)      Move sum to the sum location.
        JUMP    (R4)            Jump to the location in R4, which
                                  now points to the subroutine
                                  return.
```

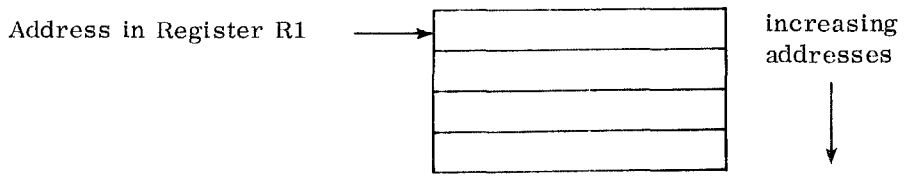
REENTRANT TECHNIQUES USING STACK PROCESSING

The auto-increment/decrement feature of the SUE Computer allows any of the seven General registers (1-7) to be used as a stack pointer. One useful type of stack is the pushdown or Last In-First Out stack (LIFO). Items are added to the top of the stack, one at a time, and can be removed only from the top of the stack. LIFO stacks are used in reentrant routines to provide the necessary storage for pointers and temporary values for each user of the routine. As one user is interrupted by another, the interrupted user variables are stacked and the new user is serviced. This stack process is continued until time is sufficient to complete a request. Then the unstacking begins handling the most recent user first until all requests are serviced.

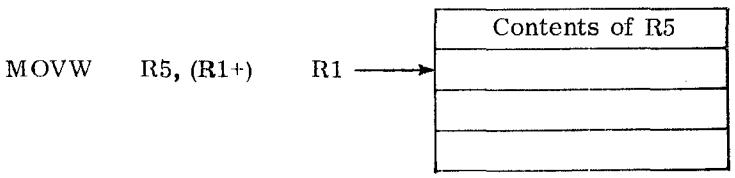
These examples illustrate LIFO stack processing:

Let R1 be initially set to point to the beginning of a three-word stack; note that auto-increments are performed after the move and auto-decrements before the move.

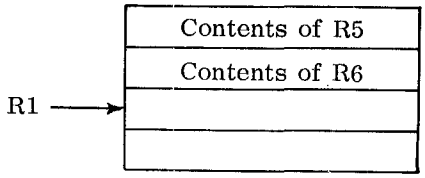
Empty Stack



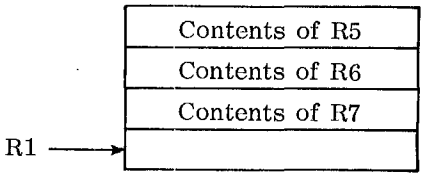
Push Items Onto Stack



MOVW R6, (R1+)



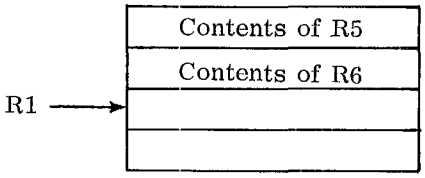
MOVW R7, (R1+)



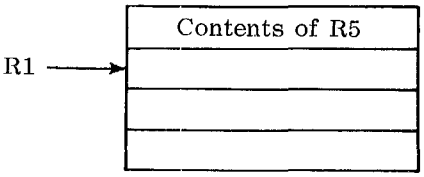
Full Stack

Remove Items from Stack

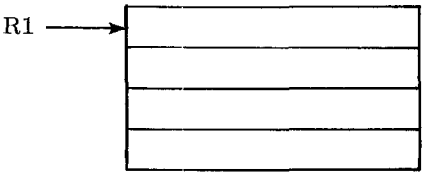
MOVW (-R1), R7



MOVW (-R1), R6



MOVW (-R1), R5



Empty Stack

SUBROUTINE REENTRANCE

Subroutines can be reentrant only if the permanent code and temporary storage elements are separated to permit multiple and reentrant execution. This requirement can be handled easily if a general register is dedicated as a stack pointer. When the subroutine is entered, all storage elements (registers, memory cells) that are modified during subroutine execution must be pushed onto the stack. Upon return these elements are popped off the stack.

This is a subroutine written in reentrant code using the general register R1 as a stack pointer to a pushdown stack:

```
Subroutine    JSBR      SUBR,R3      Jump to SUBR, save return in
   Call                                 R3.
              DATA      NUM          Subroutine parameter.
              ADDR      ARG          Pointer to subroutine parameter.
              Return from subroutine
                .         .
SUBR          DSBL      ALL          Disable all interrupts.
              MOVW      R3, (R1+)    Push return onto stack.
              MOVW      R5, (R1+)    Push register 5 onto stack.
Subroutine    MOVW      R6, (R1+)    Push register 6 onto stack.
   Entry      MOVW      (R3+), R5    Move first parameter into R5.
              MOVW      *(R3+), R6   Move second parameter into R6.
              ENBL      ALL          Enable all interrupts.
                .         .
                .         .          Subroutine body.
                .         .
              DSBL      ALL          Disable all interrupts.
              MOVW      (-R1), R6    Restore R6 from stack (Pop-up).
Subroutine    MOVW      (-R1), R5    Restore R5 from stack.
   Exit       MOVW      (-R1), R3    Restore return from stack.
              ENBL      ALL          Enable all interrupts.
              JUMP      4(R3)        Return to calling program past
                                       the two arguments (4 bytes).
ALL           DATA      H)F000
```

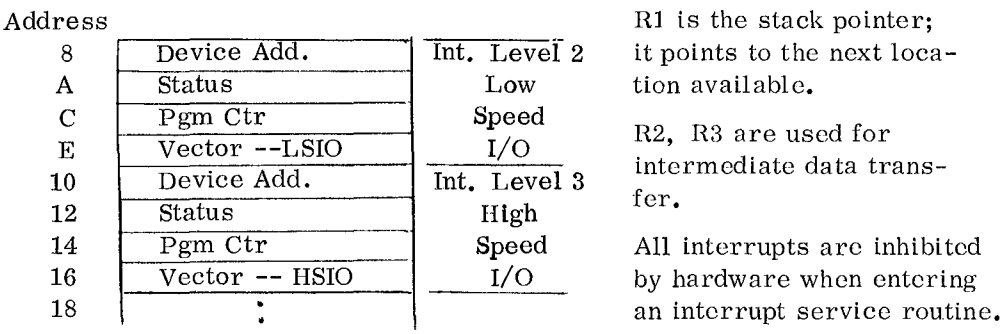
Stack size is determined by the number of subroutine calls made during system peak load times the number of words required for storing the variables for each call.

REENTRANT INTERRUPT HANDLING

An important use of reentrant coding techniques is in the servicing of input/output interrupts. The priority handling of I/O interrupts, necessary in most systems, requires that a high priority device be serviced before one of lower priority. Reentrance is one technique used to allow a single service routine to handle both devices at the same time.

This example shows the convenient use of reentrance in the SUE Computer. This technique requires allocation of a general purpose register for sole use as the stack pointer. STKEND is the address of the word following the last word of the stack and has been defined before the coding in the example.

Executive Area



R1 is the stack pointer; it points to the next location available.

R2, R3 are used for intermediate data transfer.

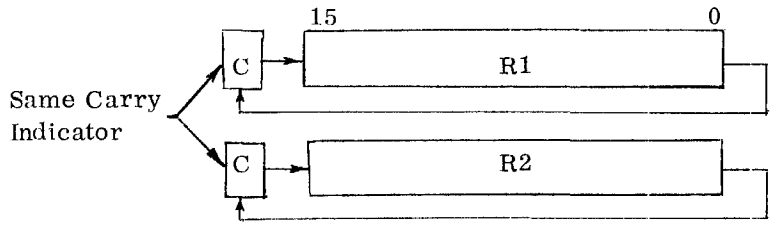
All interrupts are inhibited by hardware when entering an interrupt service routine.

```
LSIO    MOVW    R2, (R1+)           Store R2 in stack.
        MOVW    =8, R2              Move address of LS table to R2.
        BRUN    START
HSIO    MOVW    R2, (R1+)           Store R2 in stack.
        MOVW    =16, R2             Move address of HS table to R2.
START   MOVW    R3, (R1+)           Store R3 in stack.
        CMPW    =STKEND-10, R1      Check stack for room for 5 more
                                      words;
        BGTT    ERROR               go to ERROR if not enough room
        MOVW    (R2+), R3           Move device address to R3.
        MOVW    R3, (R1+)           Move device address to stack.
        MOVW    (R2+), R3           Move program status to R3.
        MOVW    R3, (R1+)           Move program status to stack.
        MOVW    (R2+), R3           Move program counter to R3.
        MOVW    R3, (R1+)           Move program counter to stack.
        ENBL    ALL                 Enable all interrupts.
          .       .                 .
          .       .                 . Body of Interrupt Service Routine.
          .       .                 .
        DSBL    ALL                 Disable all interrupts.
        MOVW    (-R1), R3           Move program counter to return
                                      area.
        MOVW    R3, RETPC
        MOVW    (-R1), R3           Move program status to return area.
        MOVW    R3, RETSTA
        MOVW    (-R1), R3           Discard device address
        MOVW    (-R1), R3           Restore R3.
        MOVW    (-R1), R2           Restore R2.
        RETN    RETSTA              Return from interrupt area
RETSTA  SAVE    2                     (2 words) to construct status
RETPC   SAVE    2                     and program counter for return.
```

INTEGER MULTIPLY SUBROUTINE

This coding example shows how to multiply two unsigned 16-bit integers. It illustrates a variety of instructions, such as the accumulator-to-memory MOVW, the branch conditionals, test operation, and the linked shift (SRLL - Single Right Logical Linked) which provides for easy double register shifting. Two sequential SRLL instructions with a count of 1 results in a double register shift because of the carry indicator transfer from one register to another.

```
RSTS  CARRY
SRLL  R1, 1
SRLL  R2, 1
```



Example:

```
CARRY   DEFN    8           Carry bit code for RSTS instruction.
        JSBR    :IMP, R7    Jump to subroutine :IMP, save
                              return in R7.
        ADDR    MCAND       Address of multiplicand.
        ADDR    MPLIER      Address of multiplier.
        ADDR    PROD        Address of first word of 2-word
                              product.
          .       .           .
          .       .           .
          .       .           .

:IMP    MOVW    *(R7+), R3  Move multiplicand into R3.
        MOVW    *(R7+), R2  Move multiplier into R2.
        MOVW    #0, R1      Clear accumulator for multiply.
        MOVW    #15, R4     Set word size counter.

$1      TSTW    #1, R2      Test least significant bit of multi-
                              plier.
        BODF    $2          Branch LSB is not ODD to $2.
        ADDW    R3, R1      LSB was a ONE, so add MCAND
                              to accumulator.
```

```
$2      RSTS    CARRY       Clear carry
        SRLL    R1,1        Double shift accumulator and
        SRLL    R2,1        Multiplier to set up for next LSB
                              test
        SUBW    =1,R4       Finished Loop?
        BNGF    $1          Branch if NEG indicator false to $1
        MOVW    R1,*(R7)    Move high order word to PROD.
        MOVW    R2,*2(R7+)  Move low order to second word of
                              PROD.
EXIT    JUMP    (R7)        Return
```

Note: $1 and $2 represent local labels. The Assembler clears these labels from the symbol table when it encounters a non-dollar-signed label (EXIT). This technique provides an efficient use of the symbol table permitting assembly of larger programs without symbol table overflow.

R2 is used as both the multiplier storage and as the least significant half of the product.

CHAPTER 9

INPUT/OUTPUT PROGRAMMING

The SUE System provides a simple method of I/O programming for many system modules. A module can be a peripheral controller, a memory, a CPU, or another processor.

Registers, whether they be a CPU general purpose register or the status, control and data register of a peripheral controller, have been assigned the upper 4K byte addresses ($FOOO_{16}$-$FFFF_{16}$). Instructions can use these addresses to initiate direct device-to-memory, memory-to-device, or device-to-device transfers in which no data passes through the CPU. Optionally, this upper address range may be restricted to 2K bytes. A system with CPU-independent I/O transfers is shown in figure 9-1.

SUE-018-72

Figure 9-1. CPU-Independent I/O Transfers

Because of unique device and memory identity on the Infibus, I/O instructions are not required. Input and output transfers are accomplished by word or byte MOV instructions exactly the same as moving data to and from a memory cell. This means that software moves control commands, data, and status between a general purpose register and a controller in the same manner that it would communicate with memory.

MOVW R3, CONTWD

This instruction moves the control command bits in R3 to the specified control register CONTWD.

I/O CONTROLLERS

Two general purpose controllers (parallel and serial) and one disk controller (IBM 5440 compatible) are available for SUE. The general purpose controllers can be configured to perform a wide range of input/output tasks through jumpers on each controller board.

One basic design on a single printed circuit board is used for all four models of the parallel controller. The difference in the 4501, 4503, 4506, and 4507 models is the insertion of different input/output components to accommodate the polarity of each peripheral device.

The serial controller is a one board asynchronous controller with both a Teletypewriter® current loop and an RS-232C compatible interface. Both bit configurations and baud rates (110, 300, 600, 1200, 2400, 4800) are controlled through jumpers on the board.

General purpose controllers, configured (jumpered) to operate with a specific device, are given a 4600 series number. The examples in this chapter referring to a specific device, will have the appropriate 4600 number. Those examples using the general capabilities of each controller will use 4501 for a parallel controller and a 4502 for a serial controller. The 4600 series includes serial controllers for Teletypewriter Models 33 and 35, asynchronous modems and certain serial interfaced CRTs. The parallel controller has been configured to control high speed paper tape equipment, line printers, card readers, special keyboards, CRTs, cassette and reel to reel magnetic tape drives, card punches, and computer-to-computer communications. Our customers have configured both controllers to interface to a host of special input/output equipment.

Both of SUE's general purpose controllers can be made into DMA device controllers by placing a 4590 Block Transfer Adapter (BTA) board in the right hand adjacent slot.

Figure 9-2 is a BTA transfer from the tape reader directly to memory concurrent with character transfers from the Teletypewriter to memory via the CPU.

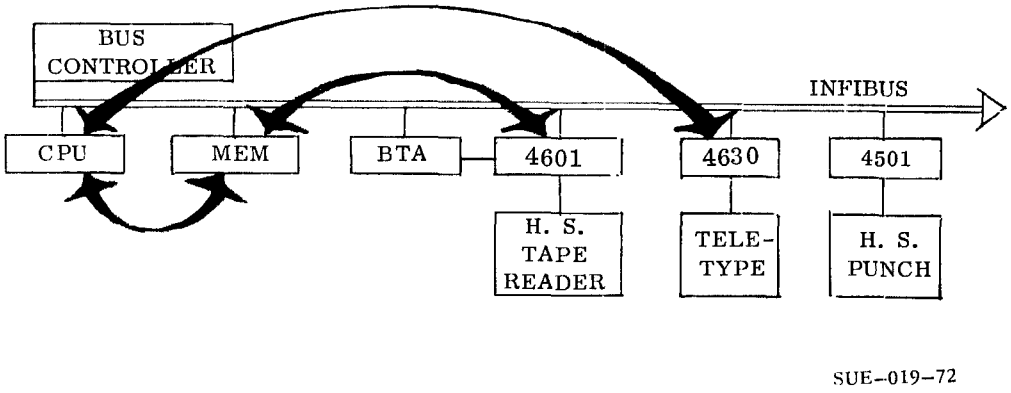
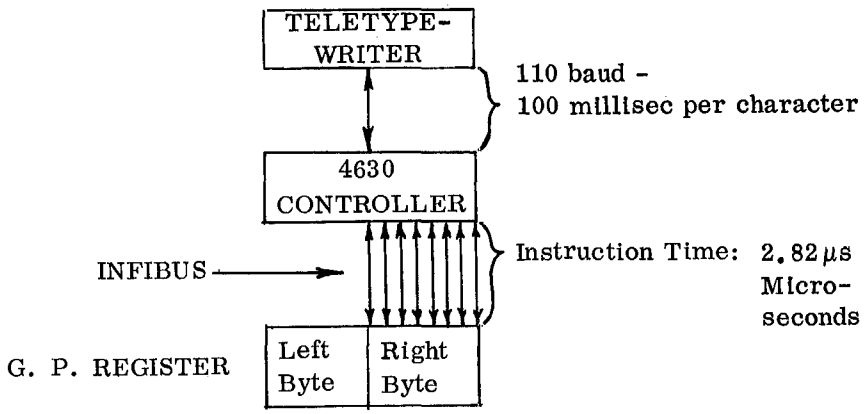


Figure 9-2. Concurrent I/O Transfers

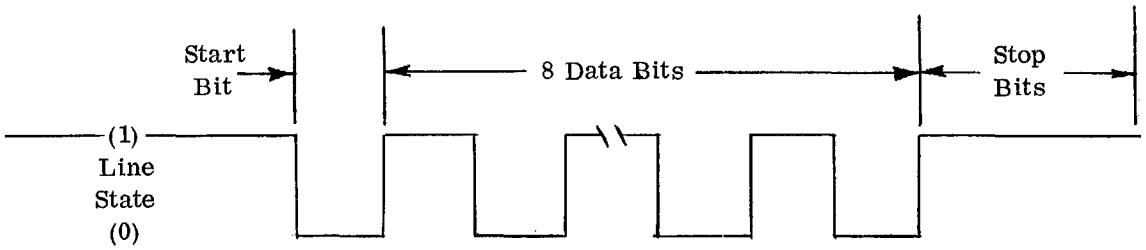
The following sections contain descriptions of I/O programming techniques using the Teletypewriter and high-speed (HS) tape equipment as examples of system communication across the Infibus.

TELETYPEWRITER (Model ASR-33) PROGRAMMING

The Model ASR-33 Teletypewriter has a keyboard, a printer, a paper tape reader, and a paper tape punch. The Teletypewriter transmits 11-bit ASCII code serially at 110 baud between the device and a 4630 controller. The controller transmits under control of the processor an 8-bit character (in parallel) between the controller data register and the right byte of a designated general register or memory word.



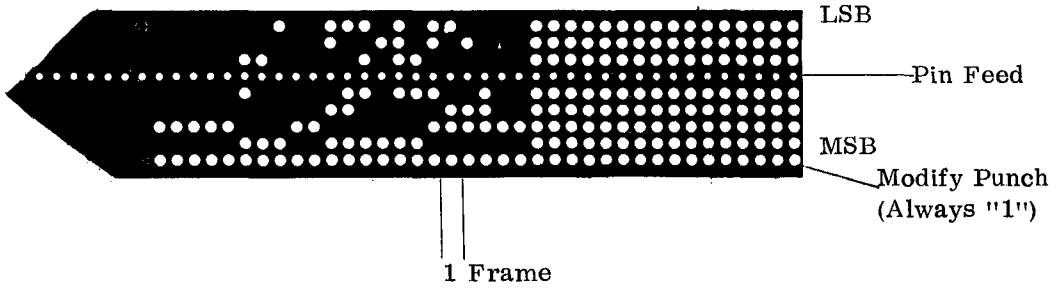
The 11-bit code Teletypewriter signals transmitted between the Teletypewriter and controller consist of one start bit, eight data bits and two stop bits.



ASCII

The 8-bit symbolic code transmitted between the controller and a register (or memory) is the United States American National Standard Code for Information Interchange (ASCII) modified. The modification is the setting of bit 8 to a one (see Appendix E).

TAPE MOTION



A symbolic paper-tape frame consists of a 7-bit ASCII code with the 8th MSB modified to a ONE. Refer to Appendix D for source and object tape format definitions.

The ASR-33 Teletypewriter generates and detects only the upper-case characters indicated on its keyboard. Appendix E has a complete list of legal characters and corresponding ASCII code.

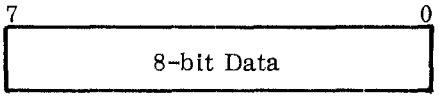
The ASR-33 Teleprinter is capable of printing a 72-character line using the carriage return/line feed sequence for proceeding to the next line.

TELETYPEWRITER CONTROL

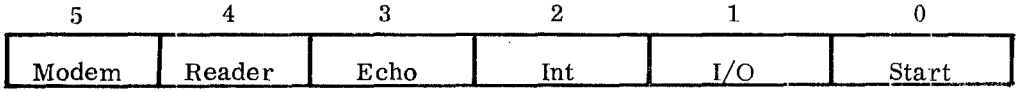
Serial information read or written by the Teletypewriter is assembled by the SUE 4630 Serial Controller for parallel transfers under control of the processor from the data register to a general purpose register. The 4630 is a half duplex 8-bit serial controller. It can stand alone or function with a Block Transfer Adapter (SUE 4590). The "stand alone" mode handles single-character transfers with the software option of CPU interrupts for each character. The BTA provides block transfer capability with CPU interrupts indicating end-of-operation.

4630 CONTROLLER REGISTERS

Data Register (8-bits)

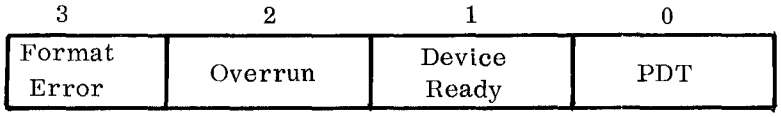


Control Register (6 bits)



Start Bit	0 = Idle State
	1 = Start Controller
I/O Bit	0 = Input Mode
	1 = Output Mode
Interrupt Bit	0 = Disable Interrupts
	1 = Enable Interrupts
Echo Bit	0 = Don't Echo
	1 = Echo Input to TTY Printer
Reader Bit	0 = Disable Tape Reader
	1 = Use Tape Reader
Modem Bit	0 = Don't Send
	1 = Send

Status Register (3 bits)



PDT Bit	Programmed Data Transfer bit = 1 in the input mode, when the 4630 has data for the CPU. After the data is placed on the Infibus, the bit goes to a zero until new data is ready. In the output mode, the PDT bit = 1 when the 4630 can accept data from the CPU and transmit it. Moving a character to the 4630 changes the PDT bit to a zero.
Device Ready Bit	0 = Device Ready 1 = Device Not Ready
Overrun Bit	0 = No Overrun 1 = The data register was not ready before new data had arrived from the TTY. The old data was lost.
Format Error	0 = No Format Error 1 = The data received has one start bit, the correct number of data bits, but no stop bit.

Writing the status register provides a clear to the controller and the device.

REGISTER ADDRESSES

STATUS	FWZ0
CONTROL	FWZ6
DATA	FWZ8

Where W and Z are any hexadecimal digits depending on system configuration.

KEYBOARD/READER

The keyboard may be operated independently of the paper tape reader. Control bit 4 connects or disconnects the reader from operation. Following is the control register setup for keyboard entry with automatic echo of the character to the printer. Teletypewriter interrupts are to level 2.

5	4	3	2	1	0
0	0	1	1	0	1
Modem	Keyboard Only	Echo	Interrupts Enabled	Input Mode	Start

A character is read from the low speed paper tape reader by setting bit 4 of the control register which, in turn, resets bit 0 of the status register indicating the controller busy. When the character bits start to enter the data register the controller de-energizes a relay in the Teletypewriter to release the tape feed latch. When released the latch mechanism stops the tape after a complete character has been read and before the next character is started. Once the character is completely available in the data buffer the PDT bit is set to indicate the 4630 has data for the CPU. If the interrupt is enabled a bus access for an interrupt is requested. Once the service routine has read the data the PDT bit is automatically reset indicating a readiness for new data.

PROGRAMMING EXAMPLE

Programmed Data Transfer with Interrupts Inhibited. The examples given on the following page are for the 4502 half duplex 8-bit serial I/O controller. This controller will handle Model 33, 35, and 37 Teletypewriters, asynchronous data modems, certain CRTs, cassettes, and any RS232C-compatible device.

KEYBOARD INPUT (INTERRUPTS DISABLED)

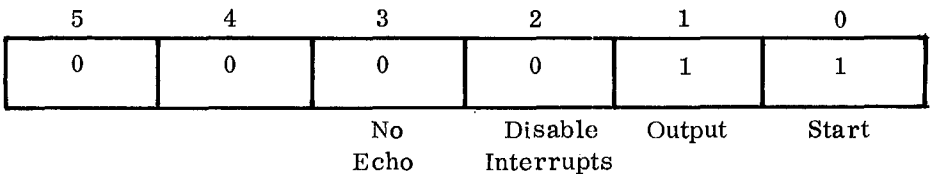
```
        MOVW    =72,R6          Place byte count in R6
        MOVB    =H)8D,R5        Place terminating Character in R5
        MOVW    TABL,R7         Place TABL location in R7
        MOVW    =9,R3           Move input code to R3  (Inhibited with
                                  echo)
        MOVW    R3,CONTWD       Move code to Control register
PDTSTA  MOVW    STAWD,R4        Move Status register to R4
        BODF    PDTSTA          Loop until the new data is in the Data
                                  register
        MOVW    DATAWD,R2       Move data from Data register to R2
        MOVB    R2,(R7+)        Move data into stack and increment
                                  pointer R7
        CMPB    R2,R5           Compare data to terminating character
        BEQT    PDTEND          If yes, then exit
        SUBW    =1,R6           Decrement byte count
        BZEF    PDTSTA          Loop back until byte count goes to zero
PDTEND  HALT                    Halt after completion
TABL    ADDR    TABLE           Address of storage table
TABLE   SAVE    72              Storage table
```

TO READ A CHARACTER FROM TAPE AND ECHO IT ON THE TELETYPEWRITER PRINTER (INTERRUPTS DISABLED)

