G. W. Orr

SAFEGUARD SYSTEMS
PROGRAMMER'S GUIDES

VOLUME I
INTRODUCTION TO THE SAFEGUARD DATA PROCESSING SYSTEM

Bell Telephone
Laboratories

SAFEGUARD SYSTEMS
PROGRAMMER"S GUIDES

GCXDPSIN@SY00XXXX
28 April 1972
SAFEGUARD System - Preliminary

VOLUME 1

INTRODUCTION TO THE SAFEGUARD DATA PROCESSING SYSTEM

Prepared by Bell Telephone Laboratories
Park Avenue, Madison, N.J. 07940
on behalf of Western Electric Company
83 Maiden Lane, New York, N.Y. 10038
for the U. S. Army SAFEGUARD System Command
under contract DAHC60-71-C-0005

RECOMMENDED CHANGES

Please address all suggested changes to this manual to
R. A. Judson, x4808, ML-1J107

Type of change requested:

____ Technical
____ Editorial
____ Change needed for clarity

Sections or pages to be changed: _____

Recommended changes (attach marked-up pages if that is easier:

Name: _____ Dept.: _____

Location: _____ Extension: _____

The ladie saith to Sir Gawayne, "Knighte, thy armor clanketh much like hardware." And Gawayne saide unto her, "Miladie, my hardware is my safeguarde."

--R. A. Nosduj

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 SCOPE, AUDIENCE, AND APPROACH

This document is the first volume in a set entitled Systems Programmer's Guides to the SAFEGUARD Data Processing System. This particular volume is intended as a brief introduction to some of the highlights of the system, with particular emphasis on the central computer; it is aimed toward the systems programmer who is just beginning to hack through the jungle of hardware before him. The succeeding manuals, much more detailed and technical in scope, deal with separate components of the system, viz., the central computer (Vol. 2: Central Logic and Control) and the peripheral subsystems that are supported by the central computer (Vol. 3: Recording Subsystem, Vol. 4: Display Subsystem; etc.)

The complete set of documents has been approached with the following ideas in mind:

1. Insofar as possible, the hardware has been presented from a "black box" point of view. For example, assume an instruction that causes the machine to add the contents of

registers A and B and place the result in register C. If the machine actually performs the addition in some internal area that cannot be accessed or manipulated by the programmer, that internal area has not been mentioned.

2. One of the major problems inherent in any species of documentation is the problem of precise terminology. Therefore, within the limits of the authors' abilities, terms special to the system have been defined, standardized, and used consistently. In a few cases, this has meant a change in language familiar to the systems programmers who have been on the SAFEGUARD project for several years, but any changes have always been made for the sake of simplicity and consistency.

3. Sections on "programming techniques" are outside the scope of these manuals. The object of these guides is to provide the systems programmer with tools and materials but not with a set of blueprints.

1.2 COMPUTER SYSTEMS IN GENERAL

Generally, the nucleus of a computer system is the central processing unit (CPU)[1], which is wired to memories

---

[1] There can be more than one CPU, one functioning as a "master" and the other as a "slave", or several working in parallel, but the generalities above still apply.

and is also wired to a "black box" that controls such input/output units as card readers and disk. The CPU is the only device in the system that can do its own bookkeeping; it functions as the "brain" of the system to perform a multitude of tasks, including:

1. Fetching and storing data and programs to and from memory.

2. Fetching data from the input/output devices.

3. Controlling input/output operations by initiating them, terminating them, and, whenever necessary, keeping track of what the I/O devices are doing.

4. Performing arithmetic operations (such as subtraction) and logical operations (such as an OR of two bit patterns).

5. Performing all system bookkeeping and handling all system problems (for example, discontinuing communication with a printer after the printer runs out of paper or falls through the floor).

Figure 1 shows the kinds of functions a CPU is likely to perform.

1.3 SAFEGUARD COMPUTER SYSTEM

The general applications[2] CPU must be able to handle a variety of tasks, many of which can be run at

---

[2] "General applications" or "general purpose" pertains to a commercially built computer capable of handling anything from inventory updating to excruciating mathematical calculations. The IBM 360 is an example of a general purpose computer.