```
CNT     DEFN    100             100 character input routine
ECHO    MOVW    =H)19,R1        (Start, Echo, TTY READER) Control Reg.
                                  Bits
        MOVW    R1,CONTWD       Control Reg. Bits to Control Word
        MOVW    =-CNT,R4        Set Minus CNT in Index Register R4
CKSTA   MOVW    STAWD,R2        I/O Status Word to R2
        BODF    CKSTA           Loop until new data is in the
                                Data Register
INPUT   MOVW    DATAWD,R3       Put Input Data in R3
        MOVB    R3, INBUF+
                CNT(R4+)        Store Character in Buffer indexed by R4
        BLPF    CKSTA           Go check for another Input Character
        HALT                    STOP
INBUF   SAVE    CNT             100-byte Input Buffer
```

TELEPRINTER/PUNCH

Under software control a character is sent in parallel from a general purpose register to the data register for transmission to the Teleprinter/punch unit. The 8-bit character code is preceded by the start code and followed by two stop codes that are generated by the controller. The 11-bit code is transferred from the controller to the Teletypewriter at the 110 baud rate requiring 100 milliseconds for completion. Following is the control register setup for the printer/punch operation with interrupts disabled.



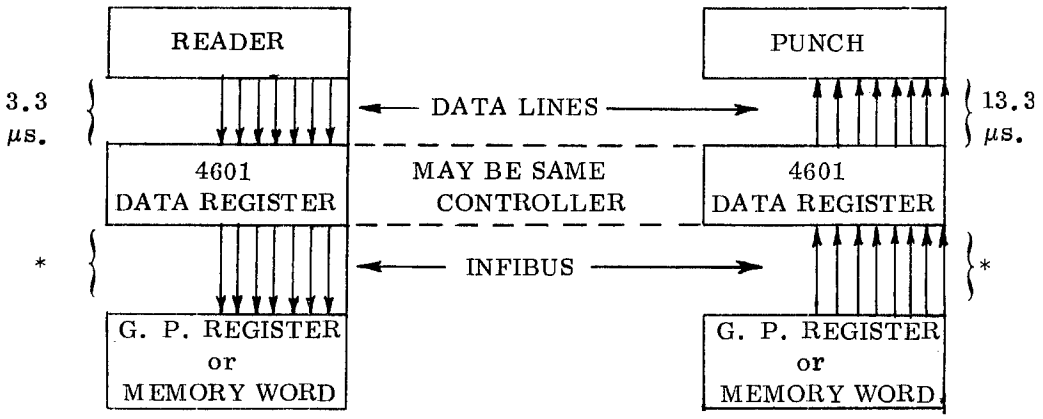
Once the controller has received the data byte it resets the PDT bit of the status register. When the last bit has been transmitted from the data register the PDT bit is set indicating not busy and ready for new data. In the interrupt disable mode the software samples the status waiting for the PDT bit to be set.

Refer to Appendix C for an example of Teleprinter/punch I/O.

HIGH-SPEED PAPER TAPE EQUIPMENT

The high-speed paper tape reader reads paper tape or Mylar-tape photoelectrically at 300 characters per second. The reader controller requests tape movement, transfers data from the reader into the data buffer and signals the CPU when data is present.

The high-speed paper tape punch punches paper tape or Mylar tape at 75 characters per second. The punch controller accepts data from the CPU or memory, outputs the data in parallel to the punch and signals the CPU for more data.



* Time between G. P. Register and Data Register 2.82 μs

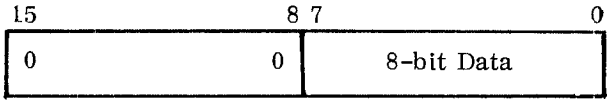
* Time between Memory Word and Data Register 3.48 μs

READER/PUNCH CONTROL

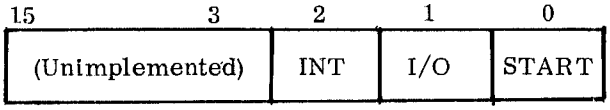
The Reader/Punch is controlled by the SUE-4601 half-duplex parallel controller. The 4601 can stand alone or function with a Block Transfer Adapter (SUE4590). The stand-alone mode handles single-character transfers with the software option of CPU interrupts for each character. The 4601 coupled with the 4590 provides block transfer capability between memory and a reader/punch with a CPU interrupt indicating end-of-block.

4601 Registers:

- Data Register



- Control Register (3 Bits)



Start Bit	0 = Idle State 1 = Start Operation
I/O Bit	0 = Input Mode 1 = Output Mode
Interrupt Bit	0 = Disable Interrupts 1 = Enable Interrupts

o Status Register

(Unimplemented)	Punch Ready	Reader Overrun	Reader Ready	PDT

PDT Bit	Programmed Data Transfer bit = 1 in the input mode when the 4601 has data for the CPU. The PDT bit = 0 when the data is placed on the Infibus and remains 0 until new data is received from the device. In the output mode the PDT bit = 1 when the 4601 can accept data from the CPU for transmission. Loading the 4601 data register with a character changes the PDT bit to "0"
Reader Ready Bit	0 = Reader Ready 1 = Reader not Ready
Reader Overrun Bit	0 = No Overrun 1 = The CPU did not remove the data from the data register before new data destroyed the old
Punch Ready Bit	0 = Punch Ready 1 = Punch Not Ready

PROGRAMMING EXAMPLE

This example illustrates the use of one 4601 controller for both input and output to two different devices. The coding does not include leader and trailer tape generation.

TO COPY A RECORD FROM THE HIGH SPEED TAPE READER TO THE HIGH SPEED TAPE PUNCH (INTERRUPTS INHIBITED). THE EXAMPLE RECORD IS 72 FRAMES IN LENGTH.

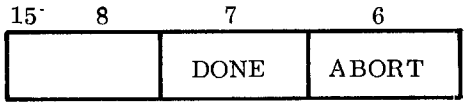
```
COPY    MOVW    =0,R4          Set Character Counter to Zero
        MOVW    =1,R1          Initialize 4601 Control
        MOVW    R1,CONTWD      Word to Start, Input, Interrupts Inhibited
PDT1    MOVW    STATWD,R2      Loop on PDT Bit of Status
        BODF    PDT1           Word for Data Available
        MOVB    DATAWD,R3      Input Character from 4601 Data
        MOVB    R3,TABLE
                (R4+)          Register and Store in Output Buffer
        CMPW    =72,R4         Check for Last Character
        BEQF    PDT1           Not Last Character - Go Read
        MOVW    =3,R1          Yes - Set 4601 to Output
        MOVW    R1,CONTWD      With Interrupts Inhibited
        MOVW    =0,R4          Zero Character Counter
PDT2    MOVW    STATWD,R2      Loop on PDT Bit of Status
        BODF    PDT2           Word for Controller Ready
        MOVB    TABLE(R4+),
                R3             No - FETCH and Output
        MOVB    R3,DATAWD      Character to 4601 Data Register
        CMPW    =72,R4         Is this the Last Character
        BEQF    PDT2           Not Last Character - Go Punch
        MOVW    =0,R1          Idle Controller
        MOVW    R1,CONTWD
```

DATA TRANSFERS USING THE SUE 4590 BLOCK TRANSFER ADAPTER (BTA)

The General Purpose Serial (4502) and Parallel (4501) Controllers for SUE are designed for single character transfer under program control. The Block Transfer Adapter (BTA) is a one-card option that provides block transfer control using DMA capability of the Infibus in conjunction with the 4501 and 4502 controller. The BTA must be plugged into the slot adjacent to the controller to be modified. All communication between the BTA and its companion controller is via the Infibus.

BTA REGISTERS

Status Register (2 Bits). –



Abort Bit
0 = Transfer not aborted
1 = An attempt to perform a direct data transfer on the Infibus was aborted by the bus controller.

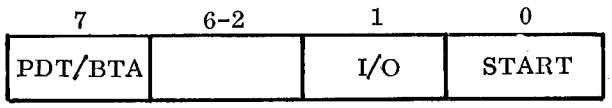
Done Bit
0 = Transfer in progress
1 = Block transfer complete. The block length register is zero, indicating that the data was properly handled by the controller. However, the device status bits indicate whether the data reached the device successfully or not.

Address Register. This 16-bit register is loaded with the starting memory location for the block to be transferred and is automatically incremented with each bus transfer. When the Block Length Counter goes to zero the Address register will contain the Address for the N+1st character (Word or Byte) of an N character block.

If the Address in the Address Register is FXXX, this register will not be incremented, allowing transfers from controller to controller.

Block Length Register. This 16-bit register is loaded with the number of words or bytes to be transferred. This register is decremented with each bus transfer.

Control Register (2 Bits).



Start Bit
0 = Idle State
1 = Start Operation

I/O Bit
0 = Input Mode
1 = Output Mode

PDT/BTA
0 = PDT transfers with Processor interrupt after each word or byte transfer to/from the device (Non-DMA).
1 = DDT block transfer with BTA active (DMA).

REGISTER ADDRESSES

The BTA registers are assigned in the upper 4K addresses as shown in the following table. The slave controller has the same status and control register addresses.

Register	Address
Status	FWZ0
Address	FWZ2
Block Length	FWZ4
Control	FWZ6
Data	FWZ8

where, W, Z are any hexadecimal digit, depending on the system configuration.

BTA OPERATION

Basic operation of the BTA for both input and output is the same. The only difference between Word and Byte modes is the way the address register is incremented; once for each byte transfer and twice for each word transfer. A jumper on the module selects the word or byte mode of operation.

Once the address register is loaded with the buffer address in memory, the block length register with the number of words or bytes to be transferred, and the control register with the start bit, the BTA bit and selected I/O mode, the DMA transfer begins. The transfer continues exclusive of the CPU operation until the block length goes to zero or until the bus controller aborts due to a system failure. In either case the controller interrupts the CPU. The firmware stores the interrupting device address, CPU status, and the current program counter. The service routine vector then is placed into the Program Counter so that the next instruction fetch is from that address. The service routine then interrogates for an abort condition and if none, processes the I/O transmission.

Block Transfers with the Block Transfer Adapter. The following example illustrates the setup of a block transfer on the BTA. Note that the status from the previous transfer must be checked to determine that the device is ready for another transfer. The example is written as a generalized subroutine and demonstrates the use of the auto-increment mode to fetch direct and indirect parameters from an argument list.

BTA Input-Output Example