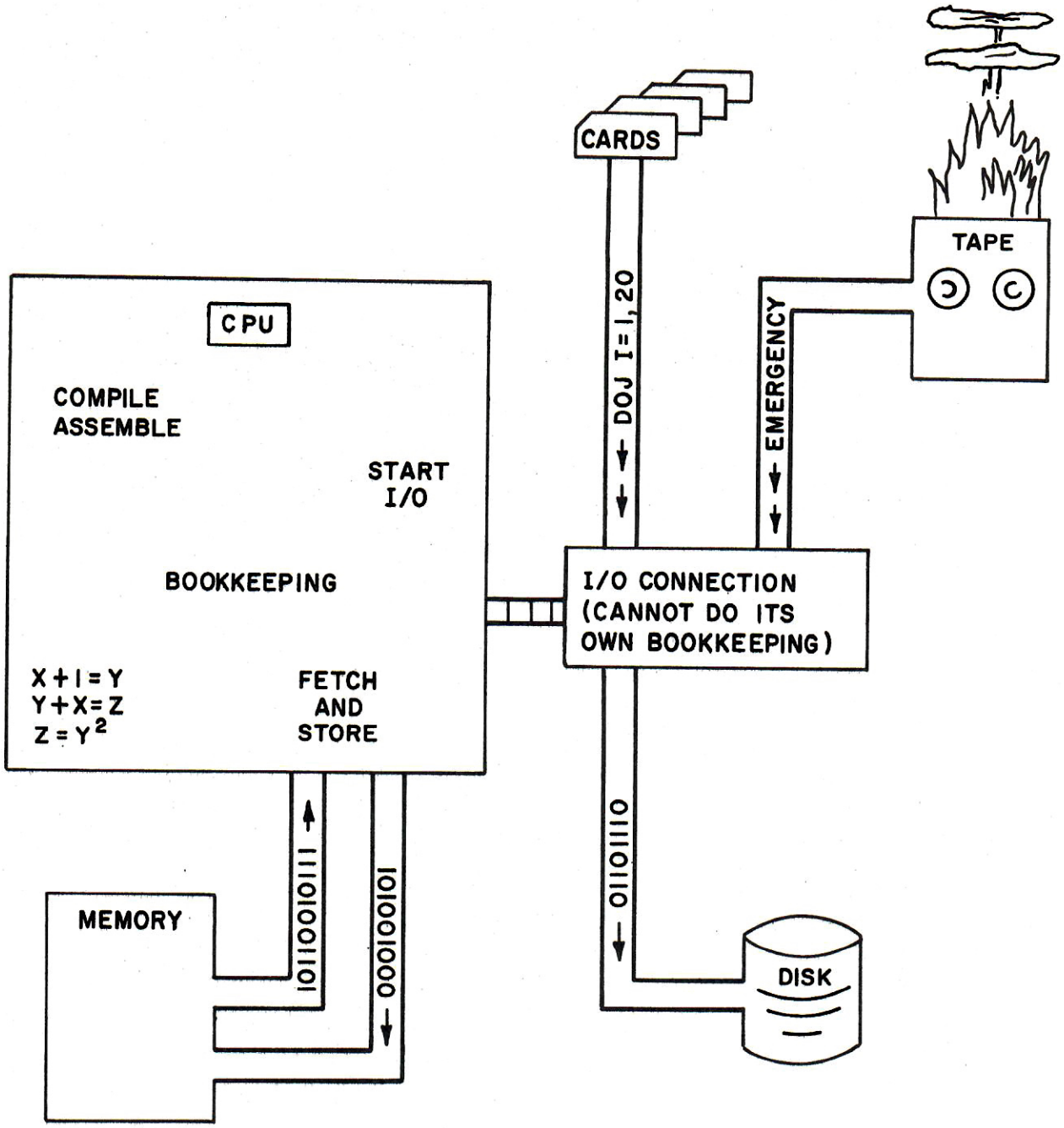


FIGURE I. SAMPLE FUNCTIONS PERFORMED BY A CPU

"off hours": assembling and compiling source languages, taking care of the bookkeeping for I/O jobs, and executing applications programs, such as payroll processing routines.

The CPU for SAFEGUARD, called the Central Logic and Control (CLC), is not a general applications computer. Rather, it is a highly specialized, dedicated machine--dedicated, that is, to supporting the needs of a real-time missile defense system. This "dedication" implies that the niceties of assembling and of linking subroutines to their routines are completed on a support machine and that they are not necessary on the CLC[3]; it implies that "goodies" such as inventory control routines have no place on the CLC. It further implies that the hardware has been designed to make each piece highly specialized and efficient.

A general purpose CPU might start executing a program. During the course of that execution, the CPU might initiate an I/O operation (and the I/O device might be wired to the CPU, rather than operated as a programmable "computer"); the CPU will undoubtedly have to do all the bookkeeping for the I/O device. On the CLC, however, these jobs will be performed by separate programmable

---

[3] An introduction to the support software is contained in the Tactical Support Software Overview Manual, Part 1, GCXSUPRT@UM01XXXX, 15 June 1971.

"computers" operating relatively independently of each other: a processor unit (PU)[4] could be executing a weapons process routine while an input/output controller (IOC) is transferring data from a disk unit to CLC memory and is doing its own bookkeeping.

In the CLC system, then, the functions usually performed by a single CPU have been divided into separate but interrelated hardware units (see Figure 2). There are two species of "computers," one for arithmetic and logical operations (the PU) and a second for I/O communication (the IOC). In addition, there are memory units used for the storage of instructions and data. There is one unit that functions as an alarm clock (the timing generator, or TG) so that, for example, certain operations might be repeated at specific intervals. And there is another unit (the status unit, or SU) that senses problems and changes in status (for example, if someone pulls the plug on a processor unit).

Again, these units are entirely separate pieces of hardware, but there must be communication between them.

---

[4] It is to be noted that although only one of each kind of unit will be referred to throughout this volume, in reality several of the same kind might be working at the same moment. For example, the system is wired to handle fifteen PU's.

The PU may need to tell the IOC to set an I/O program in motion. Or the IOC may need to set a bit in the status unit to indicate that faulty data has been received. The communication between CLC units, as well as the function of each unit, is discussed in Chapters 2 and 3.

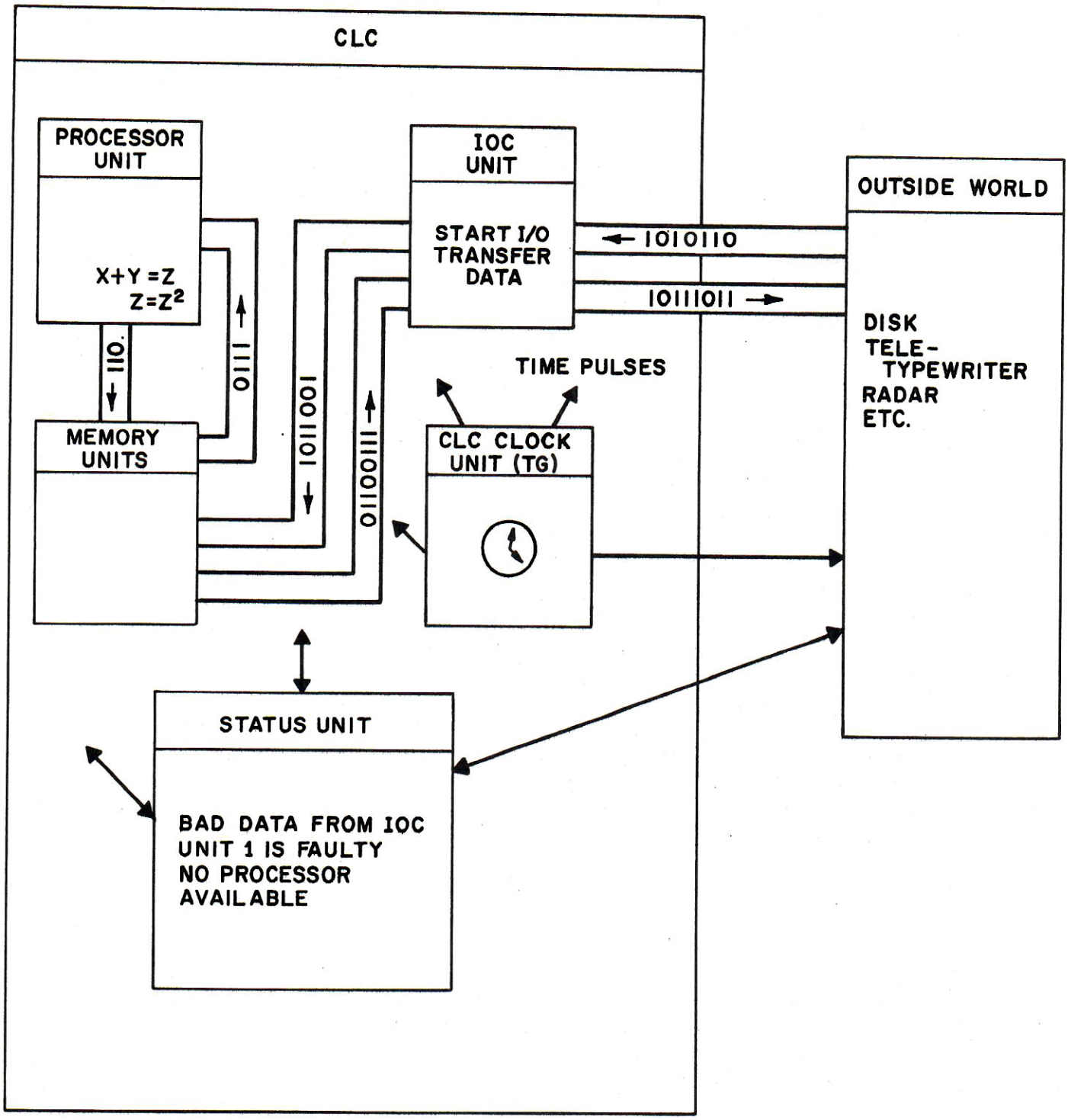


FIGURE 2. FUNCTIONS PERFORMED BY THE CLC

Chapter 2

CENTRAL LOGIC AND CONTROL

2.1 INTRODUCTION

The SAFEGUARD Central Logic and Control (CLC) consists of the following kinds of units:

1. Two varieties of programmable computing units: one, the processor unit, or PU, for arithmetic and logical manipulations; the other, the input/output controller, or IOC, for handling communications between the CLC and the world outside[1] the CLC.

2. Internal storage units: one type, program store, or PS, to hold PU instructions, the second type, variable store, or VS, to hold data and IOC instructions. In addition, there is a special unit, the store transfer unit, or STU, used to transfer data between VS and PS.

3. An alarm clock, the timing generator, or TG, used to make sure that programmer-specified actions take place on time.

---

[1] This "outside world" is comprised of individual "peripheral subsystems," which will be covered briefly in Chapter 4. But as a hint at this point in time, one of the peripheral subsystems consists of tape drives, disk units, card readers, and printers.

4. A unit that records the status or changes in status of other pieces of hardware in the CLC and in the world outside the CLC (the status unit, or SU).

In this chapter, the basic functions of these units are described in enough detail to lead the systems programmer into Volume 2 of this series[2].