```
        JSBR    BTA,R1      Jump to BTA subroutine, save return
                              in R1
        ADDR    STATUS      Address of device status register
        ADDR    BUFADD      Address of buffer in memory
        ADDR    ADDRES      Address of BTA Address register
        DATA    SIZE        Length of block to be transferred
        ADDR    BLOCK       Address of BTA Block Length register
        DATA    IO          I/O code = 1 for input; = 3 for output
        ADDR    CONTROL     Address of device Control register
          .
          .
BTA     REGM    SAVREG      Enter subroutine; save registers R1-R7
BUSY    MOVW    *(R1),R3    Move status register into R3 to check
                              status of previous operation
        BODF    BUSY        If bit 0 not set, go to busy routine
        ADDW    =2,R1       Increment R1 to the Buffer Address
                              pointer
        MOVW    (R1+),R4    Move buffer address into R4
        MOVW    R4,*(R1+)   Move R4 into the BTA Address register
        MOVW    (R1+),R4    Move the block length into R4
        MOVW    R4,*(R1+)   Move R4 into the BTA Block Length
                              register
        MOVW    (R1+),R4    Move the I/O code into R4
        ADDW    =H)80,R4    Set the BTA transfer bit
        MOVW    R4,*(R1+)   Move R4 into the Control register to
                              start the block transfer
        MOVW    R1,RETURN
        MREG    SAVREG
        JUMP    *RETURN     Return from subroutine
```

CHAPTER 10

SYSTEM SOFTWARE

SUE software helps the user develop application programs. The programmer can write in assembly language, assemble, debug, and run on a variety of available machines. SUE system software features are:

Programs that run on SUE with 4K words of memory and an ASR-33 Teleprinter:

A comprehensive one-pass assembler that produces relocatable object code.

A relocating Link Loader that produces an executable translation of a main program and links external subroutines to it.

A Basic Loader that loads the output from the Link Loader into memory for execution.

A conversational debug program for on-line test and modification of assembled programs.

An I/O control system for communication between programs and peripheral devices.

Operator utility routines that interface between the program and the operator.

Test and maintenance programs for fast field analysis and repair of faults.

Programs that run on an IBM 360 or a Lockheed Electronics' MAC 16:

SUE Cross Assembler for listings and assembled code output identical to the SUE assembler.

SUE Link Loader that builds relocatable binary-formatted output for loading by the Basic Loader on the SUE processor.

Programs written in FORTRAN to run on a variety of machines:

SUE simulator for execution and testing of SUE-assembled object code on the IBM 360 computer or any large-scale computer with a ANSI-standard-FORTRAN Compiler.

SUE ASSEMBLERS (LAP-2)

The assembler operates on a SUE computer with 4K words of memory and an ASR-33 Teletypewriter. An expanded version of the assembler that has additional features and operates additional peripherals can be used on machines of increased memory capacity. All assemblers for SUE are one pass, producing object code for the Link Loader. If additional peripherals are available an assembly listing is produced on the same pass; if not, then a listing pass is required. A Diagnostic Only option provides a listing of those statements in error.

Two cross assemblers are available for SUE. One operates on the Lockheed Electronics' MAC 16 Computer, the other on IBM 360 computers. Cross assemblers provide the user with assembly capability on readily-accessible processors having high-speed peripherals. These cross assemblers function identically to the SUE assembler and produce the same listing and object code.

An expanded assembler has many features not normally found in a minicomputer assembler. Some of these are

Full macro capability.
Fixed-point decimal conversion, single and double precision.
Floating-point decimal conversion, single and double precision.
Conditional assembly directives.
Listing formatting directives (EJECT, SPACE, etc.)
New operation definition capability (to allow assembling special op-codes implemented in a customized control ROM.

SUE LINK LOADERS

The SUE Link Loader is a relocating loader capable of building a core load by linking a main program and external subroutines. The loader accepts the output from the SUE assembler and generates output for loading by the Basic Loader. The operator may enter a relocation constant for changing the memory location of the linked program. Options include forcing the Link Loader to completion when external references remain undefined but are not necessary for the initial test runs; printing a memory map of the core load to provide the programmer with a reference for easy access of program modules; and defining externals not included in the subroutines.

The Cross Link Loaders that run on the MAC 16 and IBM 360 processors combine with the cross assemblers to provide a complete program generation system. The output can be loaded into the SUE computer for execution or loaded into the simulated memory of the SUE Simulator for execution and test.

SUE BASIC LOADER (BLOD-2)

The SUE Basic Loader loads the output generated by the Link Loaders into memory for execution. Record-by-record checking is performed with error detection causing an immediate halt to the system. Both Load and Go or Load and Halt operations are provided.

SUE BASIC OPERATING SYSTEM (BOS)

BOS serves as an off-line aid to the programmer when testing a new program. Some features included:

Change a word or byte in memory.
Execute a selected portion of the program.
Search the program for a key bit pattern.
Dump memory to the printer.
Dump memory in Basic Loader format to the punch.

SUE INPUT/OUTPUT CONTROL SYSTEM (IOCS)

IOCS provides a centralized I/O package that frees the user from details of dealing directly with peripheral devices. IOCS allows concurrent I/O operation of multiple devices and provides device independence to the user through assignment of device logical unit numbers to the various I/O devices at execution time. The user calls IOCS from a calling sequence that uses a parameter list to define the requested operation. The parameter list offers several options to the programmer such as wait or no-wait for I/O completion and, upon device error, re-try or don't re-try the request. At the completion of any requested operation IOCS returns to the calling function for further processing.

SUE OPERATOR UTILITY INTERFACE PACKAGE (OUIP)

OUIP provides program-to-operator and operator-to-program communication. This package operates in conjunction with IOCS and provides the following functions to the user:

Input data from keyboard
Fetch name
Fetch numeric
Print message
Print numeric
Print carriage return/line feed
Print space
Print character
Input symbolic source line
Input binary formatted record
Output symbolic source line
Output binary formatted record
Program return

The user program can call any of these routines for ease in communicating with I/O devices. All symbolic and binary routines are interrupt driven and double-buffered. Each allows operator assignment of the desired peripheral for flexibility.

TEST AND MAINTENANCE PROGRAMS

The SUE Maintenance System is easy to use, rapid, and provides the user with detailed testing capability, resulting in minimum down time.

Central Processor Unit Test

The CPU Test Program explores all the functional characteristics of each machine instruction. Test begins with simple functions and increases the complexity of each function until all aspects of each instruction have been checked. All applicable interrupt functions are also included.

Memory Test Programs

The Memory Test Programs exercise all memory locations using various patterns and techniques that find the cause of any dynamic memory failure.

1. Memory Address Test writes every memory location with its address, then reads and verifies each location.
2. Worst Pattern Memory Test writes every memory location with all ONEs or all ZEROs depending on the address being accessed, then reads and verifies each location.
3. Random Data Memory Test writes a random data pattern in each memory, then reads and verifies each location.

Peripheral Equipment Tests

A Peripheral Test Program is available for each peripheral device. Each test verifies the operational status of each device and its capability for performing in the system.

1. Teletypewriter
2. High-speed paper tape reader/punch
3. Mag tape (reel/cartridge)
4. Line printer
5. Card reader

The SUE programs listed operate with paper tape equipment. The assembler, for example, has the I/O executive, drivers, and operator executive assembled and link loaded into a load-and-go object tape. These programs are delivered complete with listing, tapes, specification, and user manuals.

SYSTEM GENERATION

An application system that will run on the SUE computer can be generated on an IBM 360 processor or on the SUE computer.

System generation on the IBM 360 is accomplished using:

IBM 360 Cross Assembler program
IBM 360 Link Loader program
IBM 360 Simulator program

System generation on the SUE processor is accomplished using:

SUE Assembler program
SUE Link Loader program
SUE Debug program

Individual application programs, as they are coded and punched into source tapes (decks), are assembled, link loaded, and debugged using one of the development systems described. This development is aided by I/O service routines that are in the Link Loader format. This means that the programmer can incorporate the developed and tested Input/Output Control System (IOCS) along with the device service routines directly into his program. The routines are called as EXTERNALS in a program and the library tape is supplied when called for by the Link Loader.

SUE SIMULATOR

The SUE Simulator is written in ANSI-standard FORTRAN to allow it to run on any computer that supports this standard or, with minor modifications, for any computer having a FORTRAN IV compiler.

This simulator provides 4K bytes of simulated memory and simulates the full hardware capability, including Input/Output of the SUE computer. Features that have been implemented include:

Break-point dumps of simulated memory.
Elapsed time counters for subroutine timing.
Various tracing modes.
Easy addition of new machine instructions, to accommodate special ROMs.
Patching capability, to modify programs without reassembling.

BUSINESS SYSTEM SOFTWARE (BUNDLED)

REPORT PROGRAM GENERATOR (RPG II)

The RPG II Compiler for SUE provides approximately 95% System 3 Model 6 RPG II and 90% IBM 360/20 RPG compatibility. Programs written for either system can be easily modified (generally control cards) to run on the Business System.

The RPG II compiler allows program segmenting and both foreground and background program generation. Extensions to core memory are automatically utilized by RPG II.

DISK OPERATING SYSTEM (DOS)

The Disk Operating System for SUE is the most complete minicomputer DOS in existence. It provides batch processing, foreground/background program operation, background program roll-out/roll-in (check point), program segmenting, operator intervention, extensive error detection and recovery, File management, Data management, and a full line of utility support programs. DOS was designed with the user in mind. A straight-forward conversational technique is used for all operator communications.

DISK SORT/MERGE

The Disk Sort/Merge package was designed to be utilized with RPG II. No changes are necessary when existing RPG II programs are compiled and executed on the Business System.

CHAPTER 11

SUE CONTROL PANELS

SUE control panels are System User Engineered to provide the maximum of flexibility in system design. The control panel operates as a peripheral device with its own Infibus interface, and is normally installed in a system only when required for maintenance or system program development. A removable panel provides added security and lower cost to the system user. SUE control panels may be hinged on either the front or rear, left or right side of the chassis. The panel may also be remotely mounted up to 20 feet from the SUE chassis.

Control panels offered are:

- o System Control Panel (SUE 2215)
- o Program Maintenance Panel (SUE 2220)

A feature of SUE control panels, not found on most minicomputers, is the ability to directly address and control peripheral controllers as well as all other modules on the Infibus.

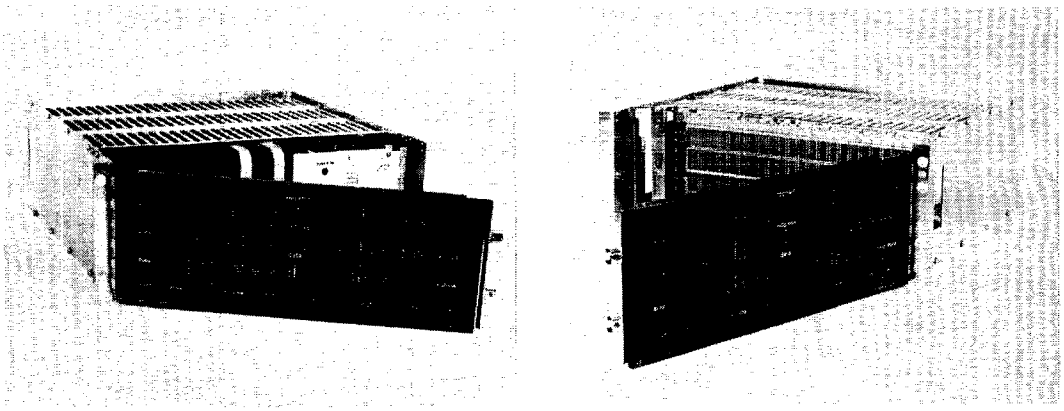


Figure 11-1. Alternate Control Panel Mounted

PROGRAM MAINTENANCE CONTROL PANEL (SUE 2220)

The SUE 2220 control panel provides complete system control and functional convenience. It is designed to be hinged on the SUE chassis or to operate remotely from the processor. The 2220 uses unique touch panel switches for easy operation and high reliability. Indicators are light emitting diodes for low power and long life. Control panel logic is located on two printed circuit cards that are inserted in the Infibus. Flat ribbon cable connectors on the outer edge of these cards provide interconnect to the panel.

Controls and Indicators

The 2220 has the following switches and indicator lights.

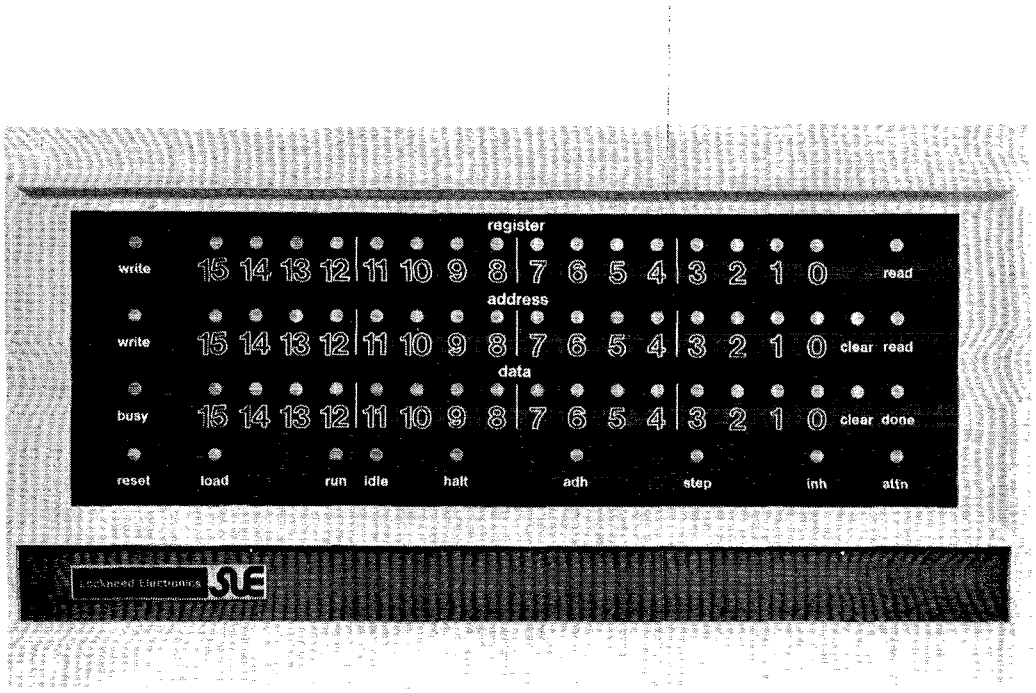


Figure 11-2. Program Maintenance Control Panel

Touch Response Switches

- REGISTER Selections
- A 16-bit ADDRESS plus CLEAR
- A 16-bit DATA plus CLEAR
- READ and WRITE for REGISTER Selections
- READ and WRITE for ADDRESS
- Control function switches:
  - o Operator Attention (ATTN)
  - o Inhibit all External Interrupts (INH)
  - o Program Step (STEP)
  - o Address Halt (ADH)
  - o Halt Processor (HALT)
  - o RUN Processor (RUN)
  - o Auto-load (LOAD)
  - o Master Reset (RESET)

Indicators. Indicators perform the dual-function of presenting a status or condition and indicating completion of switch operation. Additional indicators are:

- Panel operation complete (DONE)
- Infibus Busy (BUSY)
- All Processor Units Idle (IDLE)

Switch and Indicator Functions

Control Switches perform the following special functions:

- ATTN interrupts the central processor on level one.
- INH inhibits interrupts 1 through 4. Resetting the indicator allows interrupts 1 through 4.

- STEP, when depressed during processor operation, causes a processor halt following the execution of one instruction. The selected register is automatically displayed in the data register. Pressing the step switch when the processor is halted, causes the execution of one instruction at the location specified by the program counter.
- ADH halts the processor when the address specified by the Address register is detected on the Infibus address lines. The control panel continues to monitor infibus addresses until the ADH function is terminated by pressing the ADH switch again. The ADH light signifies ADH functions active.
- HALT halts the processor and lights the indicator light. If the processor is already in a halt state, pressing the switch has no effect.
- RUN starts the execution of program instructions at the address specified by the Program Counter (R0). NOTE: The RUN, HALT, and IDLE lights have related definitions. See discussion on IDLE indicator.
- LOAD initiates automatic loading of one block of data from the designated input device. Requires pressing the RESET switch prior to its operation.
- RESET performs a master reset to the system and arms the LOAD switch and lights its indicator. The Infibus and all system modules are effectively inhibited from operating.

Control Indicators present the following system conditions:

DONE indicates the control panel access to the Infibus has been completed.

BUSY indicates the Infibus is currently busy with system traffic. This light is seldom extinguished while the system is operating.

RUN, HALT, and IDLE indicators have interrelated definitions. The HALT light indicates the processor is not executing instructions and is not idle awaiting an operation completion. The RUN light indicates the processor is fetching and executing instructions. The IDLE light indicates the processor is not executing instructions, but is waiting for some function to complete operation, or the CPU is executing a WAIT instruction.

Register Selection

Once the processor has been halted, the contents of selected processor registers can be displayed and changed. The following table is a list of the registers and its associated switch located in the top row on the 2220 panel.

Register	Switch
Program Counter (R0)	0
General Purpose Register (R1)	1
General Purpose Register (R2)	2
General Purpose Register (R3)	3
General Purpose Register (R4)	4
General Purpose Register (R5)	5
General Purpose Register (R6)	6
General Purpose Register (R7)	7
Processor Status Register	8
Processor Instruction Register	9
Address of Last Instruction Executed	10
Operand Address of Last Memory Reference Instruction Executed	11

Pressing the selected register switch, sets its indicator and resets all other indicators in the row.

To read, press the register READ switch and the contents of the selected register are displayed in the data register.

To write place the data to be written into the data register and press the register WRITE switch.

Address Selection

Memory and device modules may be addressed, read out and written into without halting the processor module.

For Address selection enter into the address register (middle row) the register or memory address. Verify the correct address by relating the bit indicators to the ONE bits of the address. The CLEAR switch resets all the Address register bits to ZERO.

To read, press the READ switch located at the right end of the middle row. The contents of the addressed location is displayed in the Data register.

To read consecutive locations, press READ again.

To write, enter the data to be written into the data register switches. Verify the lighted indicators correspond to the ONE bits of the data. Press WRITE switch located at left-end of middle row. The contents of the Data register is written into the addressed location.

Data Selection

The data register is used to display or hold the data that is read or written to and from memory or a register. The CLEAR switch located to the right of bit 0, clears all data register bits to ZERO.

The data and address selection switches are momentary switches with alternating action. This means, that consecutive pushes will alternate the condition from set to reset or reset to set, depending on the initial condition.

Alarm Interrupt Switches

Three switches are located on the back side of the switch panel assembly; power fail, power recovery and line frequency. Power fail and line frequency are two position switches that enable interrupts and power recovery is a three-position switch that enables an interrupt or an autoload function when power is restored.

Program Load

The auto-load (LOAD) switch on the 2220 panel initiates execution of the 1240 Autoload program. This program is contained in a ROM and can be a standard loader program or a customer-provided program. RESET must be pressed prior to the operation of the auto-load function.

Messages

Control panel data and address registers can be addressed by the processor or any master system module. This provides a technique for displaying operator messages.

SYSTEM CONTROL PANEL (SUE 2215)

The SUE 2215 control panel provides system control and functional convenience. It is designed to be installed on the SUE chassis or for remote operation up to 20 feet. Indicators are light emitting diodes for low power and long life. Control panel logic is located on a printed circuit card that is inserted in the Infibus. Flat ribbon cable connectors on the outer edge of the card provide interconnect to the switch panel.

Controls and Indicators

The 2215 has the following switches and indicator lights.

Switches

- A 16-bit ADDRESS or DATA Selection
- ADDRESS/DATA Selection
- READ and WRITE for ADDRESS or DATA Select
- Control function switches:
  - o Operator Attention (ATTN)
  - o Load Register (LR)
  - o Inhibit all Interrupts (INH)
  - o Halt Processor (HALT)
  - o RUN Processor (RUN)
  - o Auto-Load (LOAD)
  - o Master Reset (RESET)
  - o Power Fail-Restart and Line Frequency, ON-OFF (PF/PR/LF)

Indicators. Sixteen indicators provide address or data display information. Additional indicators are:

- Panel operation complete (DONE)
- Processor Unit Idle (IDLE)

Switch and Indicator Functions

Control Switches perform the following special functions:

- ATTN interrupts the central processor on level one.
- INH inhibits interrupts 1 through 4. Resetting the indicator allows interrupts 1 through 4.
- HALT halts the processor and lights the indicator light. If the processor is already in a halt state, pressing the switch has no effect.
- RUN starts the execution of program instructions at the address specified by the Program Counter (R0). NOTE: The RUN, HALT, and IDLE lights have related definitions. See discussion on IDLE indicator.
- LOAD initiates automatic loading of one block of data from the designated input device. Requires pressing the RESET switch prior to its operation.

- RESET performs a master reset to the system, arms the LOAD switch and lights its indicator. The Infibus and all system modules are effectively inhibited from operating.
- PF/PR/LF ON-OFF disables power supply interrupts in the off position.

Control Indicators present the following system conditions:

- DONE indicates the control panel access to the Infibus has been completed.
- RUN, HALT, and IDLE indicators have interrelated definitions. The HALT light indicates the processor is not executing instructions and is not idle awaiting an operation completion. The RUN light indicates the processor is fetching and executing instructions. The IDLE light indicates the processor is not executing instructions, but is waiting for some function to complete operation, or the CPU is executing a WAIT instruction.

Address Selection

Memory and device modules may be addressed, read out and written into without previously halting the processor module.

For address selection set the address selection switch to ADD, enter into the address/data register, the register or memory address.

To read set the READ switch to READ and the contents of the addressed location is displayed in the register indicators.

To write set the address select switches to ADD, enter the address to be altered into the address/data switches. Set data select switch to DATA and enter the data to be written into the address data register switches. Set WRITE switch to WRITE and the contents of the address/data register switches is written into the addressed location.

Data Selection

The address/data register is used to display or hold the data that is read from memory or a register. Data that is written into memory or a register is held by the switch positions.

The address/data switches are switches with alternating action. Up is a one bit, down is a zero bit.

POWER DISTRIBUTION UNIT (SUE 2201)

The SUE 2201 Power Distribution Unit Keylock provides a key-operated, three-contact switch that can be locked into the OFF position, ON position, or the ENABLE position.

The 1-3/4 inch panel attaches to the base of the SUE chassis behind the Programmer panel. The 2201 also provides four a-c power outlets for convenient power distribution.

Functions

OFF Position. In the OFF position power to the system is disconnected. The Infibus Controller senses loss of a-c power to the Infibus and interrupts the highest-priority processor on level four.

ON Position. In the ON position power is supplied to the system. The Infibus Controller senses d-c power to the Infibus and interrupts the highest-priority processor on level four.

ENABLE Position. This position enables the panel switches and functions.

The system can be operated without a control panel. The On/Off function can be achieved manually by connecting or disconnecting the system power source. This approach relies on the automatic line-fail and line-recovery feature of the Infibus Controller to preserve the memory contents and reinitialize the system at power-up time.

POWER DISTRIBUTION UNIT, KEYLOCK, SELECT (SUE 2202)

The SUE 2202 Power Distribution Unit has all the features of the 2201. It has an additional key-operated selection switch that selects one of four processors or one of four autoload programs to be operated by the 2220 control panel.

CHAPTER 12

SUE SYSTEM OPTIONS

The SUE System is the ultimate in modular flexibility. The user engineer can select the precise system modules required for any application. As system requirements grow or change, modules can be added, deleted or changed without affecting the operation of other system components. Listed below are the categories of SUE System options contained in this chapter.

- Processors and Options
- Control Panels
- Memory (Program Memories)
- Input/Output Device Controllers
  - General Purpose I/O Controllers
  - Device Controllers
- System Power Supplies
- Peripheral Devices
- System Cables and Cable Accessories
- System Equipment

PROCESSORS AND OPTIONS

SUE Model	Description
1110	Standard Processor Unit - Eight general purpose registers, basic instruction set, multiple address modes; ROM control memory, word and byte addressing, four shared levels of programmable interrupts. Occupies two module slots. 420 CFM minimum distributed airflow required.
1112	Scientific Double Precision Processor Unit - Eight general purpose registers, the basic mnemonics and address modes of the 1110 processor, plus 36 instructions, including multiply, divide and double register shifts; ROM control memory, word and byte addressing, four shared levels of programmable interrupts. Occupies two module slots. 420 CFM minimum distributed airflow required.

PROCESSORS AND OPTIONS (continued)

SUE Model	Description
1240	Auto-Load, General - Automatically loads memory from a specified peripheral device. Customer furnishes control ROM/PROM memory. Up to 4 different ROM/PROM memories can be used for auto-load. Occupies one slot.
	The following auto-load options are preconfigured 1240 modules designed for use with LEC standard peripherals as noted:
1241	Auto-Load - Paper Tape, Teletypewriter Model 6710; High Speed Paper Tape Reader Models 6714 and 6715.
1242	Auto-Load - Disc, removable media Model 6750.
1243	Auto-Load - Card Reader Model 6736.
1244	Auto-Load - Magnetic Tape Transport Model 6794; can be used with cassette if 4590 BTA is used in conjunction with 4608 Controller.
	Multiple Device Auto-Load Options*
1251	Dual Auto-Load - Paper tape as in 1241 above plus disc device as in 1242.
1252	Dual Auto-Load - Paper tape as in 1241 above plus card reader as in 1243.
1253	Dual Auto-Load - Disc as in 1242 above plus card reader as in 1243 above.
1261	Triple Auto-Load - Paper tape, disc unit and card reader.

*Auto-load device selection is via code select switches mounted on auto-load circuit board. Device auto-load selection is available on the Model 2202-2 Power Distribution Unit.

PROCESSORS AND OPTIONS (continued)

SUE Model	Description
1810	Infibus Controller - Directs communications on and allocates access to the Infibus. Includes 25 MHz reference clock and response to power failure, power on restart and line frequency clock. One required per SUE System; occupies one slot.
1827	Infibus Driver-Receiver Kit - Consists of one mainframe driver module, one external chassis receiver module and two 7330 cables for connection to external chassis. Standard length of interconnect cables is three feet (17 feet maximum additional cable length).

CONTROL PANELS

2201	Power Distribution Unit - 1-3/4 inch panel contains three-position keylock switches and four ac convenience outlets. Mounts below control panel. 15 amperes ac maximum.
2202	Power Distribution Unit - 1-3/4 inch panel similar to 2201 except provided with an additional four-position selector switch. For control panel Model 2220 operations where more than one processor is installed in system, order Model 2202-1. For multi-device auto-load configurations, order Model 2202-2.
2215	System Control Panel - 7-inch high panel with a single 16-bit display register, 16 toggle switches for address or data register and eight additional control switches. Requires one slot.
2220	Program Maintenance Control Panel - 7-inch high panel provides three separate displays and touch switches for data, address and register select. Complete selection and control of front panel operations. Requires two slots. For remote panel operation, maximum cable length is 20 feet.
2280	Filler Panel - 7-inch blank filler panel to provide finished look to "No-Control Pnael" configurations. Snap-on panel can be used with any one of the bezels.

CONTROL PANELS (continued)

SUE Model	Description
2293	Decorative Bezel - 8-3/4 inch bezel to add finished appearance to control panels without Model 2201 power distribution unit, and including a 7921 fan pack.
2294	Same as 2293 except with provision for Model 2201 power distribution unit.
2295	Same as 2293 except with provision for Model 2202 power distribution unit.
2296	Decorative Bezel - 7-inch bezel providing a decorative trim for system configurations without fan pack or power distribution options.

MEMORY

PROGRAM MEMORIES

3311	Random Access Magnetic Core Memory with 4096 16-bit words at 850 ± 25-nanosecond access time. Expandable in 4k increments. Occupies three slots.
3312	Random Access Magnetic Core Memory with 8192 16-bit words at 950 ± 25-nanosecond access time. Expandable in 8k increments. Occupies three slots.

INPUT/OUTPUT AND DEVICE CONTROLLERS

GENERAL PURPOSE I/O CONTROLLERS

4501	Parallel I/O Controller - general purpose 16-bit parallel half-duplex control. TTL compatible high true I/O logic interface. Can be used with Block Transfer Adapter 4590. One slot required. Mating connectors available, specify Models 7771 and 7772 as required.

INPUT/OUTPUT AND DEVICE CONTROLLERS (continued)

GENERAL PURPOSE I/O CONTROLLERS (continued)

SUE Model	Description
4502	Serial I/O Controller - general purpose communications control. Contains 20 mA polar and RS232C asynchronous interface half-duplex control. Both data rate and data format are selectable and may be configured by customer. Can be used with Block Transfer Adapter 4590. One slot required. Mating connector available, specify Model 7770 as required.
4503	Parallel I/O Controller - same as 4501 except low true I/O logic interface.
4506	Parallel I/O Controller - same as 4501 except low true input logic, and high true output logic.
4507	Parallel I/O Controller - same as 4501 except high true input logic and low true output logic.
4550	Custom Bus Interface - Provides a module with Infibus Interface Logic on a printed circuit card with room for additional components to be wire wrapped for special applications. This module communicates in both master and slave modes and can be made to operate with the 4590 Block Transfer Adapter. Requires one slot.
4590	Block Transfer Adapter - Allows direct block transfer to and from devices and memory. Occupies one slot and is installed adjacent to the I/O controller.

DEVICE CONTROLLERS

4601	High Speed Paper Tape Reader and Punch Controller for operation with Models 6714, 6715, 6716 and 6719. Can operate with 4590. Occupies one slot.
4602	Line Printer Controller for Model 6762; can be operated with 4590. Occupies one slot.
4603	Card Reader Controller for Model 6736; can operate with 4590. Occupies one slot.

INPUT/OUTPUT AND DEVICE CONTROLLERS (continued)

DEVICE CONTROLLERS (continued)

SUE Model	Description
4605	Line Printer Controller for Model 6768; can be operated with 4590. Occupies one slot.
4608	Cassette and Magnetic Tape Controller for Models 6780, 6781, 6782 and 6783 cassettes and Model 6790 Magnetic Tape Formatter. A 4590 Block Transfer Adapter is mandatory when the 4608 is used with the 6790 Magnetic Tape Formatter.
4630	Teletypewriter Controller for Models 6710 and 6721; can be operated with 4590.
4751	Disc Storage Controller for Model 6750 disc drive. Infibus and disc interface for IBM 5444 hardware compatible format provided with block transfer capability. Controls up to four Model 6750 disc drive units. Occupies three slots.

SYSTEM POWER SUPPLIES

5951	Internal Power Supply - Heavy duty plug-in power supply. Interchangeable with Model 5955 power supply. Provides +5 vdc @ 38 amperes, +15 vdc @ 7 amperes, -15 vdc @ 3 amperes. 9 amperes vac current required. Contains power fail/auto restart and line frequency pulse generator. 7720 cable required for external mounting.
5952	External Power Supply - 7-inch external rack-mounted heavy duty power supply. Provides +5 vdc @ 50 amperes, +15 vdc @ 25 amperes, -15 vdc @ 5 amperes. 20 amperes vac current required. Provided with 3-terminal Hubble twistlock connector. Contains power fail/auto restart and line frequency generator. External power cable to Infibus included.
5955	Internal Power Supply - Standard plug-in power supply. Interchangeable with Model 5951 Power Supply. Provides +5 vdc @ 18 amperes, +15 vdc @ 5 amperes, -15 vdc @ 1 ampere. 9 amperes vac current required. Contains power fail/auto restart and line frequency pulse generator.

PERIPHERAL DEVICES

Each peripheral device is furnished with an I/O cable suitable for interconnection between LEC standard controllers and peripheral devices.

SUE Model	Description	Prerequisites
6710	ASR-33 Teletypewriter.	4630
6720	ASR-33 Teletypewriter with pin feed platen.	4630
6714	High-Speed Paper Tape Reader with spoolers. Reads at 300 characters per second.	4601
6715	Same as 6714 except without spoolers.	4601
6716	High-Speed Paper Tape Punch with spoolers; accepts 8-level tape at 75 characters per second.	4601
6719	Combination High-Speed Paper Tape Reader and Punch. Includes a 300-character-per-second reader and a 75-character-per-second tape perforator.	4601
6736	Card Reader, 80-column, 600 cpm.	4603
6750	Disc Storage Unit, IBM 5444 Compatible - Includes one fixed disc and accommodates one removable disc cartridge (IBM 5440 top loading type). 2.5 million-byte capacity per disc; data transfers at 198K bytes per second. Requires 32 inches of cabinet depth.	4751 6757
6757	Removable Disc Cartridge (IBM 5440 top loading type).	6750
6762-1	Printer Terminal - 132-column, 64-character set, 100 characters per second print rate, up to six (6) copies, without stand.	4602
6762-2	Same as 6762-1 with stand.	4602
6768	Line Printer - 132-column, 64-character set, 600 lpm.	4605

PERIPHERAL DEVICES (continued)

SUE Model	Description	Pre-requisites
6780	Cassette Tape Unit - Single drive unit up to 720,000 characters (bytes) storage capacity depending upon record length. Data transfers at 600 characters per second.	4608
6781	Cassette Tape Unit, double drive unit - up to 1,440,000 characters (bytes) storage capacity depending upon record length. Data transfers at 600 characters per second.	4608
6782	Cassette Tape Unit, triple drive unit - up to 2,160,000 characters (bytes) storage capacity depending upon record length. Data transfers at 600 characters per second.	4608
6783	Cassette Tape Unit, quad drive unit - up to 2,880,000 characters (bytes) storage capacity depending upon record length. Data transfers at 600 characters per second.	4608
6790	Magnetic Tape Formatter - NRZI tape formatter. Use with Model 6794 mangetic tape transports.	4608 4590
6794	Magnetic Tape Transport - 10-1/2 inch reels, 9-track, 800 BPI, 2 IPS to 45 IPS, NRZI recording format. Data transfers to 36,000 cps. 75 IPS tape speed optional.	6790

SYSTEM CABLES AND CABLE ACCESSORIES

SUE Model	Description
7710-1	Teletypewriter I/O Cable, 8-foot standard - Provides I/O connection to Model 4630 controller and Model 6710 Teletypewriter. For extended cable lengths, specify desired total length as follows: -2 25 feet, extended TTY cable -3 50 feet, extended TTY cable -4 100 feet, extended TTY cable -5 150 feet, extended TTY cable -6 200 feet, extended TTY cable
7720	External Power Supply Cable Assembly - For use with Model 5951 power supply when installed in external chassis. Provides dc voltage interconnection between chassis. Standard length is 24 inches.
7730	Signal Cable - 50-wire shielded flat ribbon cable (48 signal, 2 ground) for internal system signals. Provided with female connector both ends. Standard length is three feet. Maximum length is 20 feet.
7770-1, -2	Connector Kit, Dual 10 - Provides two rows of 10 terminations each. Specify either discrete wire (-1) or flat ribbon (-2) type cable. Can be used with Model 4502 controller.
7771-1	Connector Kit, Dual 20 - Provides two rows of 20 terminations each. For use with discrete wire type cable. If used with Models 4501, 4503, 4506 and 4507 controllers, only twisted pair cable size 24 or 26 AWG wire should be used.
7771-2	Connector Kit, Dual 20 - Provides two rows of 20 terminations each for use with flat ribbon-type cable.
7772-1	Connector Kit, Dual 25 - Provides two rows of 25 terminations each. For use with discrete wire type cable. If used with Models 4501, 4503, 4506 and 4507 controllers, only twisted pair cable size 24 or 26 AWG wire should be used.
7772-2	Connector Kit, Dual 25 - Provides two rows of 25 terminations each. For use with flat ribbon type cable.

SYSTEM EQUIPMENT

SUE Model	Description
7905	Card Guide Frame - Fundamental mechanical assembly without Infibus structure. Includes only card guides and necessary mounting hardware with power supply connector; can be used to externally mount 5951 Internal Power Supply.
7910	Chassis Assembly - 7-inch general purpose Card Guide Frame including 16 Slot Infibus for housing system modules, including space for internal power supply and memory. Contains Infibus with 16 slots.
7911	Chassis Assembly - 7-inch general purpose Card Guide Frame including 24 Slot Infibus with 24 connectors for system modules and memory modules. Power Supply Model 5951 can be connected via 7720 cable.
7921	Fan Pack Assembly - Fan assembly providing 420 CFM airflow minimum. Mounts below 7910 or 7911 chassis; adds 1.75 inches to overall chassis height.
7922	Fan Pack Assembly - Quiet fan assembly providing 300 CFM airflow minimum. Mounts below 7910 or 7911 chassis; adds 1.75 inches to overall chassis height.
7930	Table Top Cabinet - Mounting space for 8-3/4 inch chassis. For typical SUE control panel with bezel and 7910 or 7911 chassis. Requires 7921 Fan Pack and 2201 or 2202 Power Distribution Unit.
7932	Low-boy Console Cabinet - EIA standard 19-inch mounting, 28-1/2 inches high. Available vertical space is 19 inches. Cabinet depth is 32 inches. Will house 6750 disc unit.
7935	Equipment Cabinet, EIA standard 19-inch mounting, 60 inches high. Available vertical mounting space is 52.5 inches. Cabinet depth is 24 inches.
7936	Equipment Cabinet, EIA standard 19-inch mounting, 60 inches high. Available vertical mounting space is 52.5 inches. Cabinet depth is 32 inches. Will house 6750 disc unit.

SYSTEM EQUIPMENT (continued)

SUE Model	Description
7970	Module Extender Board - Provides convenient means of module troubleshooting by extending module to outside of chassis.
7971	Module Extractor - Enables easy card extraction from chassis-mounted printed circuit connectors.
7980	Universal Logic Board - Provides universal circuit board for customer-designed and implemented logic circuits. Mounting space for 98 DIPs. 1,000 wire-wrap pins available, order Model 7985. Occupies one module slot.
7985	Terminal Pin Kit - 1,000 wire-wrap pins for use on Model 7980 Universal Logic Board.
7988	Teletypewriter Modification Kit - Parts and instructions are provided for modification of customer-owned ASR-33 Model TC Teletypewriter for operation with SUE controller Model 4630.

SUE SYSTEM CONSIDERATIONS

CHASSIS

Two basic chassis versions are available for housing SUE System modules: Model 7910 incorporates space for the 5955 or 5951 internal power supply and has a total of 16 slots available. Model 7911 is designed for used with the 5952 external power supply. The 7911 has a total of 24 slots available.

Both chassis are available with alternate control panel mountings. Standard mounting provides rear access to system modules. The alternate mounting provides front-of-chassis access to system modules.

PHYSICAL SIZE AND WEIGHT

Size of basic chassis: 7 inches high by 17.5 inches wide by 18 inches deep.

Approximate weight, with CPU, memory, 5951 power supply, option boards, 7910 chassis and control panel: 70 pounds.

AC POWER

Power - 105 to 125 vac @ 9 amperes maximum for 5955 and 5951 power supply and 20 amperes maximum for 5952 power supply, line frequency: 47 to 63 Hertz.

COOLING

Cooling - attachable fan pack assembly (7921) or customer-furnished airflow of 420 CFM minimum, ambient 0^{o}-50^{o}C.

- 7922 very quiet fan 300 CFM requires an ambient of 0^{o}-40^{o}C.

APPENDIX A

BASIC MNEMONIC LISTING FOR SUE INSTRUCTIONS

BASIC MNEMONIC	HEX OP CODE*	INSTRUCTION DESCRIPTION	TIME (μs)	
ADDB	XA00	Add Byte	2.79**	
ADDW	X200	Add Word	2.79**	
ANDB	XB00	And Byte	2.50**	
ANDW	X300	And Word	2.50**	
			Fall-Thru	Branch
BCYF	8400	Branch if Carry False	1.78	2.72
BCYT	9400	Branch if Carry True	1.78	2.72
BEQF	8100	Branch if Equal False	1.78	2.72
BEQT	9100	Branch if Equal True	1.78	2.72
BF1F	8500	Branch if Flag 1 False	1.78	2.72
BFIT	9500	Branch if Flag 1 True	1.78	2.72
BF2F	8600	Branch if Flag 2 False	1.78	2.72
BF2T	9600	Branch if Flag 2 True	1.78	2.72
BF3F	8700	Branch if Flag 3 False	1.78	2.72
BF3T	9700	Branch if Flag 3 True	1.78	2.72
BGTF	8200	Branch if Greater Than False	1.78	2.72
BGTT	9200	Branch if Greater Than True	1.78	2.72
BLPF	8800	Branch if Loop Complete False	1.78	2.72
BLPT	9800	Branch if Loop Complete True	1.78	2.72
BLTF	8C00	Branch if Less Than False	1.88	3.08
BLTT	9C00	Branch if Less Than True	1.75	3.08
BNGF	8B00	Branch if Negative False	1.78	2.72
BNGT	9B00	Branch if Negative True	1.78	2.72
BODF	8900	Branch if Odd False	1.78	2.72
BOVF	8300	Branch if Overflow False	1.78	2.72
BOVT	9300	Branch if Overflow True	1.78	2.72
BRUN	9000	Branch Unconditional	2.72	
BZEF	8A00	Branch if Zero False	1.78	2.72
BZET	9A00	Branch if Zero True	1.78	2.72

* An X in the first Hex position implies one of several digits.

** Memory Reference Instruction, register-to-register time. Timings for various addressing modes are shown in Table A-1.

BASIC MNEMONIC LISTING FOR SUE INSTRUCTIONS (Continued)

BASIC MNEMONIC	HEX OP CODE*	INSTRUCTION DESCRIPTION	TIME (μs)
CMPB	XE00	Compare Byte	2.69**
CMPW	X600	Compare Word	2.69**
DSBL	0880	Disable Interrupts	1.98
DSBW	08C0	Disable Interrupts and Wait	2.80
ENBL	0800	Enable Interrupts	1.85
ENBW	0840	Enable Interrupts and Wait	2.80
EORB	XD00	Exclusive OR Byte	2.50**
EORW	X500	Exclusive OR Word	2.50**
HALT	0000	Halt	1.01
IORB	XC00	Inclusive OR Byte	2.50**
IORW	X400	Inclusive OR Word	2.50**
JSBR	4000	Jump to Subroutine	1.87**
JUMP	4000	Jump	1.87**
MOVB	X800	Move Byte	2.50**
MOVW	X000	Move Word	2.50**
MREG	0F00	Memory-to-Registers, Relative	8.25
MREG	0700	Memory-to-Registers, Absolute	7.93
MSTS	0D00	Memory-to-Status, Relative	2.79
MSTS	0500	Memory-to-Status, Absolute	2.47
NOPR	8000	No Operation	1.78
REGM	0B00	Registers-to-Memory, Relative	7.56
REGM	0300	Registers-to-Memory, Absolute	7.24
RETN	0C00	Returen from Interrupt, Relative	4.58
RETN	0400	Return from Interrupt, Absolute	4.24
RSTS	0200	Reset Status Indicators	1.59
SETS	0280	Set Status Indicators	1.72
SLAO	A000	Single Left Arithmetic Open	2.76 + .26N
SLLC	A300	Single Left Logical Closed	2.76 + .26N
SLLL	A100	Single Left Logical Linked	2.76 + .26N
SLLO	A200	Single Left Logical Open	2.76 + .26N
SRAO	A400	Single Right Arithmetic Open	2.76 + .26N
SRLC	A700	Single Right Logical Closed	2.76 + .26N
SRLL	A500	Single Right Logical Linked	2.76 + .26N
SRLO	A600	Single Right Logical Open	2.76 + .26N
STSM	0900	Status-to-Memory, Relative	2.46
STSM	0100	Status-to-Memory, Absolute	2.14
SUBB	X900	Subtract Byte	2.79**
SUBW	X100	Subtract Word	2.79**
TSTB	XF00	Test Byte	2.50**
TSTW	X700	Test Word	2.50**

* An X in the first Hex position implies one of several digits.

** Memory Reference Instructions register-to-register time. Timings for various addressing modes are shown in Table A-1.

Table A-1. General Register Instruction Times

General Instruction	Time (Microseconds) Indexed	Auto-Increment	Auto-Decrement
ACCUMULATOR TO MEMORY — Class Codes	3	2	1
Logical: MOV, AND, IOR, EOR			
Op Codes: 0 3 4 5	3.94	4.81	4.81
Arithmetic: SUB, ADD			
Op Codes: 1 2	4.03	4.90	4.90
Compare: CMP			
Op Code: 6	3.70	4.57	4.57
Test: TST			
Op Code: 7	3.35	4.22	4.22
Address Modes:			
For Extended, add 0.13			
For Indirect, add 1.14 for first level, add 1.01 for each additional level			
For Extended, Indirect, add 1.40 for first level, add 1.01 for each additional level			
JUMP, JUMP TO SUBROUTINE — Class Code	4	-	-
Instruction: JUMP, JSBR			
Op Code: 0, AR = 0, AR ≠ 0	2.79	-	-
Address Modes:			
For Extended, add 0.06	2.85	-	-
For Indirect, add 1.14 for first level, add 1.01 for each additional level	3.93	-	-
For Extended, Indirect add 1.33 for first level, add 1.01 for each additional level	4.12	-	-
DATA TO ACCUMULATOR — Class Code	4	-	-
Logical: MOV, AND, IOR, EOR			
Op Codes: 0 3 4 5 Register to	2.50	-	-
Arithmetic: SUB, ADD Register or			
Op Codes: 1 2 Immediate	2.79	-	-
Compare: CMP			
Op Code: 6	2.69	-	-
Test: TST			
Op Code: 7	2.50	-	-
Address Modes:			
For Literal add 0.68			
For Literal Indexed add 0.84			
MEMORY TO ACCUMULATOR — Class Codes	7	6	5
Logical: MOV, AND, IOR, EOR			
Op Codes: 0 3 4 5	3.35	4.09	4.09
Arithmetic: SUB, ADD			
Op Codes: 1 2	3.64	4.38	4.38
Compare: CMP			
Op Code: 6	3.67	4.41	4.41
Test: TST			
Op Code: 7	3.35	4.09	4.09
Address Modes:			
For Extended add 0.13			
For Indirect add 1.14 for first level, add 1.01 for each additional level			
For Extended, Indirect add 1.40 for first level, 1.01 for each additional level			

NOTE: All times are in microseconds.

APPENDIX B

ASSEMBLER DIRECTIVES, 8K VERSION

A. DATA DECLARATION DIRECTIVES

ADDR	Address data
DATA	Word data
BYTE	Byte data
TEXT	Character data

B. SYMBOL DEFINING DIRECTIVES

DEFN	Define symbol
XTRN	Declare, external symbol
NTRY	Declare, entry symbol

C. ASSEMBLY CONTROL DIRECTIVES

CORE	Set program location counter relocatable
CORA	Set program location counter absolute
SAVE	Reserve memory block
EVEN	Set location counter to word address
SKIP	Assemble conditionally
CONT	Continue
LTYP	Set loader type code
ERRX	Announce error message
SYMS	Symbol table save
SYMR	Symbol table reset
MODL	Declare computer MODEL
ENDF	End of file
END	End of source program

D. LISTING CONTROL DIRECTIVES

TITL	Program title
EJCT	Position to next page
SPAC	Space listing "n" lines

E. META ASSEMBLY DIRECTIVES

FORM	Word format specification
MACR	Macro prototype specification
MEND	Macro prototype end

APPENDIX C

Sample Program #1 was written to illustrate a procedure used in coding interrupt-driven programs for the SUE computer. Some of the unique features of the SUE system, as well as special coding techniques, are exemplified as well. Included is the display capability of the 2220 panel, vectored interrupts, shared interrupt device interrogation, real-time clock control, MACRO definition and utilization, subroutine entry and exit, byte manipulation, data definition, stack processing, auto-increment, auto-decrement, and many more.

The operating procedures are simple. After loading the program, a time delay routine of approximately three seconds duration is executed. The time count is displayed on the panel. The operator may then type up to 72 characters (terminated by a carriage return) through the system TTY. These characters will be sorted and repeated on the following line by the program. The direction of the sort (high to low or low to high) depends upon the direction which the indicators of the ADDRESS row on the panel are being sequentially lighted (left to right or right to left). The operator may continue entering lines of characters for sorting at this point. The lighting sequence in the ADDRESS row may be reset to the right side by depressing the operator attention (ATTN) pushbutton. The program may be restarted by turning off power to the SUE and turning it back on.

NOTE

To function properly, the Power Fail, Auto Restart, and Clock switches located on the panel back must be "ON" and the reset push-button must not be depressed.