2.2 CLC COMPUTING UNITS

The two types of computing units discussed here are the processor unit (PU) and the input/output controller (IOC). They have this in common: both are the only genuinely programmable units of the CLC.

2.2.1 Processor Unit

The features of the PU are these:

1. It is a register-oriented machine. This implies that for most instructions the programmer specifies the particular registers involved in the operations to be performed. For example, the programmer might wish to add the values contained in two storage locations. He can fetch the values into two registers and then specify something like:

```
ADD   reg1,reg2
```

The contents of the two registers, reg1 and reg2 are added, and the result replaces the contents of an accumulator register which the programmer can access.

---

[2] Systems Programmer's Guide to the SAFEGUARD Central Logic and Control.

2. The PU performs the following kinds of operations:

   a. Arithmetic - for example, the contents of two registers might be added, or a square root might be calculated.

   b. Logical - for instance, the bit patterns contained in two registers might be ORed.

   c. Control - for example, a jump may be taken from one point in a program to another.

   d. Bit manipulation - for example, the contents of a particular register might be shifted left a specified number of places.

   e. Fetch and store[3] - for instance, the contents of a specified register might replace the contents of a slot in storage.

How and where the PU obtains its instructions and data will become clearer as this chapter develops.

2.2.2 Input/Output Controller

Like the PU, the IOC is programmable. However, while the PU is register-oriented, the IOC is a

---

[3] Fetches and stores are the key means by which the PU communicates with other CLC units. When a PU issues a "fetch" instruction, in effect it grabs a piece of data from storage or from a register in another CLC unit so that it can work on the data internally. When it issues a "store", it puts a piece of data into a slot in storage.

storage-oriented computer. This implies that the programmer does not specify a register in which a particular operation is to be performed.

An IOC instruction[4] consists of information on the operation to be performed and the fetch and store locations in storage. The "Set Bit to Logic 1" instruction illustrates this point:

```
SBl  bit,n,word
```

In this case, a specific bit of a word in storage is set to one. (The n field indicates whether or not the IOC program is to end after this instruction.) The operation, then, is performed on a word in storage, rather than on a word in a register.

The instruction set for the IOC causes the IOC to perform the following kinds of operations:

1. Basic I/O - for example, a specified number of data words might be transferred from internal CLC storage to the "outside world." During this kind of operation, the IOC will perform its own bookkeeping (e.g., determining when the specified number of data words has been transferred).

2. Control - for instance, there might be a jump to another IOC instruction under particular circumstances.

---

[4] In several documents on the IOC, the term "indirect command word", or "CWi", is used. Throughout this series of manuals the term "instruction" will replace "CWi".

3. Fetch and store - for instance, the contents of one storage location might replace the contents of another slot in storage.

4. Bit manipulation - for example, a specific bit in a word in storage might be set to 1.

Note that an IOC and a PU will probably be operating asynchronously. That is, although the two "computers" are not totally independent of each other, the IOC might be pulling in data from a disk unit at the same time that a PU is multiplying two values.

2.3 STORAGE

2.3.1 PS and VS

The internal CLC storage units containing processor unit instructions are called program store units (PS), and those used to hold IOC instructions, data, and certain control information (such as control pointers for use by the IOC in performing its operations) are called variable store units (VS).

PS can be read from by a processor unit and the store transfer unit; it can be written into solely by the store transfer unit (see the next section). VS can be read from or written into by a processor unit or IOC.

The general relationship between the CLC "computers" and internal storage, based on information

supplied up to this point in Chapter 2, is shown in Figure 3. Note that Figure 3 is extremely simplified; the CLC contains many of each type of unit.

2.3.2 Store Transfer Unit

The only unit capable of writing into program store is the store transfer unit (STU).

The instructions that are to be placed in PS are originally stored as "data" in VS (or outside the CLC, on disk). A processor unit or IOC sends a command[5] to the STU telling the STU to start filling up slots in program store. The dialogue between units continues like this:

IOC (picking up a piece of data from VS and handing it to the STU): Here is a piece of data to be stored.

STU (after filling up a slot in PS): Okay, it's done.

IOC: Here's some more.

STU: Okay, I've put it away.

IOC: That's it. I'll let you know when there's another job to do.

---

[5] The term "command" is explained in Chapter 3.

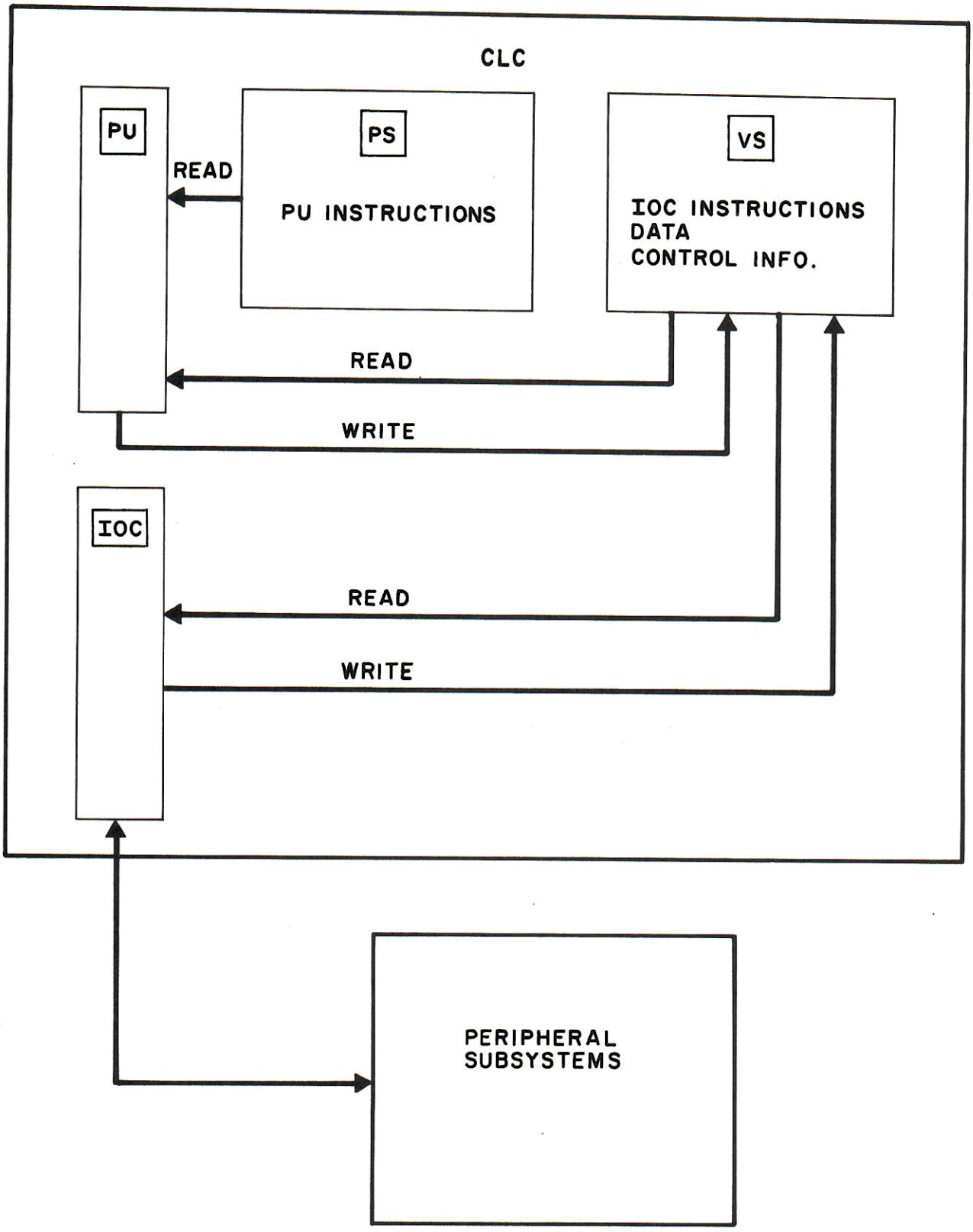


FIGURE 3. CLC "COMPUTER" UNIT/STORAGE RELATIONSHIP

2.4 TIMING GENERATOR

The timing generator (TG) contains time-of-day registers which are constantly updated by a master system clock. These registers can be accessed by a PU or IOC (via fetch instructions, in much the same manner that slots in storage are accessed via fetch instructions). Further, the outside world (that is, a peripheral device) can "tell" the TG to record the time when a particular event occurs. In addition, the TG can send commands to an IOC asking the IOC to start a program at a specified time or at a selected series of intervals.

In general, the TG operates in the following manner. (Note that the "request list" referred to below is stored in VS and that its items are picked up by the IOC. One request, for example, might be a request for the TG to send a pulse to an MSR at a particular time.)

PU or IOC:	Hey! I command you to be enabled.
TG:	Yes, sir. What do you want me to do?
IOC:	Here's an item from the request list. I want immediate action on this.
TG:	Yes, sir. (Does its thing) I'm finished. Give me another one.
IOC:	Here it is.
TG:	Okay, I'm finished. Anything else?
IOC:	No. Zap, zap, zap, you're disabled.

2.5 STATUS UNIT

2.5.1 General Functions of the SU

The status unit (SU) is a big box of registers hooked up to virtually every piece of hardware in the CLC and the world outside of hardware units the CLC. There is a register to record the status of a program store unit (for example, to indicate if the power is on), another for a disk drive controller, another for a PU, etc., etc., etc.

The SU maintains control of system hardware resources by providing the following services:

1. It accepts signals that a unit's status has changed (for example, if somebody has pulled the plug on a processor unit).

2. It collects information about errors found by other units (for instance, a bit will be set in an SU register if an illegal operation code is received by the TG).

3. The SU can pass control signals to other CLC units stating that this or that unit is not to be used in particular configurations of the system.

4. It can send control signals to the outside world.

5. It can light up a fancy status display board so that people standing around can see what is going on inside the hardware units.

6. Under particular circumstances, the SU tattles to control software that an event has occurred in a hardware unit.

The SU's registers can be accessed by a PU or IOC (via fetch instructions). Some of the bits in the registers can be set by software, and others (for instance, those registering information that power has been switched on) can only be changed by hardware signals.