```
                0002 *
                0003 *      THE FOLLOWING MACRO WILL BE USED TO CONTROL THE OPERATION
                0004 *  OF THE TTY.
                0005 *
                0006 SETUP  MACR
                0007        MOVW    R1,(R7)         CLEAR CONTROLLER
                0008        MOVW    =P)1,R1         SET CONTROL WORD
                0009        MOVW    R1,6(R7)
                0010        MEND
                0011 *
                0012 *
                0013 *      THE FOLLOWING DIRECTIVES DEFINE THE BIT POSITIONS
                0014 *  UTILIZED IN THE SET AND RESET STATUS BIT COMMANDS
                0015 *
A 0010          0016 F1     DEFN    H)10
A 0020          0017 F2     DEFN    H)20
A 0040          0018 F3     DEFN    H)40
A 0008          0019 CARRY  DEFN    H)8
A 0004          0020 OVRFLW DEFN    H)4
                0021 *
                0022 *      THE FOLLOWING DIRECTIVES DEFINE THE INTERRUPT
                0023 *  LINKAGE LOCATIONS WHICH ARE USED BY THIS PROGRAM.
                0024 *
                0025 *
A 0018          0026 LEV4AD DEFN    H)18            LEVEL 4 INTERRUPTING DEVICE ADDRESS
A 001A          0027 LV4STA DEFN    LEV4AD+2        LEVEL 4 INTERRUPT STATUS STORAGE
A 0008          0028 LEV2AD DEFN    H)8             LEVEL 2 INTERRUPTING DEVICE ADDRESS
A 000A          0029 LV2STA DEFN    LEV2AD+2        LEVEL 2 INTERRUPT STATUS STORAGE
A F800          0030 TTY    DEFN    H)F800          TTY CONTROLLER ADDRESS
                0031 *
A 002E          0032 LV6VEC DEFN    H)2E            VECTOR ADDRESS DEFINITIONS
A 0026          0033 LV5VEC DEFN    H)26
A 001E          0034 LV4VEC DEFN    LEV4AD+6
A 0016          0035 LV3VEC DEFN    H)16
A 000E          0036 LV2VEC DEFN    LEV2AD+6
A 0006          0037 LV1VEC DEFN    6
                0038 *
A FF80          0039 ADDRS  DEFN    H)FF80          PANEL ADDRESS LIGHTS
A FF82          0040 DATA   DEFN    H)FF82          PANEL DATA LIGHTS
                0041 *
A 0100          0042 LV2REG DEFN    H)100           ACTIVE LEVEL 2 REGISTER STORAGE
A 010E          0043 LEV2SV DEFN    LV2REG+14       TEMPORARY REGISTER STORAGE
A 011C          0044 COMSTA DEFN    LEV2SV+14       COMMON STATUS STORAGE FOR LEVELS 2&4
                0045 *
A 0200          0046        CORA    H)200           START CORE STORAGE AT 0200
```

```
                    0048 *
                    0049 *       LEVEL 6 - BUS CYCLE ABORT INTERRUPT
                    0050 *
0200 A 0006         0051 LEVEL6 HALT   6
                    0052 *
                    0053 *       LEVEL 5 - UNIMPLEMENTED INSTRUCTION INTERRUPT
                    0054 *
0202 A 0005         0055 LEVEL5 HALT   5
                    0056 *
                    0057 *       LEVEL 4  - POWER FAIL - AUTO RESTART - LINE FREQUENCY
                    0058 *
0204 A 3018         0059 LEVEL4 MOVW   R1,SAVR1         SAVE R1
0206 * 0000
0208 A 7018         0060        MOVW   LEV4AD,R1        CHECK INTERRUPTING DEVICE ADDRESS
020A A 0018
020C A A592         0061        SRLL   R1,2
020E * 8400         0062        BEYF   $1               POWER FAIL ADDRESS = BIT 1
0210 A 0041         0063        HALT   H'41             HALT ON POWER FAILURE
0212 * 8900         0064 $1     BUOF   $8               RE-INITIALIZE IF AUTO RESTART
0214 A 4008         0065        JUMP   INITL
0216 * 0000
0218 * 9800         0066 $8     BNGT   $2               BRANCH IF LINE FREQUENCY INTERRUPT
021A A 0042         0067        HALT   H'42             SOME OTHER INTERRUPT
021C * 9000         0068        BRUN   LEV4OT
021E A 49F1         0069 $2     SUBW   =1,R7            COUNT PULSES AND CHECK FOR COMPLETION
0220 * 8A00         0070        BZEF   LEV4OT           AND IGNORE IF INCOMPLETE
0222 A 058E         0071        MSTS   COMSTA           SET STATUS
0224 A 7078         0072        MOVW   FCOUNT,R7        RESTORE COUNT
0226 * 0000
0228 A 7018         0073        MOVW   FLASH,R1         PICK UP FLASH DATA
022A * 0000
022C * 9500         0074        BF1T   $3               BRANCH IF RIGHT SHIFT
022E * 8800         0075        BNGF   $6               CHECK FOR LEFT END OF REGISTER
0230 A 0290         0076        SETS   F1               AND SET FOR RIGHT SHIFT
0232 * 9000         0077        BRUN   $4
0234 * 9900         0078 $3     BUOT   $5               CHECK RIGHT END OF REGISTER
0236 A A691         0079 $4     SRLO   R1,1             DO RIGHT SHIFT
0238 * 9000         0080        BRUN   $7
023A A 0210         0081 $5     RSTS   F1               SET FOR LEFT SHIFT AND
023C A A291         0082 $6     SLLO   R1,1             DO IT
023E A 3018         0083 $7     MOVW   R1,FLASH         SAVE THE DISPLAY AND
0240 * 0000
0242 A 3018         0084        MOVW   R1,ADDRS         PLACE IN PANEL ADDRESS LIGHTS
0244 A FF80
0246 A 018E         0085        SISM   COMSTA           SAVE COMMON STATUS
0249 A 7018         0086 LEV4OT MOVW   SAVR1,R1         RESTORE R1 AND RETURN
024A * 0000
024C A 040D         0087        RETN   LV4STA
```

```
                    0089 *
                    0090 *       LEVEL 3  - HIGH SPEED DEVICE INTERRUPT
                    0091 *
024E A 0003         0092 LEVEL3 HALT   3
                    0093 *
                    0094 *       LEVEL 2  - LOW SPEED (TTY) DEVICE INTERRUPT
                    0095 *
0250 A 0387         0096 LEVEL2 REGM   LEV2SV          SAVE ALL CURRENT REGISTERS AND
0252 A 0780         0097        MREG   LV2REG          FETCH LEVEL 2 REGISTER STORAGE
0254 A 058E         0098        MSTS   COMSTA          SET STATUS
0256 A 3178         0099        SUBW   R7,LEV2AD       IS THIS TTY INPUT
0258 A 0008
025A * 9A00         0100        BZET   LV2IN
025C A 3168         0101        SUBW   R6,LEV2AD       NO, CHECK TTY OUTPUT
025E A 0008
0260 * 9A00         0102        BZET   LV2OUT
0262 A 0020         0103        HALT   H'20            ILLEGAL INTERRUPT
                    0104 *
0264 * 9600         0105 LV2IN  BF2T   LV2RET          BRANCH IF OUTPUT FLAG IS TRUE
0266 A 701F         0106        MOVW   8(R7),R1        FETCH DATA AND CHECK
0268 A 0008
026A A 4E18         0107        CMPW   =H'8D,R1        FOR THE END OF THE LINE
026C A 008D
026E * 9100         0108        BEQT   INEND
0270 A 4E18         0109        CMPW   =H'8A,R1        DO NOT ALLOW LINE FEEDS TO
0272 A 008A
0274 * 9100         0110        BEQT   LV2RET          BE ENTERED
0276 A 281D         0111        MOVB   R1,TABL(R5+)    PLACE DATA INTO TABLE AND COUNT
0278 * 0000
027A A 4E58         0112        CMPW   =72,R5          CHECK FOR FULL LINE
027C A 0048
027E * 9100         0113        BEQT   INEND
0280 A 0380         0114 LV2RET REGM   LV2REG          SAVE LEVEL 2 REGISTERS
0282 A 018E         0115        STSM   COMSTA          SAVE COMMON STATUS
0284 A 0787         0116        MREG   LEV2SV          RESTORE THE ACTIVE REGISTERS AND
0286 A 0405         0117        RETN   LV2STA          RETURN TO THE INTERRUPTED PROGRAM
                    0118 *
0288 A 3017         0119 INEND  SETUP  7               SET TTY CONTROLLER FOR OUTPUT
028A A 4897
028C A 301F
028E A 0006
0290 A 02A0         0120        SETS   F2
0292 A 4028         0121        JSBR   OUTB,R2         WAIT FOR BUFFER READY
0294 * 0000
0296 A 4038         0122        JSBR   CRLF,R3         RETURN THE CARRIAGE
0298 * 0000
029A A 4855         0123        MOVW   R5,R5           CHECK FOR NO INPUT
029C * 9A00         0124        BZET   NOOUT
029E A 4E56         0125        CMPW   R6,R5           OR SINGE ENTRY
```

```
02A0 * 9100    0126          BEQT    SORTDN
02A2 A 4845    0127          MOVW    R5,R4           SET R4 FOR SORT-MAX
02A4 A 49C1    0128          SUBW    =1,R4           ADJUST FOR ZERO
02A6 A 4880    0129 LOOP1    MOVW    =0,R3           INITIALIZE TABLE POINTER
02A8 A 0240    0130          RSTS    F3              RESET SWAP FLAG
02AA A 681B    0131 LOOP2    MOVB    TABL(R3+),R1    CHECK TWO CONSECUTIVE BYTES
02AC * 0000
02AE A 7E1B    0132          CMPB    TABL(R3),R1
02B0 * 0000
02B2 * 9500    0133          BF1T    $1              ADJUST SORT DIRECTION ACCORDING
02B4 * 8200    0134          BGTF    NOSWAP          TO PANEL FLASH
02B6 * 9000    0135          BRUN    $2
02B8 * 8C00    0136 $1       BLTF    NOSWAP
02BA A 782B    0137 $2       MOVB    TABL(R3),R2     PERFORM THE SWAP
02BC * 0000
02BE A 381B    0138          MOVB    R1,TABL(R3)
02C0 * 0000
02C2 A 182B    0139          MOVB    R2,TABL(-R3)
02C4 * 0000
02C6 A 02C0    0140          SETS    F3              INDICATE ACTION
02C8 A 4A36    0141          ADDW    R6,R3           ADJUST POINTER
02CA A 4E43    0142 NOSWAP   CMPW    R3,R4           CHECK FOR TABLE END
02CC A 81EF    0143          BEQF    LOOP2
02CE * 8700    0144          BF3F    SORTDN          IF NO CHANGE THEN ITS DONE
02D0 A 4946    0145          SUBW    R6,R4           ADJUST TABLE END AND CHECK
02D2 A 8AEA    0146          BZEF    LOOP1           FOR COMPLETION
02D4 A 48B0    0147 SORTDN   MOVW    =0,R3           START OUTPUT OF THE SORTED
02D6 A 681B    0148 $1       MOVB    TABL(R3+),R1    TABLE
02D8 * 0000
02DA A 4028    0149          JSBR    OUTPUT,R2
02DC * 0000
02DE A 4E53    0150          CMPW    R3,R5
02E0 A 81FB    0151          BEQF    $1              LOOP UNTIL IT IS ALL OUT
02E2 A 48D0    0152          MOVW    =0,R5           ZERO THE BYTE COUNT
02E4 A 4038    0153          JSBR    CRLF,R3         RETURN CARRIAGE
02E6 * 0000
02E8 A 3017    0154 NOOUT    SETUP   HID             SET TTY CONTROLLER FOR INPUT
02EA A 489D
02EC A 301F
02EE A 0006
02F0 A 0220    0155          RSTS    F2              RESET OUTPUT FLAG
02F2 A 90C7    0156          BRUN    LV2RET          RETURN
               0157 *
02F4 A 86C6    0158 LV2OUT   BF2F    LV2RET          BRANCH IF OUTPUT FLAG NOT ON
02F6 A 4008    0159          JUMP    LV2OT           THIS JUMP TECHNIQUE IS A USEFUL
02F8 * 0000
     A 02F8    0160 LV2OT    DEFN    *-2             METHOD FOR PROGRAM SWITCHES
```

```
                    0162 *
02FA A 301F         0163 OUTPUT MOVW   R1,8(R7)         OUTPUT THE DATA
02FC A 0008
02FE A 3028         0164 OUTB   MOVW   R2,LV2OT         SAVE RETURN IN THE SWITCH
0300 A 02F8
0302 A 90BF         0165        BRUN   LV2RET           RETURN
                    0166 *
0304 A 4818         0167 CRLF   MOVW   =H'8D',R1         PERFORM CARRIAGE RETURN AND
0306 A 008D
0308 A 4028         0168        JSBR   OUTPUT,R2
030A A 02FA
030C A 4818         0169        MOVW   =H'8A',R1         LINE FEED
030E A 008A
0310 A 4028         0170        JSBR   OUTPUT,R2
0312 A 02FA
0314 A 4003         0171        JUMP   (R3)              RETURN THROUGH REGISTER 3
                    0172 *
                    0173 *      LEVEL 1  - PANEL INTERRUPT
                    0174 *
0316 A 4008         0175 LEVEL1 JUMP   NEW               PERFORM PARTIAL RE-INITIALIZE
0318 * 0000
                    0176 *
                    0177 *
                    0178 *      THE BASE PROGRAM ILLUSTRATES THE USEAGE OF A MONITOR
                    0179 *  TECHNIQUE WHICH CAN UTILIZE A SOFTWARE EXECUTIVE TO SCHEDULE
                    0180 *  AND SUPERVISE THE EXECUTION OF TASKS IN A MULTIPLE PROGRAMING
                    0181 *  SYSTEM.  THIS EXAMPLE MERELY DISPLAYS A COUNTER ON THE PANEL
                    0182 *  DATA LIGHTS. THE BASE PROGRAM IS INTERRUPTED BY THE SYSTEM
                    0183 *  WHEN DATA TRANSFER OR POWER SUPPLY INPUTS NEED SERVICE.
                    0184 *
031A A 020C         0185 BASE   RSTS   CARRY+OVRFLW      RESET CARRY AND OVERFLOW INDICATORS
031C A 4AD2         0186        ADDW   =2,R5             UPDATE COUNTER AND WAIT
031E A 83FE         0187        BOVF   BASE              FOR OVERFLOW (1/8 OF SECOND)
0320 A 48D0         0188        MOVW   =0,R5             RESET COUNTER
0322 A 4AC1         0189        ADDW   =1,R4             UPDATE DATA DISPLAY
0324 A 3048         0190        MOVW   R4,DATA
0326 A FF82
0328 A 90F9         0191        BRUN   BASE              LOOP BACK
```