2.5.2 Green and Amber Systems

The units of the CLC can be partitioned into two systems, a Green System and an Amber System.[6] Status units are central to the control of partitioning, because, for example, one unit can be prevented from communicating with another if a particular bit is set in an SU register.

Briefly, the Green System is comprised of CLC units which are on line for the use of the real-time tactical programs, and the Amber System is made up of standby units. Units may be shifted from system to system. For example, a problem may develop in a program store unit in the Green System (and the SU will be immediately informed about the problem). The PS unit will be placed in the Amber System, replaced by a unit that works, and checked out while in the Amber partition.

---

[6] It is possible to partition a unit out of both systems, although this is not likely to happen very often.

This is not to imply, however, that the Amber System is made up solely of diseased units requiring medical attention. Rather, the Amber System contains operational CLC units that can perform useful software- and hardware-related functions, including exercising the Green System by simulating a missile attack.

2.6 SUMMARY

The CLC is, in effect, a CPU (plus its memories) broken into separate units that can communicate with each other. The "master" units (i.e., those that can be programmed to execute instructions) are the PU and the IOC; the remainder of the units are "slaves" which must wait for commands or control signals before they can do anything.

Each unit has its own unique functions. For instance, the TG serves as an alarm clock, and the SU records changes in status. Analogously, the whole, the CLC itself, serves a unique function in the SAFEGUARD system: supporting the "outside world" (the peripheral subsystems) to which the CLC is connected by cables and wires.

Chapter 3

COMMUNICATION

3.1 INTRODUCTION

The purpose of this very brief chapter is to give the systems programmer a feel for how communication is accomplished. Included is information on communication between CLC units, communication with the "outside world", and software communication.

3.2 COMMUNICATION BETWEEN CLC UNITS

The SAFEGUARD CLC is designed so that its units can "talk" to each other (and to the peripheral subsystems). There has to be a means, for example, for the PU to access the TG's time registers. And there must be a way for the IOC to tell the STU to start filling up PS with instructions.

The means for communication is via a series of cables (which are, in effect, data channels). But these cables are not "plugged in" directly from unit to unit. Instead, they are connected to intermediary pieces of hardware, called interface switching units (ISU's).[1] Each ISU can accept a series of requests for service and line them up internally (in other words, it can queue the requests for service).

---

[1] It is to be noted that ISU's cannot be programmed. However, although most of the CLC's hardware that cannot be programmed will be presented from a "black box" point of view in this series of manuals, a rudimentary understanding of cable connections is essential.

The data cables, then, are "plugged into" ISU's, which in turn, are connected to CLC racks (see Figure 4). A physical rack contains one or more CLC units. Normally the systems programmer need not be concerned about this because in most cases one rack is made up of one unit. However, an STU, an SU, and a TG unit are all contained in one physical rack (the T&S rack), even though the units are not related functionally.

There is one ISU per variable store, program store, and IOC rack; there are two per PU and T&S rack. Of the two ISU's connected to a PU rack, one communicates with program store and the other with the rest of the CLC. Of the two ISU's connected to a T&S rack, one serves all three units of the rack and the other functions as an interface between the STU and program store.

3.3 COMMUNICATION WITH THE OUTSIDE WORLD

Just as units of the CLC communicate with each other via cables, the CLC communicates with the outside world via cables. However, while CLC inter-unit cables are data channels connected to ISU's, CLC/peripheral device communication is carried on via I/O channels connected to the IOC[2]. The IOC, then, would "talk" to

---

[2] There is one I/O channel connection within the CLC: between the T&S rack and the IOC (not shown in Figure 4). Like other I/O channels, it is not connected via an ISU.

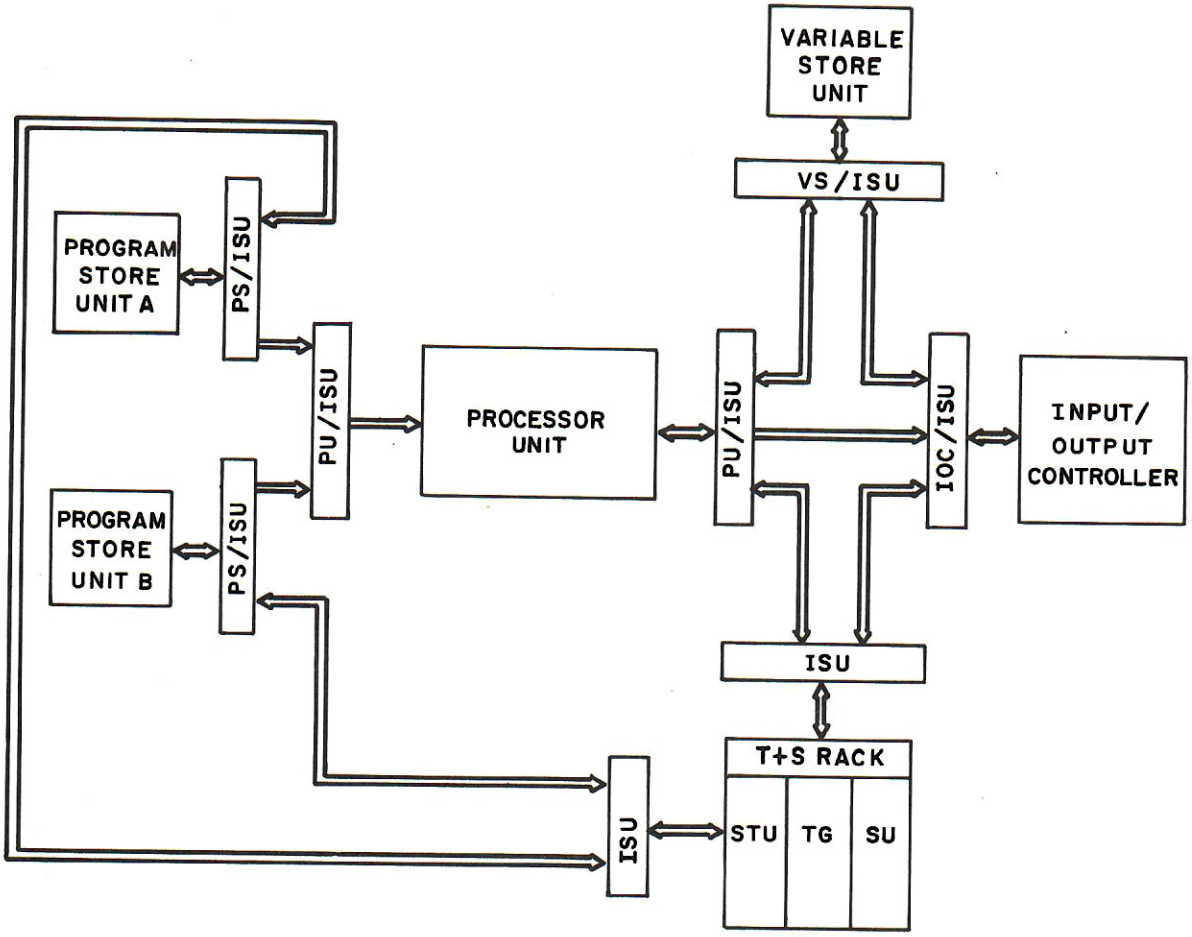


FIGURE 4. SAMPLE CLC UNIT INTERFACES

VS via a data channel, and it would communicate with a disk controller via an I/O channel.

One additional type of "plug in" exists between the CLC and the outside world: interface connections. These are wires between the internal logic portions of the SU (and the TG) and peripheral devices, as well as between the internal logic portions of the system's two SU's. These wires are built to carry logic levels (either on or off).

3.4 SOFTWARE COMMUNICATION

3.4.1 Definitions

Throughout this series of manuals, a program will be considered one or more related instructions (such as "add" or "shift left"). As was pointed out in Chapter 2, PU instructions are stored in PS, and IOC instructions are stored in VS; both types of instructions (as well as the data needed by those instructions) are obtained from PS and VS via data channels and ISU's (see Figure 4).

Instructions are executed internally by the PU or IOC. However, in addition to executing instructions internally, a PU or IOC may trigger an action in another unit; that "trigger", that software communication across a data channel to another CLC unit, is called a command. For example, the PU may send a command to the IOC to demand

that a certain I/O task be started; in order to comply with the command, the IOC will have to execute a set of instructions internally.

A command, then, is a meaningful string of bits that travels across a data channel and that triggers an action in another CLC unit. If, however, the string of bits travels across an I/O channel from either a peripheral device or the T&S rack to the IOC, it is called a peripheral command. Or if an IOC sends a string of bits that triggers an action in a peripheral device, that string of bits is called an order.

The following table summarizes the terms command, peripheral command, and order:

Sender	Recipient	Path	Name
PU	IOC	Data Channel	Command
PU	T&S Rack	Data Channel	Command
IOC	T&S Rack	Data Channel	Command
T&S Rack	IOC	I/O Channel	Peripheral Command
Peripheral	IOC	I/O Channel	Peripheral Command
IOC	Peripheral	I/O Channel	Order

3.4.2 Issuing Commands and Orders

There is nothing particularly complicated about issuing commands and orders from the PU or IOC. As was pointed out in the previous chapter, there are fetch and

store instructions in the repertoires for the PU and IOC. Ordinarily, the programmer might want to address a VS rack (in a store instruction) in order to place some data in VS. However, by changing a couple of bits in the store instruction (that is, by addressing another rack), he can store the "data" (which, in this case, is actually a valid command) in, say, the T&S rack. The command he has "stored" will trigger an action by the appropriate T&S unit.

Chapter 4

PERIPHERAL SUBSYSTEMS

4.1 INTRODUCTION

4.1.1 General

The function of this chapter is to give the systems programmer a very general idea about the world that is outside the CLC and that is connected to the CLC by I/O channels and interface wires (see Section 4.3). This chapter is far from exhaustive; only a sampling of the "outside world" is presented. For additional details, reference should be made to the remainder of the Systems Programmer's Guides and to the many excellent hardware oriented documents on the individual peripheral subsystems.

4.1.2 Definitions

Throughout this series of manuals, the definitions presented in the following paragraphs will be used consistently.