```
                        0193 *      THE INITIALIZATION SECTION IS THE FIRST PORTION OF THE
                        0194 *  PROGRAM WHICH IS EXECUTED AFTER IT IS LOADED.  IF A POWER
                        0195 *  FAILURE OCCURS,  INITIALIZATION IS RE-ENTERED AFTER POWER
                        0196 *  IS RESTORED TO RESTART THE PROGRAM.
                        0197 *
032A A 4890             0198 INITL  MOVW   =0,R1           AFTER POWER IS RESTORED, TIME
032C A 3018             0199        MOVW   R1,ADDRS
032E A FF80
0330 A 3018             0200        MOVW   R1,DATA         MUST BE ALLOTTED FOR THE TTY TO
0332 A FF82
0334 A 4891             0201        MOVW   =1,R1
0336 A 48A6             0202        MOVW   =6,R2           REACH A READY CONDITION. THIS
0338 A 0208             0203 LOOPW  RSTS   CARRY
033A A 3218             0204        ADDW   R1,DATA         LOOP COUNTS APPROXIMATELY 3
033C A FF82
033E A 84FD             0205        BCYF   LOOPW
0340 A 3218             0206        ADDW   R1,ADDRS        SECONDS AND DISPLAYS THE COUNT
0342 A FF80
0344 A 3628             0207        CMPW   R2,ADDRS        ON THE PANEL
0346 A FF80
0348 A 81F8             0208        BCSF   LOOPW
                        0209 *
034A A 4818             0210        MOVW   =LEVEL6,R1      THIS SECTION INITIALIZES ALL OF
034C A 0200
034E A 3018             0211        MOVW   R1,LV6VEC
0350 A 002E
0352 A 4818             0212        MOVW   =LEVEL5,R1      THE INTERRUPT VECTORS
0354 A 0202
0356 A 3018             0213        MOVW   R1,LV5VEC
0358 A 0026
035A A 4818             0214        MOVW   =LEVEL4,R1
035C A 0204
035E A 3018             0215        MOVW   R1,LV4VEC
0360 A 001E
0362 A 4818             0216        MOVW   =LEVEL3,R1
0364 A 024E
0366 A 3018             0217        MOVW   R1,LV3VEC
0368 A 0016
036A A 4818             0218        MOVW   =LEVEL2,R1
036C A 0250
036E A 3018             0219        MOVW   R1,LV2VEC
0370 A 000E
0372 A 4818             0220        MOVW   =LEVEL1,R1
0374 A 0316
0376 A 3018             0221        MOVW   R1,LV1VEC
0378 A 0006
```

```
                    0223 *
037A A 4818         0224          MOVW    =H)F820,R1      SET COMMON STATUS STORAGE
037C A F820
037E A 3018         0225          MOVW    R1,COMSTA       TO LEFT SHIFT AND TTY OUTPUT
0380 A 011C
0382 A 4818         0226          MOVW    =CRLF,R1        SET LEVEL 2 TO OUTPUT OF
0384 A 0304
0386 A 3018         0227          MOVW    R1,LV2OT        CARRIAGE RETURN INTERRUPT
0388 A 02F8
038A * 0700         0228          MREG    LV2STR          INITIALIZE REGISTER STORAGE
038C A 0380         0229          REGM    LV2REG          FOR LEVEL 2
038E A 3017         0230          SETUP   H)7             SET TTY CONTROL TO OUTPUT AND
0390 A 4897
0392 A 301F
0394 A 0006
0396 * 0700         0231 NEW      MREG    INIREG          INITIALIZE BASE REGISTERS
0398 A 3068         0232          MOVW    R6,FLASH        SET PANEL FLASH
039A * 0000
039C A 3068         0233          MOVW    R6,ADDRS
039E A FF80
03A0 A 4818         0234          MOVW    =H)FFEF,R1      SET LEFT SHIFT FOR FLASH
03A2 A FFEF
03A4 A 3318         0235          ANDW    R1,COMSTA
03A6 A 011C
03A8 * 0400         0236          RETN    BASE1           GO TO BASE PROGRAM
```

```
                    0238 *
                    0239 *       TEMPORARY STORAGE AND DATA CONSTANTS
                    0240 *
 03AA A 0002        0241 FLASH   SAVE   2                DISPLAY DATA FOR ADDRESS REGISTER
 03AC A 0002        0242 SAVR1   SAVE   2                R1 STORAGE FOR LEVEL 4
                    0243 *
                    0244 *       INITIAL REGISTER CONDITIONS FOR BASE PROGRAM
                    0245 *  AND LEVEL 2
                    0246 *
 03AE A 0000        0247 LV2SIR  DATA   0                R1 - TEMPORARY
 03B0 A 0000        0248         DATA   0                R2 - TEMP BYTE SWAP STORAGE
 03B2 A 02E8        0249         ADDR   NOOUT            R3 - TEMP SORT AND SUBROUTINE STORAGE
 03B4 A 0000        0250         DATA   0                R4 - SORT MAXIMUM
 03B6 A 0000        0251         DATA   0                R5 - INPUT BYTE COUNT
 03B8 A 0001        0252         DATA   1                R6 - CONSTANT = 1
 03BA A F800        0253         DATA   TTY              R7 - TTY ADDRESS
 03BC A 0000        0254 INIREG  DATA   0,0,0,0,0,1,30    R1,R2,R3,R4,R5,R6,R7
 03BE A 0000
 03C0 A 0000
 03C2 A 0000
 03C4 A 0000
 03C6 A 0001
 03C8 A 001E
      A 03C8        0255 FCOUNT  DEFN   *-2              RTC PULSE COUNTER
                    0256 *       BASE PROGRAM INITIAL STATUS AND PROGRAM COUNTER
                    0257 *
 03CA A 082C        0258 BASE1   DATA   H)082C,BASE
 03CC A 031A
                    0259 *
 03CE A 0048        0260 TABL    SAVE   72               TTY STORAGE TABLE
      A 032A        0261         END    INITL            AUTOMATICALLY START AT INITIALIZATION
0000 ERRORS
```

PC	X	0000	R0	X	0000	R1	X	0001	R2	X	0002
R3	X	0003	R4	X	0004	R5	X	0005	R6	X	0006
R7	X	0007	SETUP	M	0000	F1	A	0010	F2	A	0020
F3	A	0040	CARRY	A	0008	OVRFLW	A	0004	LEV4AD	A	0018
LV4STA	A	001A	LEV2AD	A	0008	LV2STA	A	000A	TTY	A	F800
LV6VEC	A	002E	LV5VEC	A	0026	LV4VEC	A	001E	LV3VEC	A	0016
LV2VEC	A	000E	LV1VEC	A	0006	ADDRS	A	FF80	DATA	A	FF82
LV2REG	A	0100	LEV2SV	A	010E	COMSTA	A	011C	LEVEL6	A	0200
LEVEL5	A	0202	LEVEL4	A	0204	SAVR1	A	03AC	INITL	A	032A
LEV4OT	A	0248	FCOUNT	A	0368	FLASH	A	03AA	LEVEL3	A	024E
LEVEL2	A	0250	LV2IN	A	0264	LV2OUT	A	02F4	LV2RET	A	0280
INEND	A	0288	TABL	A	036E	OUTE	A	02FE	CRLF	A	0304
NOOUT	A	02E8	SORTDN	A	0204	LOOP1	A	02A6	LOOP2	A	02AA
NOSWAP	A	02CA	OUTPUT	A	02FA	LV2OT	A	02F8	LEVEL1	A	0316
NEW	A	0396	BASE	A	031A	LOOPW	A	0338	LV2STR	A	03AE
INIREG	A	03BC	BASE1	A	03CA						

APPENDIX D

PAPER TAPE FORMATS

GENERAL

Paper tape is still the most used media for loading minicomputers. Procedures for loading SUE from other devices such as discs, cards or magnetic tapes have been developed but will not be discussed in this book. Programs punched in binary format on paper tape may be placed in the reader and loaded by the Basic Loader (BLOD-2). BLOD-2 is a sixty-four word program which is incorporated in the autoload option and is also available on paper tape. The autoload version executes out of a ROM starting at location $FB00_{16}$. The paper tape version may be read into core memory by positioning it in the reader such that the first character read is the first character of BLOD-2 and executing a small hand entered bootstrap program.

SOURCE (ASCII) FORMAT

The United States American National Standard Code for Information Interchange (ASCII) uses all eight channels of paper tape to represent a single character (letter, digit or symbol). This code has been modified to have the most significant bit always on for SUE (see Appendix E). LAP-2 and the Edit programs process an 80-character ASCII card image. This image can be produced by each of the input drivers for the SUE Input/Output Control System (IOCS). The format used by the paper tape driver enables it to produce the card image from each paper tape record regardless of the size of the record. After the leader at the head of the tape, the first and each succeeding record begins with a line feed (8A) and ends with a carriage return (8D). Each record will produce one 80-column card image (blanks are filled in). A program (source file) is terminated by the last record being either an "END" or END F" directive. Immediately after the carriage return (8D) of the last record, the end of file character (CNTRL, SHIFT, L or 9C) must be punched. This enables the operating system to detect the end of tape and return control to the operator just as it detects the last card of a card deck by reading the '/*' end of file card.

OBJECT FORMAT (figure D-3)

Paper tapes (cards, etc.) generated by the assemblers and compilers will be in object format and can be processed by the Link Loader. Object tapes consist of a series of one-byte type codes (table D-1) intermixed with the data which each type code describes. Type code one (1), for instance, announces

that the two bytes following it are to be stored (unmodified) into the location currently pointed to by the Location Counter and that the Location Counter is to be incremented by 2.

Figures D-2, D-3 and D-4 illustrate an assembler listing of a program, the object tape produced by assembling the program and the binary tape produced by link loading the program.

Table D-1. Type Codes recognized by the Link Loader

Type Code No.	Definition
01	Absolute word
02	Relocatable word
03	Undefined local symbol ref. or undefined displacement address field
04	External reference
05	Absolute data byte
06	Defines an "undefined" reference
07	Memory Block
08	Relocatable core location counter
09	Absolute core location counter
10	Entry point name
11	Forward reference name
12	External name
13	Checksum
14	End of program unit
15	End of file
16	Blank common size
17	Blank common reference word
18	Labeled common name and size
19	Labeled common ref. word
20	Overlay data
21	Labeled common data initializing constant

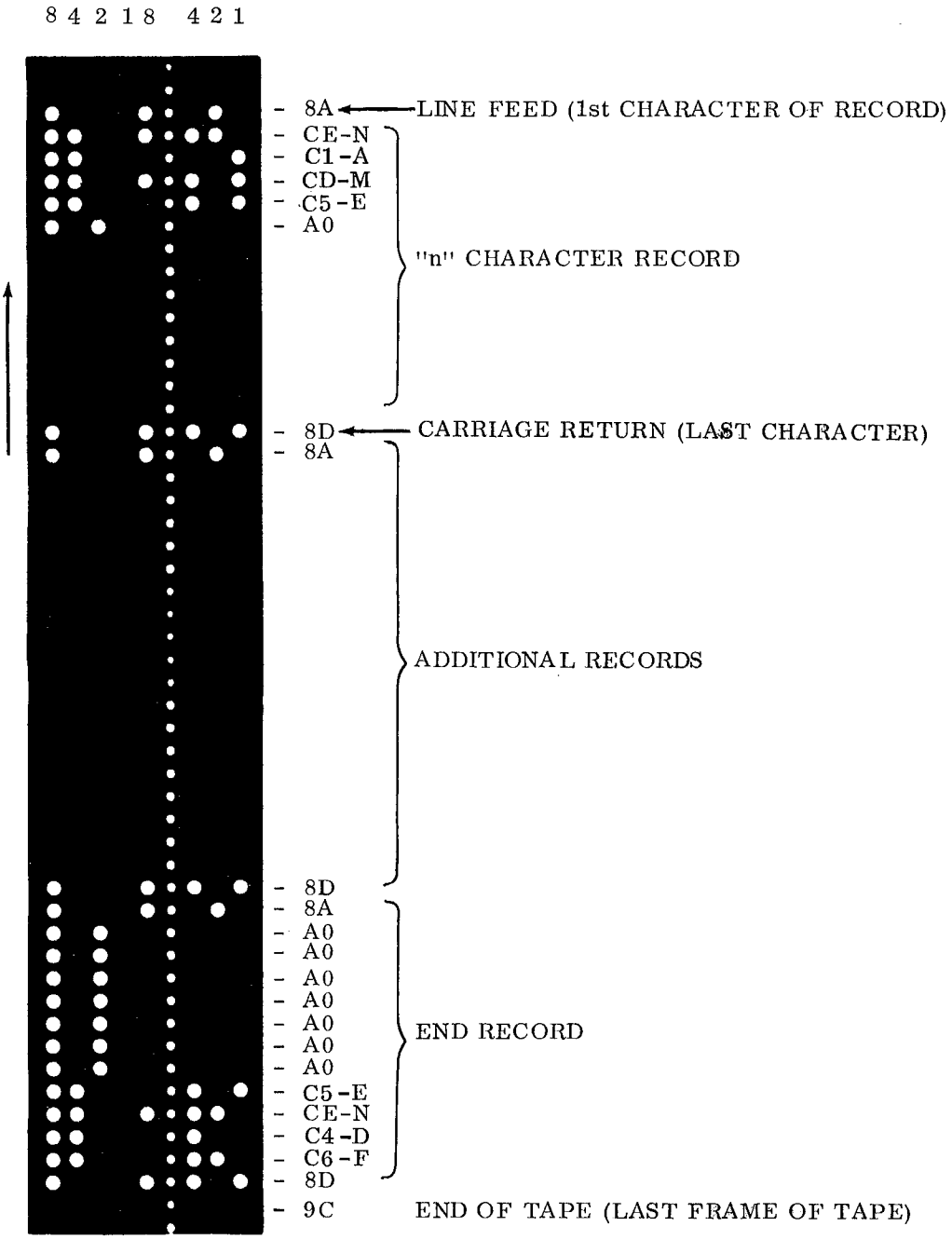


Figure D-1. Source Paper Tape Format for SUE Assembler

```
                    0001 *       SAMPLE PROGRAM ILLUSTRATING SEVERAL
                    0002 *  DIFFERENT TYPE CODES.
                    0003 *
                    0004         XIRV    ABC              TYPE 6
                    0005         NIRY    FCD              TYPE 10
0000 A 007B         0006         DATA    123              TYPE 1
0002 R 0002         0007         DATA    *                TYPE 2
0004 E 0000         0008         DATA    ABC              TYPE 6&4
0006 B   01         0009         BITE    1,2              TYPE 5
0007 B   02
     A 0100         0010         CORE    256              TYPE 8
0100 A 0000         0011 ZZXC    DATA    0                TYPE 1
                    0012 *       TYPE 13 (CHECKSUM)
0102 A 0001         0013 FCD     DATA    1                TYPE 11,12,&1
     R 0102         0014         END     FCD              TYPE 14&13
0000 ERRORS
```

Figure D-2. Example Program of Different Type Codes

BINARY FORMAT (figure D-4)

Programs produced by the Link Loader are in binary format and may be loaded into core by the Basic Loader. The format consists of a series of records which contain a count (byte), location (word), data (number of bytes indicated by the count) and a checksum. The checksum is the 16-bit sum of the bytes in the record (excluding the checksum itself). The last record of the tape will have a count of 127 (FF_1) and will contain a transfer address and a checksum.

BINARY OBJECT FORMAT

Paper tapes generated by the Link Loader are in the binary object format; these tapes are used as input to the Basic Loader. The core load generated by linking process is output as a collection of records, each record is a formatted string of bytes. There are two formats for the records, one for the data record and one for the last record. Both are shown in figure D-4.

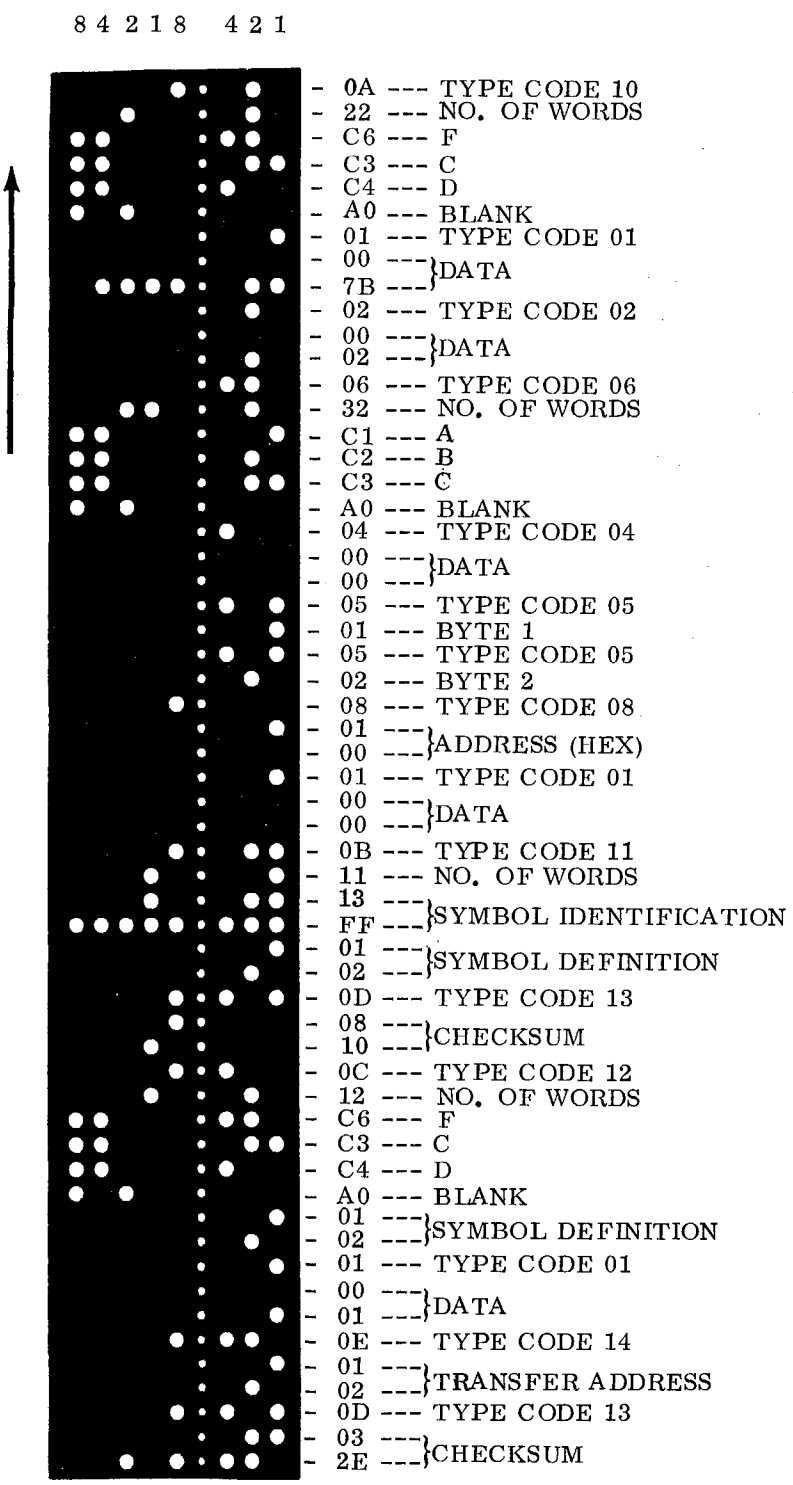


Figure D-3. Link Loader Tape Format

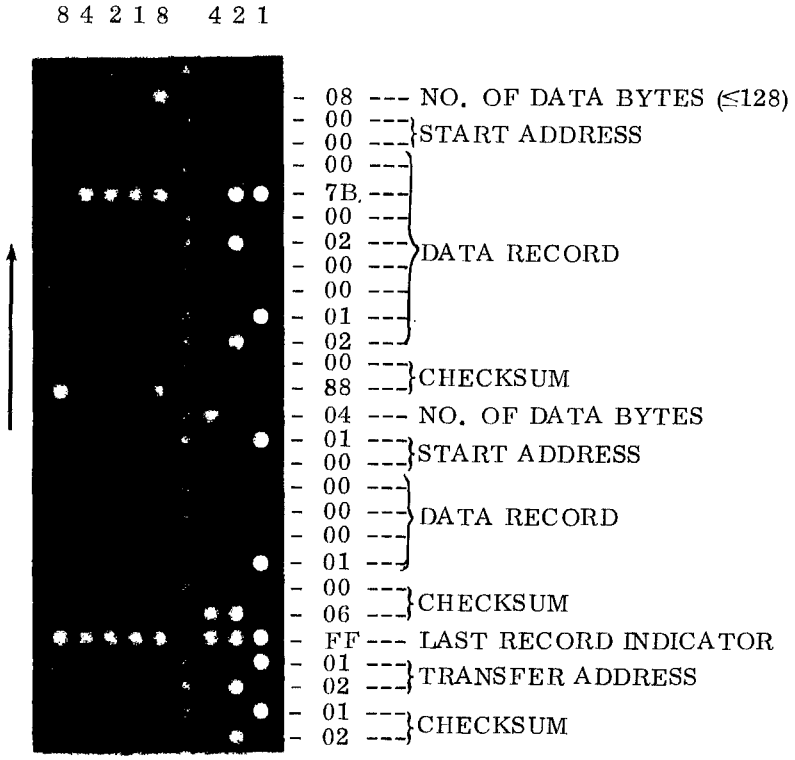


Figure D-4. Binary Tape Format

APPENDIX E

ANSI CHARACTER SET AND HEXADECIMAL CODES

HEX	CHARACTER
A0	space
A1	!
A2	"
A3	#
A4	$
A5	%
A6	&
A7	' (apostrophe)
A8	(
A9	)
AA	*
AB	+
AC	, (comma)
AD	-
AE	. (period)
AF	/
B0	0
B1	1
B2	2
B3	3
B4	4
B5	5
B6	6
B7	7
B8	8
B9	9
BA	:
BB	;
BC	< less than
BD	=
BE	> greater than
BF	?
C0	@

HEX	CHARACTER
C1	A
C2	B
C3	C
C4	D
C5	E
C6	F
C7	G
C8	H
C9	I
CA	J
CB	K
CC	L
CD	M
CE	N
CF	O
D0	P
D1	Q
D2	R
D3	S
D4	T
D5	U
D6	V
D7	W
D8	X
D9	Y
DA	Z
DB	[left bracket
DC	\ back slash
DD	] right bracket
DE	↑ up arrow
DF	← left arrow
87	bell
8A	line feed
8D	carriage return

APPENDIX F

HEXADECIMAL TO DECIMAL CONVERSION

DIRECT CONVERSION TABLE

This table provides direct conversion of decimal and hexadecimal number in these ranges:

HEXADEIMAL	DECIMAL
000 to FFF	0000 to 4095

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

HEXADECIMAL	DECIMAL
1000	4096
2000	8192
3000	12288
4000	16384
5000	20480
6000	24576
7000	28672
8000	32768
9000	36864
A000	40960
B000	45056
C000	49152
D000	53248
E000	57344

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00_	0000	0001	0002	0003	0004	0005	0006	0007	0008	0009	0010	0011	0012	0013	0014	0015
01_	0016	0017	0018	0019	0020	0021	0022	0023	0024	0025	0026	0027	0028	0029	0030	0031
02_	0032	0033	0034	0035	0036	0037	0038	0039	0040	0041	0042	0043	0044	0045	0046	0047
03_	0048	0049	0050	0051	0052	0053	0054	0055	0056	0057	0058	0059	0060	0061	0062	0063
04_	0064	0065	0066	0067	0068	0069	0070	0071	0072	0073	0074	0075	0076	0077	0078	0079
05_	0080	0081	0082	0083	0084	0085	0086	0087	0088	0089	0090	0091	0092	0093	0094	0095
06_	0096	0097	0098	0099	0100	0101	0102	0103	0104	0105	0106	0107	0108	0109	0110	0111
07_	0112	0113	0114	0115	0116	0117	0118	0119	0120	0121	0122	0123	0124	0125	0126	0127
08_	0128	0129	0130	0131	0132	0133	0134	0135	0136	0137	0138	0139	0140	0141	0142	0143
09_	0144	0145	0146	0147	0148	0149	0150	0151	0152	0153	0154	0155	0156	0157	0158	0159
0A_	0160	0161	0162	0163	0164	0165	0166	0167	0168	0169	0170	0171	0172	0173	0174	0175
0B_	0176	0177	0178	0179	0180	0181	0182	0183	0184	0185	0186	0187	0188	0189	0190	0191
0C_	0192	0193	0194	0195	0196	0197	0198	0199	0200	0201	0202	0203	0204	0205	0206	0207
0D_	0208	0209	0210	0211	0212	0213	0214	0215	0216	0217	0218	0219	0220	0221	0222	0223
0E_	0224	0225	0226	0227	0228	0229	0230	0231	0232	0233	0234	0235	0236	0237	0238	0239
0F_	0240	0241	0242	0243	0244	0245	0246	0247	0248	0249	0250	0251	0252	0253	0254	0255
10_	0256	0257	0258	0259	0260	0261	0262	0263	0264	0265	0266	0267	0268	0269	0270	0271
11_	0272	0273	0274	0275	0276	0277	0278	0279	0280	0281	0282	0283	0284	0285	0286	0287
12_	0288	0289	0290	0291	0292	0293	0294	0295	0296	0297	0298	0299	0300	0301	0302	0303
13_	0304	0305	0306	0307	0308	0309	0310	0311	0312	0313	0314	0315	0316	0317	0318	0319
14_	0320	0321	0322	0323	0324	0325	0326	0327	0328	0329	0330	0331	0332	0333	0334	0335
15_	0336	0337	0338	0339	0340	0341	0342	0343	0344	0345	0346	0347	0348	0349	0350	0351
16_	0352	0353	0354	0355	0356	0357	0358	0359	0360	0361	0362	0363	0364	0365	0366	0367
17_	0368	0369	0370	0371	0372	0373	0374	0375	0376	0377	0378	0379	0380	0381	0382	0383
18_	0384	0385	0386	0387	0388	0389	0390	0391	0392	0393	0394	0395	0396	0397	0398	0399
19_	0400	0401	0402	0403	0404	0405	0406	0407	0408	0409	0410	0411	0412	0413	0414	0415
1A_	0416	0417	0418	0419	0420	0421	0422	0423	0424	0425	0426	0427	0428	0429	0430	0431
1B_	0432	0433	0434	0435	0436	0437	0438	0439	0440	0441	0442	0443	0444	0445	0446	0447
1C_	0448	0449	0450	0451	0452	0453	0454	0455	0456	0457	0458	0459	0460	0461	0462	0463
1D_	0464	0465	0466	0467	0468	0469	0470	0471	0472	0473	0474	0475	0476	0477	0478	0479
1E_	0480	0481	0482	0483	0484	0485	0486	0487	0488	0489	0490	0491	0492	0493	0494	0495
1F_	0496	0497	0498	0499	0500	0501	0502	0503	0504	0505	0506	0507	0508	0509	0510	0511

HEXADECIMAL TO DECIMAL CONVERSION

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20_	0512	0513	0514	0515	0516	0517	0518	0519	0520	0521	0522	0523	0524	0525	0526	0527
21_	0528	0529	0530	0531	0532	0533	0534	0535	0536	0537	0538	0539	0540	0541	0542	0543
22_	0544	0545	0546	0547	0548	0549	0550	0551	0552	0553	0554	0555	0556	0557	0558	0559
23_	0560	0561	0562	0563	0564	0565	0566	0567	0568	0569	0570	0571	0572	0573	0574	0575
24_	0576	0577	0578	0579	0580	0581	0582	0583	0584	0585	0586	0587	0588	0589	0590	0591
25_	0592	0593	0594	0595	0596	0597	0598	0599	0600	0601	0602	0603	0604	0605	0606	0607
26_	0608	0609	0610	0611	0612	0613	0614	0615	0616	0617	0618	0619	0620	0621	0622	0623
27_	0624	0625	0626	0627	0628	0629	0630	0631	0632	0633	0634	0635	0636	0637	0638	0639
28_	0640	0641	0642	0643	0644	0645	0646	0647	0648	0649	0650	0651	0652	0653	0654	0655
29_	0656	0657	0658	0659	0660	0661	0662	0663	0664	0665	0666	0667	0668	0669	0670	0671
2A_	0672	0673	0674	0675	0676	0677	0678	0679	0680	0681	0682	0683	0684	0685	0686	0687
2B_	0688	0689	0690	0691	0692	0693	0694	0695	0696	0697	0698	0699	0700	0701	0702	0703
2C_	0704	0705	0706	0707	0708	0709	0710	0711	0712	0713	0714	0715	0716	0717	0718	0719
2D_	0720	0721	0722	0723	0724	0725	0726	0727	0728	0729	0730	0731	0732	0733	0734	0735
2E_	0736	0737	0738	0739	0740	0741	0742	0743	0744	0745	0746	0747	0748	0749	0750	0751
2F_	0752	0753	0754	0755	0756	0757	0758	0759	0760	0761	0762	0763	0764	0765	0766	0767
30_	0768	0769	0770	0771	0772	0773	0774	0775	0776	0777	0778	0779	0780	0781	0782	0783
31_	0784	0785	0786	0787	0788	0789	0790	0791	0792	0793	0794	0795	0796	0797	0798	0799
32_	0800	0801	0802	0803	0804	0805	0806	0807	0808	0809	0810	0811	0812	0813	0814	0815
33_	0816	0817	0818	0819	0820	0821	0822	0823	0824	0825	0826	0827	0828	0829	0830	0831
34_	0832	0833	0834	0835	0836	0837	0838	0839	0840	0841	0842	0843	0844	0845	0846	0847
35_	0848	0849	0850	0851	0852	0853	0854	0855	0856	0857	0858	0859	0860	0861	0862	0863
36_	0864	0865	0866	0867	0868	0869	0870	0871	0872	0873	0874	0875	0876	0877	0878	0879
37_	0880	0881	0882	0883	0884	0885	0886	0887	0888	0889	0890	0891	0892	0893	0894	0895
38_	0896	0897	0898	0899	0900	0901	0902	0903	0904	0905	0906	0907	0908	0909	0910	0911
39_	0912	0913	0914	0915	0916	0917	0918	0919	0920	0921	0922	0923	0924	0925	0926	0927
3A_	0928	0929	0930	0931	0932	0933	0934	0935	0936	0937	0938	0939	0940	0941	0942	0943
3B_	0944	0945	0946	0947	0948	0949	0950	0951	0952	0953	0954	0955	0956	0957	0958	0959
3C_	0960	0961	0962	0963	0964	0965	0966	0967	0968	0969	0970	0971	0972	0973	0974	0975
3D_	0976	0977	0978	0979	0980	0981	0982	0983	0984	0985	0986	0987	0988	0989	0990	0991
3E_	0992	0993	0994	0995	0996	0997	0998	0999	1000	1001	1002	1003	1004	1005	1006	1007
3F_	1008	1009	1010	1011	1012	1013	1014	1015	1016	1017	1018	1019	1020	1021	1022	1023

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
40_	1024	1025	1026	1027	1028	1029	1030	1031	1032	1033	1034	1035	1036	1037	1038	1039
41_	1040	1041	1042	1043	1044	1045	1046	1047	1048	1049	1050	1051	1052	1053	1054	1055
42_	1056	1057	1058	1059	1060	1061	1062	1063	1064	1065	1066	1067	1068	1069	1070	1071
43_	1072	1073	1074	1075	1076	1077	1078	1079	1080	1081	1082	1083	1084	1085	1086	1087
44_	1088	1089	1090	1091	1092	1093	1094	1095	1096	1097	1098	1099	1100	1101	1102	1103
45_	1104	1105	1106	1107	1108	1109	1110	1111	1112	1113	1114	1115	1116	1117	1118	1119
46_	1120	1121	1122	1123	1124	1125	1126	1127	1128	1129	1130	1131	1132	1133	1134	1135
47_	1136	1137	1138	1139	1140	1141	1142	1143	1144	1145	1146	1147	1148	1149	1150	1151
48_	1152	1153	1154	1155	1156	1157	1158	1159	1160	1161	1162	1163	1164	1165	1166	1167
49_	1168	1169	1170	1171	1172	1173	1174	1175	1176	1177	1178	1179	1180	1181	1182	1183
4A_	1184	1185	1186	1187	1188	1189	1190	1191	1192	1193	1194	1195	1196	1197	1198	1199
4B_	1200	1201	1202	1203	1204	1205	1206	1207	1208	1209	1210	1211	1212	1213	1214	1215
4C_	1216	1217	1218	1219	1220	1221	1222	1223	1224	1225	1226	1227	1228	1229	1230	1231
4D_	1232	1233	1234	1235	1236	1237	1238	1239	1240	1241	1242	1243	1244	1245	1246	1247
4E_	1248	1249	1250	1251	1252	1253	1254	1255	1256	1257	1258	1259	1260	1261	1262	1263
4F_	1264	1265	1266	1267	1268	1269	1270	1271	1272	1273	1274	1275	1276	1277	1278	1279
50_	1280	1281	1282	1283	1284	1285	1286	1287	1288	1289	1290	1291	1292	1293	1294	1295
51_	1296	1297	1298	1299	1300	1301	1302	1303	1304	1305	1306	1307	1308	1309	1310	1311
52_	1312	1313	1314	1315	1316	1317	1318	1319	1320	1321	1322	1323	1324	1325	1326	1327
53_	1328	1329	1330	1331	1332	1333	1334	1335	1336	1337	1338	1339	1340	1341	1342	1343
54_	1344	1345	1346	1347	1348	1349	1350	1351	1352	1353	1354	1355	1356	1357	1358	1359
55_	1360	1361	1362	1363	1364	1365	1366	1367	1368	1369	1370	1371	1372	1373	1374	1375
56_	1376	1377	1378	1379	1380	1381	1382	1383	1384	1385	1386	1387	1388	1389	1390	1391
57_	1392	1393	1394	1395	1396	1397	1398	1399	1400	1401	1402	1403	1404	1405	1406	1407
58_	1408	1409	1410	1411	1412	1413	1414	1415	1416	1417	1418	1419	1420	1421	1422	1423
59_	1424	1425	1426	1427	1428	1429	1430	1431	1432	1433	1434	1435	1436	1437	1438	1439
5A_	1440	1441	1442	1443	1444	1445	1446	1447	1448	1449	1450	1451	1452	1453	1454	1455
5B_	1456	1457	1458	1459	1460	1461	1462	1463	1464	1465	1466	1467	1468	1469	1470	1471
5C_	1472	1473	1474	1475	1476	1477	1478	1479	1480	1481	1482	1483	1484	1485	1486	1487
5D_	1488	1489	1490	1491	1492	1493	1494	1495	1496	1497	1498	1499	1500	1501	1502	1503
5E_	1504	1505	1506	1507	1508	1509	1510	1511	1512	1513	1514	1515	1516	1517	1518	1519
5F_	1520	1521	1522	1523	1524	1525	1526	1527	1528	1529	1530	1531	1532	1533	1534	1535

HEXADECIMAL TO DECIMAL CONVERSION

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
60_	1536	1537	1538	1539	1540	1541	1542	1543	1544	1545	1546	1547	1548	1549	1550	1551
61_	1552	1553	1554	1555	1556	1557	1558	1559	1560	1561	1562	1563	1564	1565	1566	1567
62_	1568	1569	1570	1571	1572	1573	1574	1575	1576	1577	1578	1579	1580	1581	1582	1583
63_	1584	1585	1586	1587	1588	1589	1590	1591	1592	1593	1594	1595	1596	1597	1598	1599
64_	1600	1601	1602	1603	1604	1605	1606	1607	1608	1609	1610	1611	1612	1613	1614	1615
65_	1616	1617	1618	1619	1620	1621	1622	1623	1624	1625	1626	1627	1628	1629	1630	1631
66_	1632	1633	1634	1635	1636	1637	1638	1639	1640	1641	1642	1643	1644	1645	1646	1647
67_	1648	1649	1650	1651	1652	1653	1654	1655	1656	1657	1658	1659	1660	1661	1662	1663
68_	1664	1665	1666	1667	1668	1669	1670	1671	1672	1673	1674	1675	1676	1677	1678	1679
69_	1680	1681	1682	1683	1684	1685	1686	1687	1688	1689	1690	1691	1692	1693	1694	1695
6A_	1696	1697	1698	1699	1700	1701	1702	1703	1704	1705	1706	1707	1708	1709	1710	1711
6B_	1712	1713	1714	1715	1716	1717	1718	1719	1720	1721	1722	1723	1724	1725	1726	1727
6C_	1728	1729	1730	1731	1732	1733	1734	1735	1736	1737	1738	1739	1740	1741	1742	1743
6D_	1744	1745	1746	1747	1748	1749	1750	1751	1752	1753	1754	1755	1756	1757	1758	1759
6E_	1760	1761	1762	1763	1764	1765	1766	1767	1768	1769	1770	1771	1772	1773	1774	1775
6F_	1776	1777	1778	1779	1780	1781	1782	1783	1784	1785	1786	1787	1788	1789	1790	1791
70_	1792	1793	1794	1795	1796	1797	1798	1799	1800	1801	1802	1803	1804	1805	1806	1807
71_	1808	1809	1810	1811	1812	1813	1814	1815	1816	1817	1818	1819	1820	1821	1822	1823
72_	1824	1825	1826	1827	1828	1829	1830	1831	1832	1833	1834	1835	1836	1837	1838	1839
73_	1840	1841	1842	1843	1844	1845	1846	1847	1848	1849	1850	1851	1852	1853	1854	1855
74_	1856	1857	1858	1859	1860	1861	1862	1863	1864	1865	1866	1867	1868	1869	1870	1871
75_	1872	1873	1874	1875	1876	1877	1878	1879	1880	1881	1882	1883	1884	1885	1886	1887
76_	1888	1889	1890	1891	1892	1893	1894	1895	1896	1897	1898	1899	1900	1901	1902	1903
77_	1904	1905	1906	1907	1908	1909	1910	1911	1912	1913	1914	1915	1916	1917	1918	1919
78_	1920	1921	1922	1923	1924	1925	1926	1927	1928	1929	1930	1931	1932	1933	1934	1935
79_	1936	1937	1938	1939	1940	1941	1942	1943	1944	1945	1946	1947	1948	1949	1950	1951
7A_	1952	1953	1954	1955	1956	1957	1958	1959	1960	1961	1962	1963	1964	1965	1966	1967
7B_	1968	1969	1970	1971	1972	1973	1974	1975	1976	1977	1978	1979	1980	1981	1982	1983
7C_	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999
7D_	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015
7E_	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031
7F_	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044	2045	2046	2047

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
80_	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063
81_	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079
82_	2080	2081	2082	2083	2084	2085	2086	2087	2088	2089	2090	2091	2092	2093	2094	2095
83_	2096	2097	2098	2099	2100	2101	2102	2103	2104	2105	2106	2107	2108	2109	2110	2111
84_	2112	2113	2114	2115	2116	2117	2118	2119	2120	2121	2122	2123	2124	2125	2126	2127
85_	2128	2129	2130	2131	2132	2133	2134	2135	2136	2137	2138	2139	2140	2141	2142	2143
86_	2144	2145	2146	2147	2148	2149	2150	2151	2152	2153	2154	2155	2156	2157	2158	2159
87_	2160	2161	2162	2163	2164	2165	2166	2167	2168	2169	2170	2171	2172	2173	2174	2175
88_	2176	2177	2178	2179	2180	2181	2182	2183	2184	2185	2186	2187	2188	2189	2190	2191
89_	2192	2193	2194	2195	2196	2197	2198	2199	2200	2201	2202	2203	2204	2205	2206	2207
8A_	2208	2209	2210	2211	2212	2213	2214	2215	2216	2217	2218	2219	2220	2221	2222	2223
8B_	2224	2225	2226	2227	2228	2229	2230	2231	2232	2233	2234	2235	2236	2237	2238	2239
8C_	2240	2241	2242	2243	2244	2245	2246	2247	2248	2249	2250	2251	2252	2253	2254	2255
8D_	2256	2257	2258	2259	2260	2261	2262	2263	2264	2265	2266	2267	2268	2269	2270	2271
8E_	2272	2273	2274	2275	2276	2277	2278	2279	2280	2281	2282	2283	2284	2285	2286	2287
8F_	2288	2289	2290	2291	2292	2293	2294	2295	2296	2297	2298	2299	2300	2301	2302	2303
90_	2304	2305	2306	2307	2308	2309	2310	2311	2312	2313	2314	2315	2316	2317	2318	2319
91_	2320	2321	2322	2323	2324	2325	2326	2327	2328	2329	2330	2331	2332	2333	2334	2335
92_	2336	2337	2338	2339	2340	2341	2342	2343	2344	2345	2346	2347	2348	2349	2350	2351
93_	2352	2353	2354	2355	2356	2357	2358	2359	2360	2361	2362	2363	2364	2365	2366	2367
94_	2368	2369	2370	2371	2372	2373	2374	2375	2376	2377	2378	2379	2380	2381	2382	2383
95_	2384	2385	2386	2387	2388	2389	2390	2391	2392	2393	2394	2395	2396	2397	2398	2399
96_	2400	2401	2402	2403	2404	2405	2406	2407	2408	2409	2410	2411	2412	2413	2414	2415
97_	2416	2417	2418	2419	2420	2421	2422	2423	2424	2425	2426	2427	2428	2429	2430	2431
98_	2432	2433	2434	2435	2436	2437	2438	2439	2440	2441	2442	2443	2444	2445	2446	2447
99_	2448	2449	2450	2451	2452	2453	2454	2455	2456	2457	2458	2459	2460	2461	2462	2463
9A_	2464	2465	2466	2467	2468	2469	2470	2471	2472	2473	2474	2475	2476	2477	2478	2479
9B_	2480	2481	2482	2483	2484	2485	2486	2487	2488	2489	2490	2491	2492	2493	2494	2495
9C_	2496	2497	2498	2499	2500	2501	2502	2503	2504	2505	2506	2507	2508	2509	2510	2511
9D_	2512	2513	2514	2515	2516	2517	2518	2519	2520	2521	2522	2523	2524	2525	2526	2527
9E_	2528	2529	2530	2531	2532	2533	2534	2535	2536	2537	2538	2539	2540	2541	2542	2543
9F_	2544	2545	2546	2547	2548	2549	2550	2551	2552	2553	2554	2555	2556	2557	2558	2559

HEXADECIMAL TO DECIMAL CONVERSION

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A0_	2560	2561	2562	2563	2564	2565	2566	2567	2568	2569	2570	2571	2572	2573	2574	2575
A1_	2576	2577	2578	2579	2580	2581	2582	2583	2584	2585	2586	2587	2588	2589	2590	2591
A2_	2592	2593	2594	2595	2596	2597	2598	2599	2600	2601	2602	2603	2604	2605	2606	2607
A3_	2608	2609	2610	2611	2612	2613	2614	2615	2616	2617	2618	2619	2620	2621	2622	2623
A4_	2624	2625	2626	2627	2628	2629	2630	2631	2632	2633	2634	2635	2636	2637	2638	2639
A5_	2640	2641	2642	2643	2644	2645	2646	2647	2648	2649	2650	2651	2652	2653	2654	2655
A6_	2656	2657	2658	2659	2660	2661	2662	2663	2664	2665	2666	2667	2668	2669	2670	2671
A7_	2672	2673	2674	2675	2676	2677	2678	2679	2680	2681	2682	2683	2684	2685	2686	2687
A8_	2688	2689	2690	2691	2692	2693	2694	2695	2696	2697	2698	2699	2700	2701	2702	2703
A9_	2704	2705	2706	2707	2708	2709	2710	2711	2712	2713	2714	2715	2716	2717	2718	2719
AA_	2720	2721	2722	2723	2724	2725	2726	2727	2728	2729	2730	2731	2732	2733	2734	2735
AB_	2736	2737	2738	2739	2740	2741	2742	2743	2744	2745	2746	2747	2748	2749	2750	2751
AC_	2752	2753	2754	2755	2756	2757	2758	2759	2760	2761	2762	2763	2764	2765	2766	2767
AD_	2768	2769	2770	2771	2772	2773	2774	2775	2776	2777	2778	2779	2780	2781	2782	2783
AE_	2784	2785	2786	2787	2788	2789	2790	2791	2792	2793	2794	2795	2796	2797	2798	2799
AF_	2800	2801	2802	2803	2804	2805	2806	2807	2808	2809	2810	2811	2812	2813	2814	2815
B0_	2816	2817	2818	2819	2820	2821	2822	2823	2824	2825	2826	2827	2828	2829	2830	2831
B1_	2832	2833	2834	2835	2836	2837	2838	2839	2840	2841	2842	2843	2844	2845	2846	2847
B2_	2848	2849	2850	2851	2852	2853	2854	2855	2856	2857	2858	2859	2860	2861	2862	2863
B3_	2864	2865	2866	2867	2868	2869	2870	2871	2872	2873	2874	2875	2876	2877	2878	2879
B4_	2880	2881	2882	2883	2884	2885	2886	2887	2888	2889	2890	2891	2892	2893	2894	2895
B5_	2896	2897	2898	2899	2900	2901	2902	2903	2904	2905	2906	2907	2908	2909	2910	2911
B6_	2912	2913	2914	2915	2916	2917	2918	2919	2920	2921	2922	2923	2924	2925	2926	2927
B7_	2928	2929	2930	2931	2932	2933	2934	2935	2936	2937	2938	2939	2940	2941	2942	2943
B8_	2944	2945	2946	2947	2948	2949	2950	2951	2952	2953	2954	2955	2956	2957	2958	2959
B9_	2960	2961	2962	2963	2964	2965	2966	2967	2968	2969	2970	2971	2972	2973	2974	2975
BA_	2976	2977	2978	2979	2980	2981	2982	2983	2984	2985	2986	2987	2988	2989	2990	2991
BB_	2992	2993	2994	2995	2996	2997	2998	2999	3000	3001	3002	3003	3004	3005	3006	3007
BC_	3008	3009	3010	3011	3012	3013	3014	3015	3016	3017	3018	3019	3020	3021	3022	3023
BD_	3024	3025	3026	3027	3028	3029	3030	3031	3032	3033	3034	3035	3036	3037	3038	3039
BE_	3040	3041	3042	3043	3044	3045	3046	3047	3048	3049	3050	3051	3052	3053	3054	3055
BF_	3056	3057	3058	3059	3060	3061	3062	3063	3064	3065	3066	3067	3068	3069	3070	3071

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
C0_	3072	3073	3074	3075	3076	3077	3078	3079	3080	3081	3082	3083	3084	3085	3086	3087
C1_	3088	3089	3090	3091	3092	3093	3094	3095	3096	3097	3098	3099	3100	3101	3102	3103
C2_	3104	3105	3106	3107	3108	3109	3110	3111	3112	3113	3114	3115	3116	3117	3118	3119
C3_	3120	3121	3122	3123	3124	3125	3126	3127	3128	3129	3130	3131	3132	3133	3134	3135
C4_	3136	3137	3138	3139	3140	3141	3142	3143	3144	3145	3146	3147	3148	3149	3150	3151
C5_	3152	3153	3154	3155	3156	3157	3158	3159	3160	3161	3162	3163	3164	3165	3166	3167
C6_	3168	3169	3170	3171	3172	3173	3174	3175	3176	3177	3178	3179	3180	3181	3182	3183
C7_	3184	3185	3186	3187	3188	3189	3190	3191	3192	3193	3194	3195	3196	3197	3198	3199
C8_	3200	3201	3202	3203	3204	3205	3206	3207	3208	3209	3210	3211	3212	3213	3214	3215
C9_	3216	3217	3218	3219	3220	3221	3222	3223	3224	3225	3226	3227	3228	3229	3230	3231
CA_	3232	3233	3234	3235	3236	3237	3238	3239	3240	3241	3242	3243	3244	3245	3246	3247
CB_	3248	3249	3250	3251	3252	3253	3254	3255	3256	3257	3258	3259	3260	3261	3262	3263
CC_	3264	3265	3266	3267	3268	3269	3270	3271	3272	3273	3274	3275	3276	3277	3278	3279
CD_	3280	3281	3282	3283	3284	3285	3286	3287	3288	3289	3290	3291	3292	3293	3294	3295
CE_	3296	3297	3298	3299	3300	3301	3302	3303	3304	3305	3306	3307	3308	3309	3310	3311
CF_	3312	3313	3314	3315	3316	3317	3318	3319	3320	3321	3322	3323	3324	3325	3326	3327
D0_	3328	3329	3330	3331	3332	3333	3334	3335	3336	3337	3338	3339	3340	3341	3342	3343
D1_	3344	3345	3346	3347	3348	3349	3350	3351	3352	3353	3354	3355	3356	3357	3358	3359
D2_	3360	3361	3362	3363	3364	3365	3366	3367	3368	3369	3370	3371	3372	3373	3374	3375
D3_	3376	3377	3378	3379	3380	3381	3382	3383	3384	3385	3386	3387	3388	3389	3390	3391
D4_	3392	3393	3394	3395	3396	3397	3398	3399	3400	3401	3402	3403	3404	3405	3406	3407
D5_	3408	3409	3410	3411	3412	3413	3414	3415	3416	3417	3418	3419	3420	3421	3422	3423
D6_	3424	3425	3426	3427	3428	3429	3430	3431	3432	3433	3434	3435	3436	3437	3438	3439
D7_	3440	3441	3442	3443	3444	3445	3446	3447	3448	3449	3450	3451	3452	3453	3454	3455
D8_	3456	3457	3458	3459	3460	3461	3462	3463	3464	3465	3466	3467	3468	3469	3470	3471
D9_	3472	3473	3474	3475	3476	3477	3478	3479	3480	3481	3482	3483	3484	3485	3486	3487
DA_	3488	3489	3490	3491	3492	3493	3494	3495	3496	3497	3498	3499	3500	3501	3502	3503
DB_	3504	3505	3506	3507	3508	3509	3510	3511	3512	3513	3514	3515	3516	3517	3518	3519
DC_	3520	3521	3522	3523	3524	3525	3526	3527	3528	3529	3530	3531	3532	3533	3534	3535
DD_	3536	3537	3538	3539	3540	3541	3542	3543	3544	3545	3546	3547	3548	3549	3550	3551
DE_	3552	3553	3554	3555	3556	3557	3558	3559	3560	3561	3562	3563	3564	3565	3566	3567
DF_	3568	3569	3570	3571	3572	3573	3574	3575	3576	3577	3578	3579	3580	3581	3582	3583

HEXADECIMAL TO DECIMAL CONVERSION

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
E0_	3584	3585	3586	3587	3583	3589	3590	3591	3592	3593	3594	3595	3596	3597	3598	3599
E1_	3600	3601	3602	3603	3604	3605	3606	3607	3608	3609	3610	3611	3612	3613	3614	3615
E2_	3616	3617	3618	3619	3620	3621	3622	3623	3624	3625	3626	3627	3628	3629	3630	3631
E3_	3632	3633	3634	3635	3636	3637	3638	3639	3640	3641	3642	3643	3644	3645	3646	3647
E4_	3648	3649	3650	3651	3652	3653	3654	3655	3656	3657	3658	3659	3660	3661	3662	3663
E5_	3664	3665	3666	3667	3668	3669	3670	3671	3672	3673	3674	3675	3676	3677	3678	3679
E6_	3680	3681	3682	3683	3684	3685	3686	3687	3688	3689	3690	3691	3692	3693	3694	3695
E7_	3696	3697	3698	3699	3700	3701	3702	3703	3704	3705	3706	3707	3708	3709	3710	3711
E8_	3712	3713	3714	3715	3716	3717	3718	3719	3720	3721	3722	3723	3724	3725	3726	3727
E9_	3728	3729	3730	3731	3732	3733	3734	3735	3736	3737	3738	3739	3740	3741	3742	3743
EA_	3744	3745	3746	3747	3748	3749	3750	3751	3752	3753	3754	3755	3756	3757	3758	3759
EB_	3760	3761	3762	3763	3764	3765	3766	3767	3768	3769	3770	3771	3772	3773	3774	3775
EC_	3776	3777	3778	3779	3780	3781	3782	3783	3784	3785	3786	3787	3788	3789	3790	3791
ED_	3792	3793	3794	3795	3796	3797	3798	3799	3800	3801	3802	3803	3804	3805	3806	3807
EE_	3808	3809	3810	3811	3812	3813	3814	3815	3816	3817	3818	3819	3820	3821	3822	3823
EF_	3824	3825	3826	3827	3828	3829	3830	3831	3832	3833	3834	3835	3836	3837	3838	3839
F0_	3840	3841	3842	3843	3844	3845	3846	3847	3848	3849	3850	3851	3852	3853	3854	3855
F1_	3856	3857	3858	3859	3860	3861	3862	3863	3864	3865	3866	3867	3868	3869	3870	3871
F2_	3872	3873	3874	3875	3876	3877	3878	3879	3880	3881	3882	3883	3884	3885	3886	3887
F3_	3888	3889	3890	3891	3892	3893	3894	3895	3896	3897	3898	3899	3900	3901	3902	3903
F4_	3904	3905	3906	3907	3908	3909	3910	3911	3912	3913	3914	3915	3916	3917	3918	3919
F5_	3920	3921	3922	3923	3924	3925	3926	3927	3928	3929	3930	3931	3932	3933	3934	3935
F6_	3936	3937	3938	3939	3940	3941	3942	3943	3944	3945	3946	3947	3948	3949	3950	3951
F7_	3952	3953	3954	3955	3956	3957	3958	3959	3960	3961	3962	3963	3964	3965	3966	3967
F8_	3968	3969	3970	3971	3972	3973	3974	3975	3976	3977	3978	3979	3980	3981	3982	3983
F9_	3984	3985	3986	3987	3988	3989	3990	3991	3992	3993	3994	3995	3996	3997	3998	3999
FA_	4000	4001	4002	4003	4004	4005	4006	4007	4008	4009	4010	4011	4012	4013	4014	4015
FB_	4016	4017	4018	4019	4020	4021	4022	4023	4024	4025	4026	4027	4028	4029	4030	4031
FC_	4032	4033	4034	4035	4036	4037	4038	4039	4040	4041	4042	4043	4044	4045	4046	4047
FD_	4048	4049	4050	4051	4052	4053	4054	4055	4056	4057	4058	4059	4060	4061	4062	4063
FE_	4064	4065	4066	4067	4068	4069	4070	4071	4072	4073	4074	4075	4076	4077	4078	4079
FF_	4080	4081	4082	4083	4084	4085	4086	4087	4088	4089	4090	4091	4092	4093	4094	4095

APPENDIX G

ADDRESS ALLOCATION

GENERAL

The area from 0100 to EFFF of address space is assigned to user programs. Addresses 0000 - 00FF (256 bytes) are defined as the "Executive Space". Addresses F000 - FFFF are reserved for devices; these addresses represent registers defining I/O devices and other functional devices.

EXECUTIVE SPACE

Executive Space is used for interrupt functions; this area is addressable by the programmer.

Addresspace (Hex Bytes)	Description	
0000	Interrupting Device Number	
0002	Processor Status	LEVEL 1
0004	Current Program Counter	
0006	Device Service Routine Vector	
0008		
000A	Same as level 1	LEVEL 2
000C		
000E		
0010		
0012	Same as level 1	LEVEL 3
0014		
0016		
0018		
001A	Same as level 1	LEVEL 4
001C		
001E		
0020	Unimplemented Instruction	
0022	Processor Status	
0024	Address of the Unimplemented Instruction	
0026	Unimplemented Instruction Routine Vector	
0028	Aborted Instruction Operand Address	CPU #0
002A	Processor Status	
002C	Address of the Aborted Instruction	
002E	Abort Routine Vector	

Addresspace (Hex Bytes)	Description	
0030 0032 0034 0036 0038 003A 003C 003E	Same as CPU #0	CPU #1
0040 0042 0044 0046 004A 004C 004E	Same as CPU #0	CPU #2
0050 0052 0054 0056 0058 005A 005C 005E	Same as CPU #0	CPU #3
0060 - 00FF	Available for tables and system data.	

DEVICE ADDRESSES

These address (F000 - FFFF) are reserved for device addressing. Addresses are assigned to registers within the different I/O controllers, control panels, processors, and other devices when the system is initially configured.

Address Space (Hex Bytes)	Description
F000 - FEFF	I/O Device Registers
FF00	CPU #0 - Register 0 (Program Counter)
FF02	CPU #0 - Register 1
FF04	CPU #0 - Register 2
FF06	CPU #0 - Register 3
FF08	CPU #0 - Register 4
FF0A	CPU #0 - Register 5
FF0C	CPU #0 - Register 6
FF0E	CPU #0 - Register 7

Address Space (Hex Bytes)	Description
FF10	CPU #0 - Register 8 (Status Register/Indicators)
FF12	CPU #0 - Register 9 (Instruction Register)
FF14	CPU #0 - Register 10 (Address of last instruction executed)
FF16	CPU #0 - Register 11 (Miscellaneous contents)
FF18	CPU #0 - Reserved
FF1A	CPU #0 - Reserved
FF1C	CPU #0 - Reserved
FF1E	CPU #0 - Control FFs
FF20 - FF3E	CPU #1 - Same as CPU 0
FF40 - FF5E	CPU #2 - Same as CPU 0
FF60 - FF7E	CPU #3 - Same as CPU 0
FF80	Control Panel #1 Address Register
FF82	Control Panel #1 Data Register
FF84	Control Panel #2 Address Register
FF86	Control Panel #2 Data Register
FF88	Control Panel #3 Address Register
FF8A	Control Panel #3 Data Register
FF8C	Control Panel #4 Address Register
FF8E	Control Panel #4 Data Register
FF90 - FF9E	RESERVED
FFA0	Real Time Clock - Increment
FFA2	Real Time Clock - Limit
FFA4 - FFAE	Not Assigned
FFB0 - FFEE	RESERVED
FFF0 - FFFE	Not Assigned

MODULE ADDRESSES

Module address assignment is variable by jumper wires connected on each controller. The addresses shown are recommended for standard I/O devices.

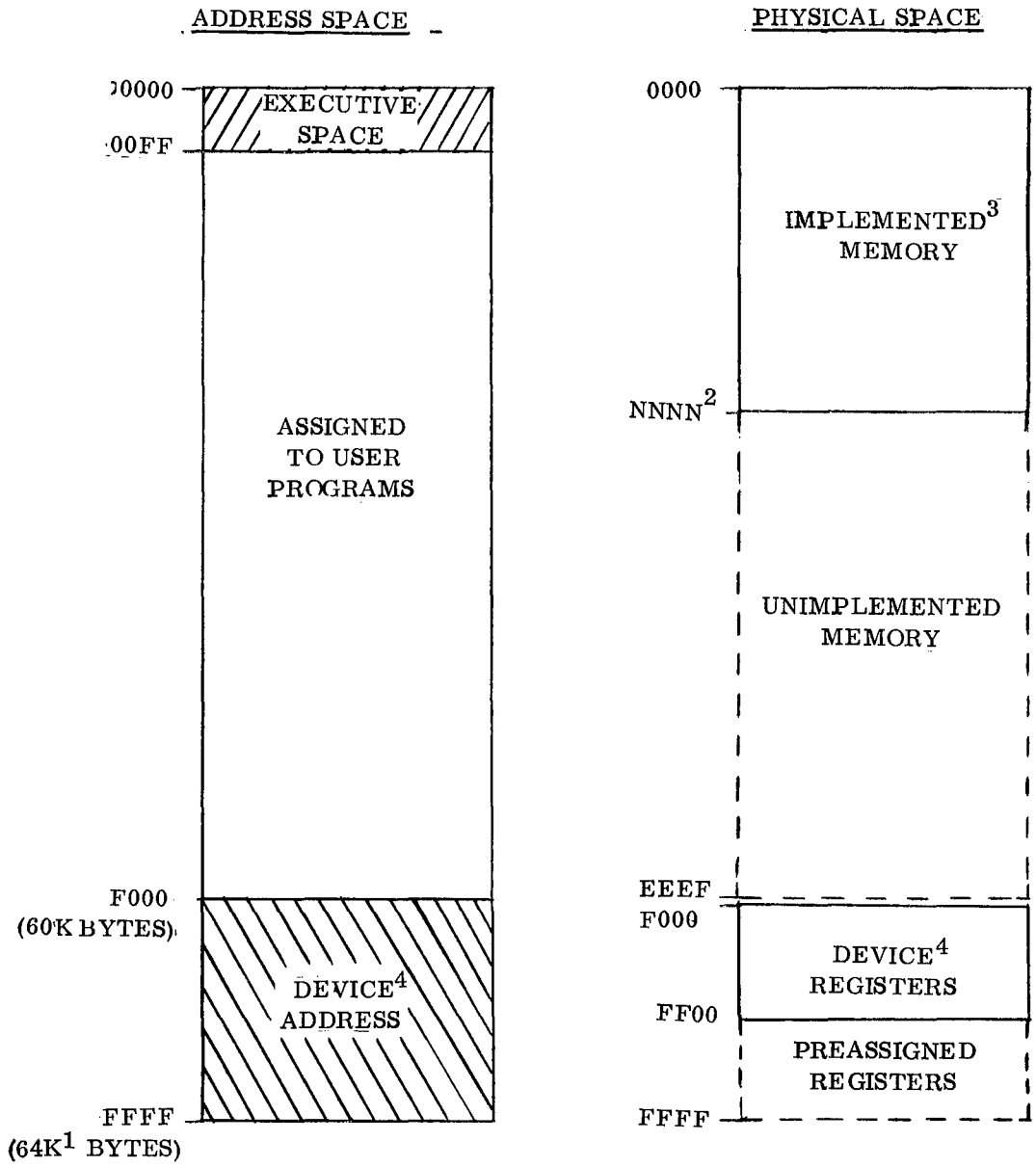
Address (Hex)	Input/Output Device Controller
F800	Teletypewriter No. 1
F810	Teletypewriter No. 2
F820	High Speed Paper Tape Reader No. 1
F830	High Speed Paper Tape Punch No. 1
F840	High Speed Paper Tape Reader No. 2
F850	High Speed Paper Tape Punch No. 2
F860	Card Reader No. 1
F870	Card Reader No. 2
F880	Card Punch No. 1
F890	Card Punch No. 2
F8A0	Line Printer No. 1
F8B0	Line Printer No. 2
F8C0	Magnetic Tape No. 1 (handles 4 Drives)
F8D0	Magnetic Tape No. 2 (handles 4 Drives)
F8E0	Bulk File No. 1 (Fixed Head)
F8F0	Bulk File No. 2 (Fixed Head)
F900	Disc File No. 1
F910	Disc File No. 2
F920	Cassette No. 1
F930	Cassette No. 2
FA00	CRT Display, Alphanumeric No. 1
FA10	CRT Display, Alphanumeric No. 2
.	.
.	.
.	.
FAF0	CRT Display, Alphanumeric No. 16

SYSTEM FUNCTION ADDRESSES

System functions that use the SUE interrupt structure are assigned the following module addresses:

Module Address		Interrupt Level
0001	Line Frequency	4
0002	Power Failure	4
0004	Power Restart	4
0001	External Attention	1
FF80	Operator Attention	1

Memory Map



Note:

1. K = 1024
2. The number of the last location of physically implemented memory.
3. May be mixed magnetic core and semiconductor. Address recognition of each block is physically adjustable by user.
4. Optional 2K byte area (Device Registers F800-FF00).

Figure G-1. Address Memory Allocation

APPENDIX H

INFIBUS TIMING INFORMATION

GENERAL

A. The following notes apply to the timing diagram at the end of this Appendix; the numerals correspond to the numerals on the waveforms of the same name.

B. (D) indicates timing and/or waveform at input of bus driver. Unless otherwise stated, driver is a non-inverting driver with integral receiver which does invert. Worst-case D-R pair delay is 50 ns; differential delay (skew) between drivers on the same Infibus (and therefore the same load) is no more than 25 ns.

C. (R) indicates timing and/or waveform at the output of a bus receiver. Unless otherwise stated, receiver is a part of D-R pair covered above. Receiver inverts logic signals.

SELECT CYCLE

1. SRLD-N initiates the bus controller select cycle if or when previous SACK is removed. SRLD must be negated as soon as possible after SACK is returned, and not before. SRL1, SRL2, SRL3, SRL4, and SRLC are equivalent in timing operation.

   t(SRLD ↑)-t(SACK ↓) > 0, 6-20 ns typical at (D)

2. SELD-N is returned by the bus controller in response to SRLD. (SEL1, SEL2, SEL3, SEL4, and SELC are the responses to similar requests, in order of priority). SELD is removed after SACK is received.

   t(SELx ↑)-t(SACK ↑): 4-10 ns typical at (D)

   There is no unique receiver for SELx and therefore it is used directly in the logic. This signal can have one load in each device.

3. PCDA-P is the precedence chain pulse received at the input to a given device. If the device is inactive on the level of the particular SELx currently asserted, PCDA is propogated by a single 74S11 or equivalent to become PCDB as an output. This is connected on the Infibus to become PCDA of the adjacent card. The pulse-width out of bus controller will be 40 ns min, 60 ns. maximum. The propagating gate must not distort the pulse width by more than ±1 ns. The bus controller will delay PCD following SELx by at least 25 ns. Each device will add propagation delay to PCD, causing a wide

variation in the interval between assertion of SELx and reception of PCDB. SUE logic is designed to accommodate this variation easily.

4. SACK-N is asserted by a device upon receipt of PCDA when the device is asserting a given SRLx and perceives the corresponding SELx to be asserted. The device must maintain this assertion of SACK until the current bus cycle is concluded by the removal of STRB, and this now-selected device has asserted its own STRB. SACK must be removed within 20 ns after the device sees its own STRB at the receiver. When SACK is received by the bus controller, SELx is negated (removed) and the select cycle is concluded.

   As soon as SACK is negated by the selected device, the bus controller will respond to any SRLx still asserted. If no SACK is received by the bus controller within 1.0 microsecond, the bus controller will assert SACK.

SERVICE CYCLE (TYPICAL)

1. STRB-N negated at the receiver output of a device selected and waiting for Bus Access will initiate the service cycle for that device. The device asserts its own STRB, and when detected by its own STRB receiver removes SACK.

2. SACK-N must be removed within 20 ns after STRB is asserted. (See above, and Select Cycle Description).

3. ABxx, RITE, BYTE, and HCYC (if used) are asserted as soon as possible after the recognition of an idle bus by a selected device. This typically takes 10-30 ns. There should be no more than 25 ns skew at the receivers of these lines. These lines are negated at the same time as STRB in the master. This results in a ±25 ns uncertainty at the end of STRB. For this reason slave address recognition and action initiation should use ABxx only at the leading edge transition of STRB.

4. STRB is asserted later than ABxx, etc. The same logic signal which gates ABxx (and DBxx in a write) is passed down a time delay whose minimum delay is at least as great as the relative skew between ABxx driver/receiver pairs. This will insure that STRB is the last of these signals to arrive. The maximum skew for D/R's is 25 ns. A delay line of 50 ± 5 ns. is used in SUE modules to accommodate the additional delay encountered in multiplexing data to DBxx. STRB should be removed as soon as possible after receipt of DONE.

5. DBxx (in write operations) must be stable within 20 ns. of STRB at the receivers of both signals. This is accomplished by previous data having been removed within 20 ns of negated STRB and proper delay of STRB assertion to accommodate delay of Data getting to DBxx drivers.

6. DBxx (in read operation) must preceed DONE (at the slave drivers) by 40 ns to insure a minimum of 15 ns differential at the receivers in the masters (for register set up) DBxx in the slave must be gated off within 20 ns after STRB is negated at the slave receivers. If STRB is removed at any time, even before DBxx or DONE is asserted, this must disable slave output of DBxx and DONE.

7. DONE in read is asserted a minimum of 40 ns after DBxx driver input is stable. DONE must be between 40 and 60 ns in length. In write, DONE is asserted as soon as data has been strobed into the receiving logic element.

8. Device access time (slave mode) is defined as the time between the leading edge of STRB and the leading edge of DONE at the receiver outputs.

9. Bus Cycle Time is defined as the time between the trailing edge of the previous STRB to the trailing edge of the current STRB. If this time exceeds 2.0 microseconds the bus controller will assert Quit. The Quit signal will persist until STRB is removed; if there has been no STRB, SACK must be removed.

SERVICE CYCLE-READ MODIFY-WRITE (RMW) AND/OR CORE MEMORIES

The descriptive paragraphs found under Service Cycle, (typical), also apply to RMW, with these additional conditions or exceptions:

- ABxx is not required to remain asserted between the Read half-cycle and the Write half cycle. No address is needed for the Write portion with SUE core memories. Most semiconductor memories will require the address to be asserted during the write portion of the (R-M-W) cycle.

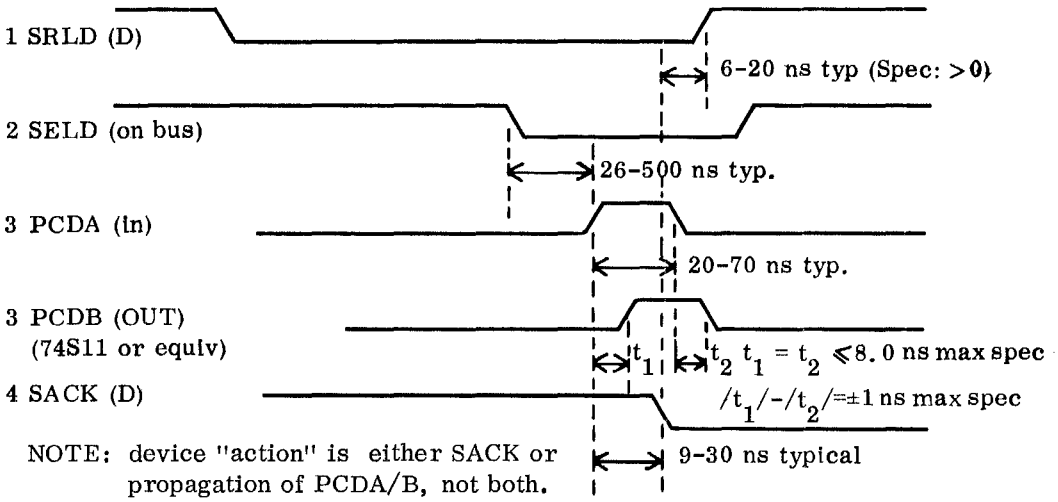
- STRB-N must remain asserted through the complete (R-M-W) cycle in order to retain control of the bus and of the memory cell being altered.

- DBxx asserted during the read portion must be negated by the slave device/memory 12.5 to 37.5 ns after DONE (25 ns preferred) even though STRB remains asserted.

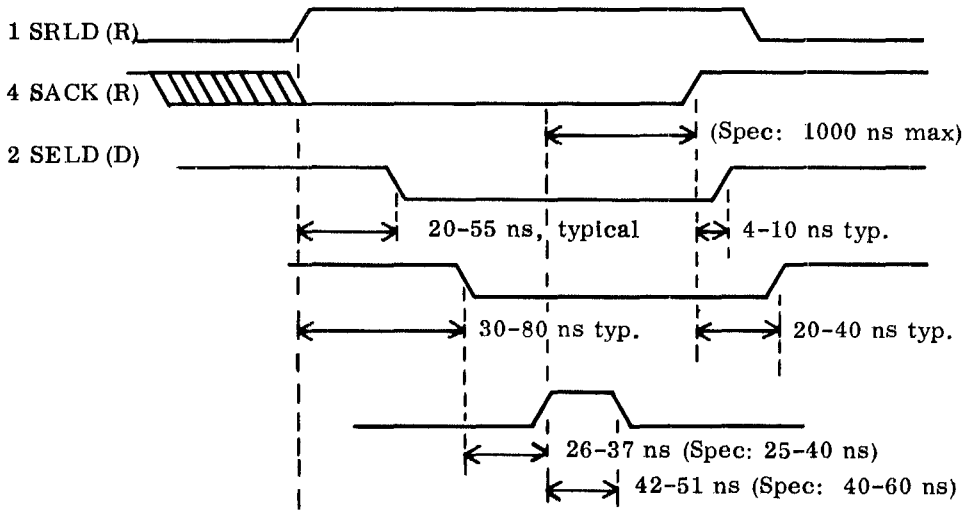
- DONE must be returned by the slave for both the read portion and the write portion. The read DONE may be asserted as soon as 37.5 ns after data in core memories, when taps on delay lines are spaced at 12.5 ns intervals.
- HCYC may be asserted for only the first (read) half-cycle.
- RITE serves to initiate the write portion as well as control writing.

I. Select Cycle begins when SRLx is asserted, or previous SACK is negated, whichever is last. Maximum of 1000 ns allowed.

A. Requesting Device (D = into driver; R - out of receiver)

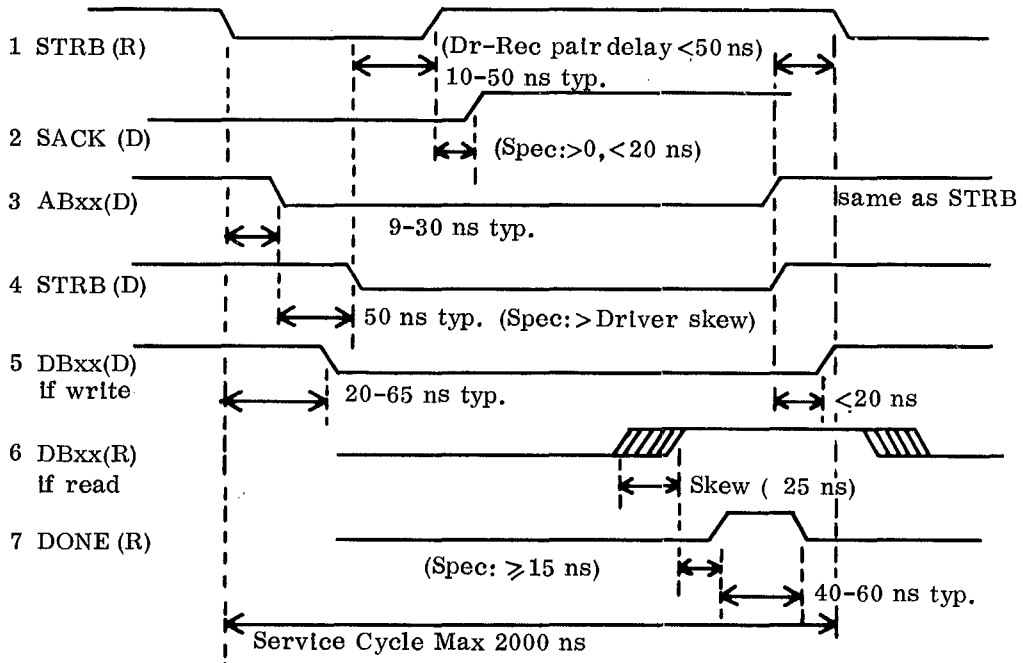


B. Bus Controller

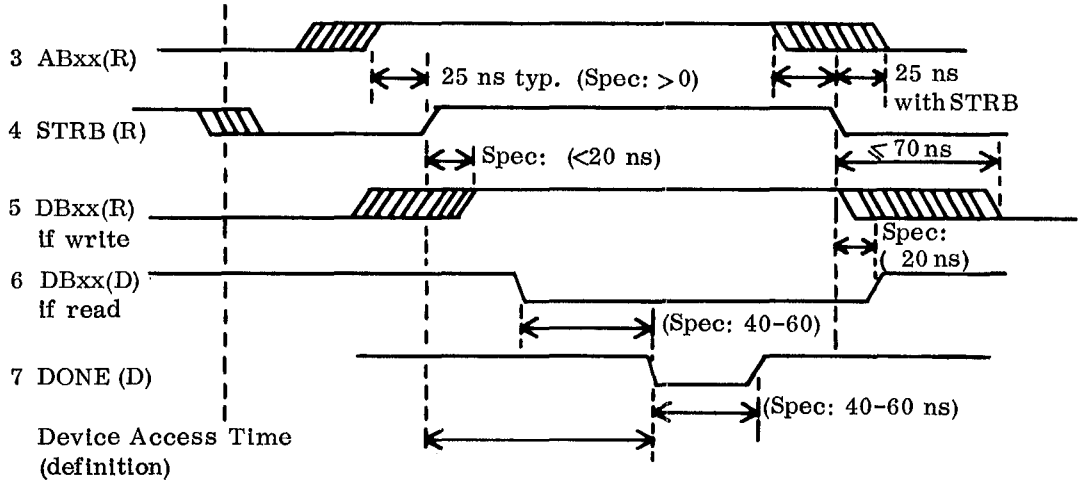


II. Service Cycle (typical) begins when SACK is asserted, or the previous STRB is negated, whichever occurs last. Maximum of 2000 ns allowed.

A. Master Device (was requesting device) (D = into driver; R = out of receiver)

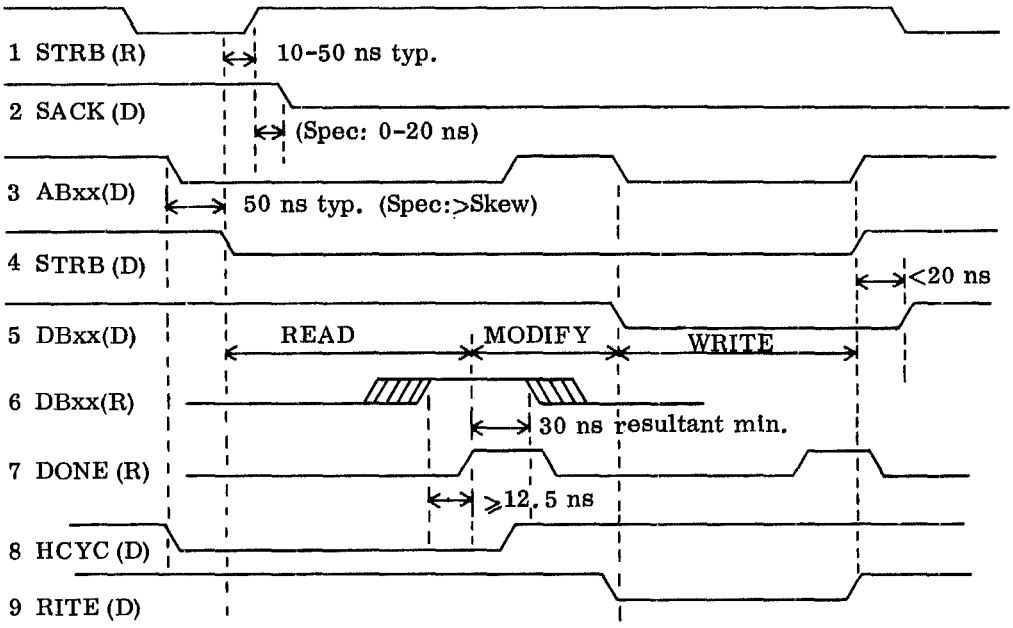


B. Target Device (except core memory)

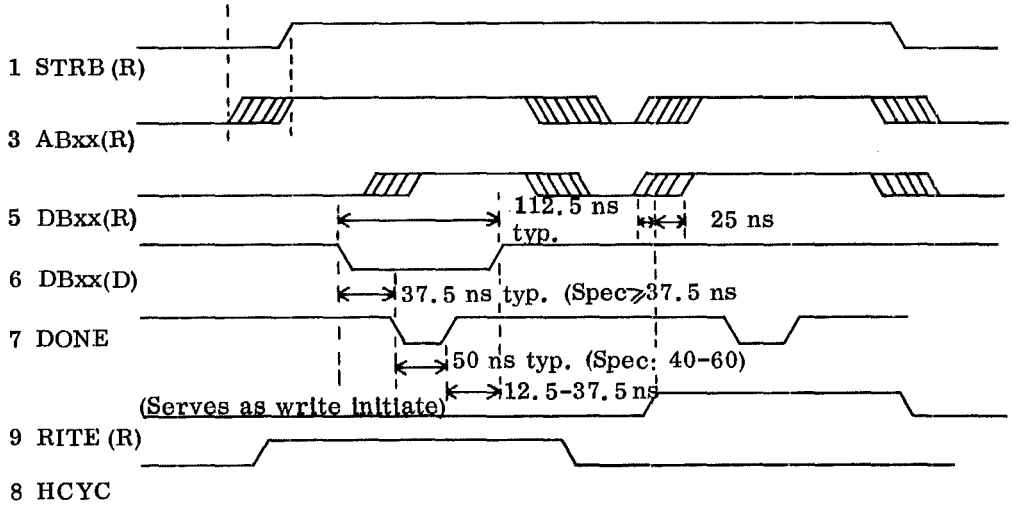


III. Service Cycle: Read-Modify Write

A. Master



B. Target



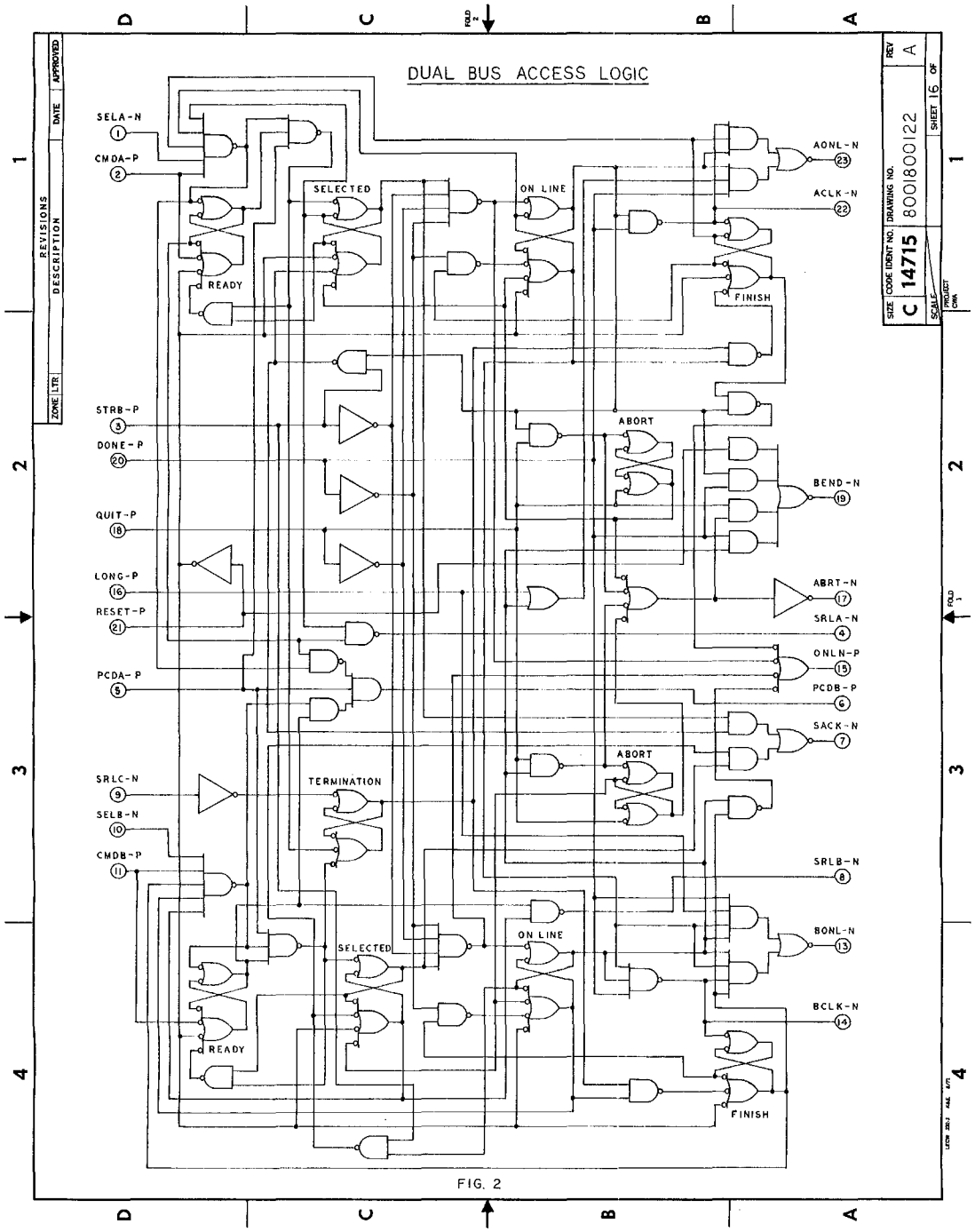

DUAL BUS ACCESS LOGIC
SELA-N
CMDA-P
SELECTED
ON LINE
READY
FINISH
AONL-N
ACLK-N
STRB-P
DONE-P
QUIT-P
LONG-P
RESET-P
PCDA-P
ABORT
BEND-N
ABRT-N
SRLA-N
ONLN-P
PCDB-P
SACK-N
SRLC-N
SELB-N
CMDB-P
TERMINATION
SRLB-N
BONL-N
BCLK-N
REVISIONS
DESCRIPTION
DATE
APPROVED
ZONE LTR
SIZE C
CODE IDENT NO. 14715
DRAWING NO. 8001800122
REV A
SHEET 16 OF
SCALE
PROJECT CWA

FIG. 2

APPENDIX I

INFIBUS PIN ASSIGNMENTS

Pin #	A Side (Note)	B Side (Note)	Pin #	A Side (Note)	B Side (Note)
01	GND	GND	28	+5V	+5V
02	GND	GND	29	+5V	+5V
03	+15V	+15V	30	BT1A(4)	BT1B(4)
04	+15V	+15V	31	BT2A(4)	BT2B(4)
05	MINH(2)	ATLD(2)	32	BT3A(4)	BT3B(4)
06	MRES(3)	PWST(2)	33	BT4A(4)	BT4B(4)
07	REPB(1)	LFRQ(1)	34	SRLC(3)	SELC(3)
08	RUNN(1)	ØPAT(1)	35	SRLD(3)	SELD(3)
09	KEY0(alt.)	KEY1(alt.)	36	SRL1(3)	SEL1(3)
10	AB00(2)	AB08(2)	37	SRL2(3)	SEL2(3)
11	AB01(2)	AB09(2)	38	SRL3(3)	SEL3(3)
12	AB02(2)	AB10(2)	39	SRL4(3)	SEL4(3)
13	AB03(2)	AB11(2)	40	GND(5)	GND(5)
14	AB04(2)	AB12(2)	41	DB00(2)	DB08(2)
15	GND(5)	GND(5)	42	DB01(2)	DB09(2)
16	+5(5)	+5(5)	43	DB02(2)	DB10(2)
17	AB05(2)	AB13(2)	44	DB03(2)	DB11(2)
18	AB06(2)	AB14(2)	45	DB04(2)	DB12(2)
19	AB07(2)	AB15(2)	46	DB05(2)	DB13(2)
20	KEY0/AB16(2)	KEY1/AB17(2)	47	DB06(2)	DB14(2)
21	PRAL(1)	PFIN(1)	48	DB07(2)	DB15(2)
22	PRIN(1)	LFIN(1)	49	PBLØ(2)	PBHI(2)
23	RITE(3)	BYTE(3)	50	CLKA(3)	SPARE
24	HCYC(3)	HØLD(3)	51	+5V (5)	+5V (5)
25	PCDA(4)	PCDB(4)	52	-15V	-15V
26	SACK(3)	QUIT(3)	53	-15V	-15V
27	DØNE(3)	STRB(3)	54	GND	GND
			55	GND	GND

Notes:
(1) 1000 Ohms to +5V
(2) 2 x 150 Ohms to +5V
(3) 2 x 100 Ohms to +5V
(4) No term. Resistor; A side Connects to B side of adjacent connector. All modules not utilizing these lines must short the A side to the B side; i.e., Jumper 25A to 25B if PCDA/B is not used.
(5) Utilize these for Bus Drivers on plug-in PC boards (2 layer boards only).

APPENDIX J

SPECIAL PURPOSE PROCESSORS

Special purpose processors may be bundled into specific hardware configurations and not available separately.

SUE 1111 PROCESSOR INSTRUCTION SET

The SUE 1111 Business Processor contains the complete instruction set of the 1110 Processor, thus assuring 1110 user's full upward compatability. In addition, the 1111 contains nine memory to memory decimal arithmetic and character string manipulation instructions specifically designed to enhance the execution of programs written in the RPG (Report Program Generator) language.

SUE 1111 DECIMAL ARITHMETIC INSTRUCTIONS

The six memory to memory decimal arithmetic instructions operate on "packed decimal" data. Two binary coded decimal digits are packed in each byte, except the low order (rightmost) byte, which contains one digit and the sign. A positive number has the sign "C_{16}," a negative number, "D_{16}," with all non-valid signs treated as positive.

The six decimal arithmetic instructions are:

ZADD — Zero and Add - Moves the source field contents into the destination field.

ADDD — Add Decimal - Adds the signed source field operand to the signed destination field operand, with the sum stored in the destination field.

SUBD — Subtract Decimal - Numerically subtracts the signed source field operand from the signed destination field operand, with the difference placed in the destination field.

SFTR — Shift Decimal Right - The decimal number is shifted to the right. The sign is unchanged.

SFTL — Shift Decimal Left - The decimal number is shifted to the left. The sign is unchanged.

CMPD — Compare Decimal - Numerically compares the signed source field operand to the signed destination field operand.

SUE 1111 CHARACTER MANIPULATION INSTRUCTIONS

These three instructions operate on ASCII coded data.

MOVR Move Field, from Right to Left - The source field character string is moved into the destination field, one character at a time, beginning with the rightmost character.

MOVL Move Field, from Left to Right - The source field character string is moved into the destination field, one character at a time, beginning with the leftmost character.

COMP Compare Fields - The source and destination character strings are compared, one character at a time from left to right. The compare assumes the ASCII collating sequence.

SUE 1112 PROCESSOR INSTRUCTION SET

The SUE 1112 Scientific Processor set contains the complete instruction set of the 1110 processor; thus, assuring 1110 users of full upward compatibility. In addition, the 1112 has 36 extended arithmetic instructions.

SUE 1112 EXTENDED ARITHMETIC INSTRUCTIONS

The extended arithmetic instructions are divided into seven groups: Bit manipulation; additional move instructions; normalize, count operations; additional control instructions; double length shift operations; single precision fixed point operations; double precision fixed point operations.

BIT MANIPULATION INSTRUCTIONS

Mnemonic	Description
RBIT	Make the designated bit or bits a 0, all others unchanged.
SBIT	Make the designated bit or bits a 1, all others unchanged.
CBIT	Change the designated bit or bits, all others unchanged.
IBIT	Isolate the designated bit or bits, all others equal to 0.
TSBT	Test the designated bit or bits and shift register (AR) left by one. The bit shifted out of $(AR)_{15}$ is lost and a zero is shifted into $(AR)_0$.
TBIT	Test the designated bit or bits.

MOVE INSTRUCTIONS

Mnemonic	Description
MOVT	Move the two's complement value of register XR to register AR. The contents of XR is unchanged.
MOVO	Move the one's complement value of register XR to register AR. The contents of XR is unchanged.
MOVP	Move the positive magnitude of register XR to register AR. The contents of XR is unchanged.
MOVN	Move the negative magnitude of register XR to register AR. The contents of XR is unchanged.

SINGLE PRECISION FIXED POINT OPERATIONS

Operation	Description
MLTA (Multiply, Add)	Multiply the data in the odd numbered register by the effective operand, add the contents of the even numbered register, generating a two-word product in the combined registers.
MULT (Multiply)	Multiply the data in the odd numbered register by the effective operand, generating a two-word product in the combined registers.
DIVD	Divide the data in the two register accumulator by the effective operand, generating a properly signed quotient in the odd-numbered register, with the remainder (in the even-numbered register) having the same sign as the original dividend.

DOUBLE PRECISION FIXED POINT OPERATIONS

Mnemonic	Description
DLOD	Move the contents of the two consecutive words located at the effective address to the combined registers.
DSTA	Move the contents of the two registers to the two consecutive words located at the effective address.
DADD	Add the contents of the two consecutive words located at the effective address to the two registers.
DSUB	Subtract the contents of the two consecutive words located at the effective address from the two registers.

NORMALIZE, COUNT OPERATIONS

Mnemonic	Operation Performed
SLAN	Single left arithmetic normalize
SLLN	Single left logical normalize
SRAN	Single right arithmetic normalize
SRLN	Single right logical normalize
DLAN	Double left arithmetic normalize
DLLN	Double left logical normalize
DRAN	Double right arithmetic normalize
DRLN	Double right logical normalize

CONTROL INSTRUCTIONS

Mnemonic	Description
JKEY	Store the value K into the key bits and the address M into the program counter (i. e. jump).
LCPU	Load the CPU number into XR bits 5-6. The remaining bits in XR are cleared.
LKEY	Load the key bits into XR right justified.

DOUBLE LENGTH SHIFT OPERATIONS

Mnemonic	Operation
DLAO	Double Left Arithmetic Open
DLLL	Double Left Logical Linked
DLLO	Double Left Logical Open
DLLC	Double Left Logical Closed
DRAO	Double Right Arithmetic Open
DRLL	Double Right Logical Linked
DRLO	Double Right Logical Open
DRLC	Double Right Logical Closed

APPENDIX K

SUE AUTOLOAD OPTIONS

SUE AUTOLOAD

SUE autoload modules are designated as follows:

1240	General Purpose, ROM installed by User
1241	Paper Tape (Teletype and High Speed Paper Tape Reader)
1242	Disc (Removable and Fixed)
1243	Card Reader
1244	Magnetic Tape (Cassette and Transport)
1251	Paper Tape and Disc
1252	Paper Tape and Card Reader
1253	Card Reader and Disc
1261	Paper Tape, Disc and Card Reader

PHYSICAL DESCRIPTION

Each autoload module consists of one printed circuit board that may contain from one to four 256 x 4-bit ROMs. Each ROM contains a bootstrap basic loader.

Each autoload board has four Singer T8001 toggle switches (S1 through S4) and a 3M 3428-1002 20-pin right angle connector (J1) mounted on its free edge. Switches S1 and S2 are used for ROM selection, S3 and S4 for device selection. The connector J1 enables remoting of the ROM and Device Select, Reset, Autoload and External Attention functions.

FUNCTIONAL DESCRIPTION

Assertion of RESET (REPB) followed by AUTOLOAD (ALTD) initiates the Autoload operation. RESET and AUTOLOAD may be asserted by the control panel pushbuttons or remotely via Connector J1.

The Autoload module first transfers the start address of the selected ROM Basic Loader into Location 0006, the Level 1 interrupt service routine vector address. All LEC ROM Basic Loaders start at FB00; however, jumpers on the Autoload board allow customer ROMs with start addresses in the range E000 to FF00.

The Autoload module then automatically generates a Level 1 interrupt. The interrupting device address (to be stored in core location 0000) will be 0, 2, 4 or 6 as determined by switches S3 and S4 which set bits 1 and 2 of the device number. Those ROM basic loaders with a choice of physical input devices will interrogate Location 0000 to determine which input device to utilize.

The SUE Processor, in servicing the Level 1 interrupt, vectors to the start address of the ROM Basic Loader and begins normal program execution.

ROM AND DEVICE SELECT

Switches S1 and S2 select which of four ROM basic loaders will be executed: Paper Tape, Cards, Disc or Magnetic Tape. Switches S3 and S4 select a physical input device. The ROM/Device selection is summarized in the following table:

	Switch Positions			
Input Device	S1	S2	S3	S4
Paper Tape (Teletype)	Down	Down	Down	Down
Paper Tape (High Speed Reader)	Down	Down	Up	Down
Disc (Fixed)	Down	Up	Down	Down
Disc (Removable)	Down	Up	Up	Down
Card Reader	Up	Down	Down	Down
Magnetic Tape Unit 0	Up	Up	Down	Down
Magnetic Tape Unit 1	Up	Up	Up	Down
Magnetic Tape Unit 2	Up	Up	Down	Up
Magnetic Tape Unit 3	Up	Up	Up	Up

REMOTE ROM/DEVICE SELECTION

The contact connectors of Switches S1 through S4 are connected in parallel with the contact pairs, J1A1-J1B1 through J1A4-J1B4, respectively. Because the connector and switch contacts are in parallel, the switches must be OFF (DOWN) for operation via J1 and, conversely, the J1 connectors must be "OFF" (OPEN) for Device/ROM selection via S1-S4.

REMOTE AUTOLOAD OPERATION

RESET and AUTOLOAD, normally initiated by Control Panel pushbuttons, can also be initiated via Connector J1. J1A5-J1B5 initiates RESET (REPB-N), J1A6-J1B6 initiates REMOTE AUTOLOAD (RALD-N). At least a 50-millisecond delay is required between the assertion of RESET and AUTOLOAD. An 80-nanosecond (minimum) negative going pulse is recommended for RALD-N.

Although not required to remotely initiate the AUTOLOAD function, External Attention (EXAT) can also be asserted via Connector J1, Pins J1A7 to J1B7. EXAT must be a negative going pulse of 100-microsecond minimum duration. A jumper between E1 and E2 on the Autoload board must first be installed.

APPENDIX L

HEXADECIMAL ADDITION AND MULTIPLICATION TABLES

HEXADECIMAL ADDITION TABLE

	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	1
2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	2
3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12	3
4	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13	4
5	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14	5
6	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15	6
7	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16	7
8	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17	8
9	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18	9
A	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19	A
B	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	B
C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	C
D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	D
E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	E
F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	F
	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

HEXADECIMAL MULTIPLICATION TABLE

	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
1	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	1
2	2	4	6	8	A	C	E	10	12	14	16	18	1A	1C	1E	2
3	3	6	9	C	F	12	15	18	1B	1E	21	24	27	2A	2D	3
4	4	8	C	10	14	18	1C	20	24	28	2C	30	34	38	3C	4
5	5	A	F	14	19	1E	23	28	2D	32	37	3C	41	46	4B	5
6	6	C	12	18	1E	24	2A	30	36	3C	42	48	4E	54	5A	6
7	7	E	15	1C	23	2A	31	38	3F	46	4D	54	5B	62	69	7
8	8	10	18	20	28	30	38	40	48	50	58	60	68	70	78	8
9	9	12	1B	24	2D	36	3F	48	51	5A	63	6C	75	7E	87	9
A	A	14	1E	28	32	3C	46	50	5A	64	6E	78	82	8C	96	A
B	B	16	21	2C	37	42	4D	58	63	6E	7F	84	8F	9A	A5	B
C	C	18	24	30	3C	48	54	60	6C	78	84	90	9C	A8	B4	C
D	D	1A	27	34	41	4E	5B	68	75	82	8F	9C	A9	B6	C3	D
E	E	1C	2A	38	46	54	62	70	7E	8C	9A	A8	B6	C4	D2	E
F	F	1E	2D	3C	4B	5A	69	78	87	96	A5	B4	C3	D2	E1	F
	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

Lockheed Electronics Company, Inc.
Data Products Division
6201 East Randolph Street
Los Angeles, California 90040

Printed in U.S.A. 5M 7/73 copyright Lockheed Electronics Co., Inc.