In a number of documents on the SAFEGUARD System, the term subsystem is used. There is a Recording Subsystem (RSS), a Command and Control Display Subsystem (CCDSS), a Maintenance and Diagnostics Subsystem (M&DSS), and so on. Unfortunately, the term will have to be accepted a priori (and will be used sparingly in these manuals). The term is

applied arbitrarily to a group of units that appear to perform a common function. For instance, all the units of the Recording Subsystem (e.g., tape and disk) perform a recording function (although it may be difficult to think of a card reader, which is part of the RSS, as a recording device--and it may be difficult to discover why teletypewriters are units in another subsystem).

The more meaningful terms pertinent to the "outside world" (that is, the peripheral subsystems) are peripheral device and peripheral unit. A peripheral device is one "black box" control unit (plugged into the IOC via an I/O channel) and all the units it controls; Figure 5 illustrates this concept.[1] A peripheral unit is one unit hooked up to a controller.

Thus, a peripheral subsystem may be made up of one or more peripheral devices. A peripheral device is comprised of a controller and its associated peripheral units.

Extending the concept depicted in Figure 5, the CLC and several of the subsystems it supports are illustrated in Figure 6.

---

[1] However, note that Figure 5 does not represent the structure of all subsystems - although it does represent most of them.

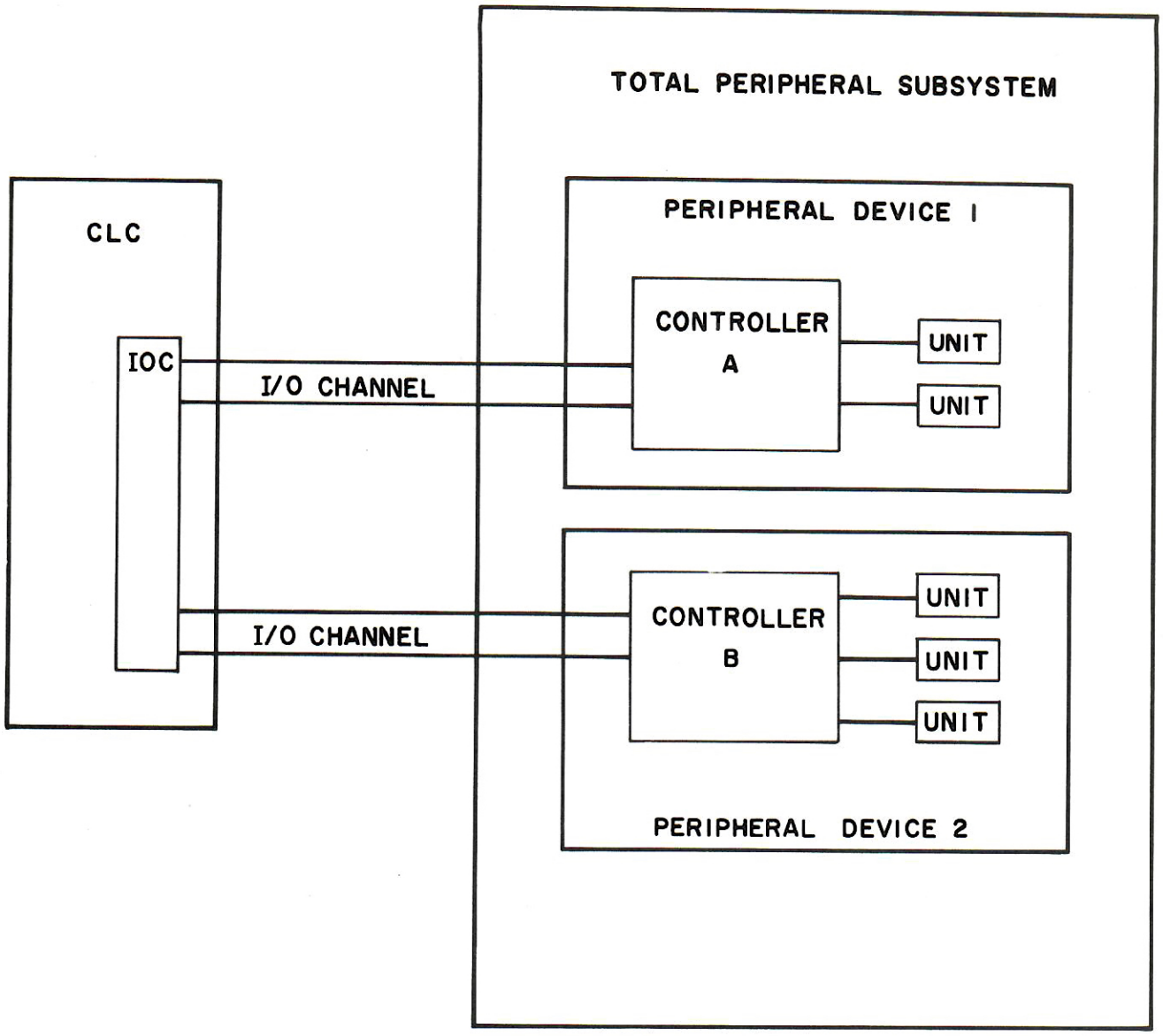


FIGURE 5. DEFINITIONS OF "PERIPHERAL DEVICE" AND "PERIPHERAL UNIT"

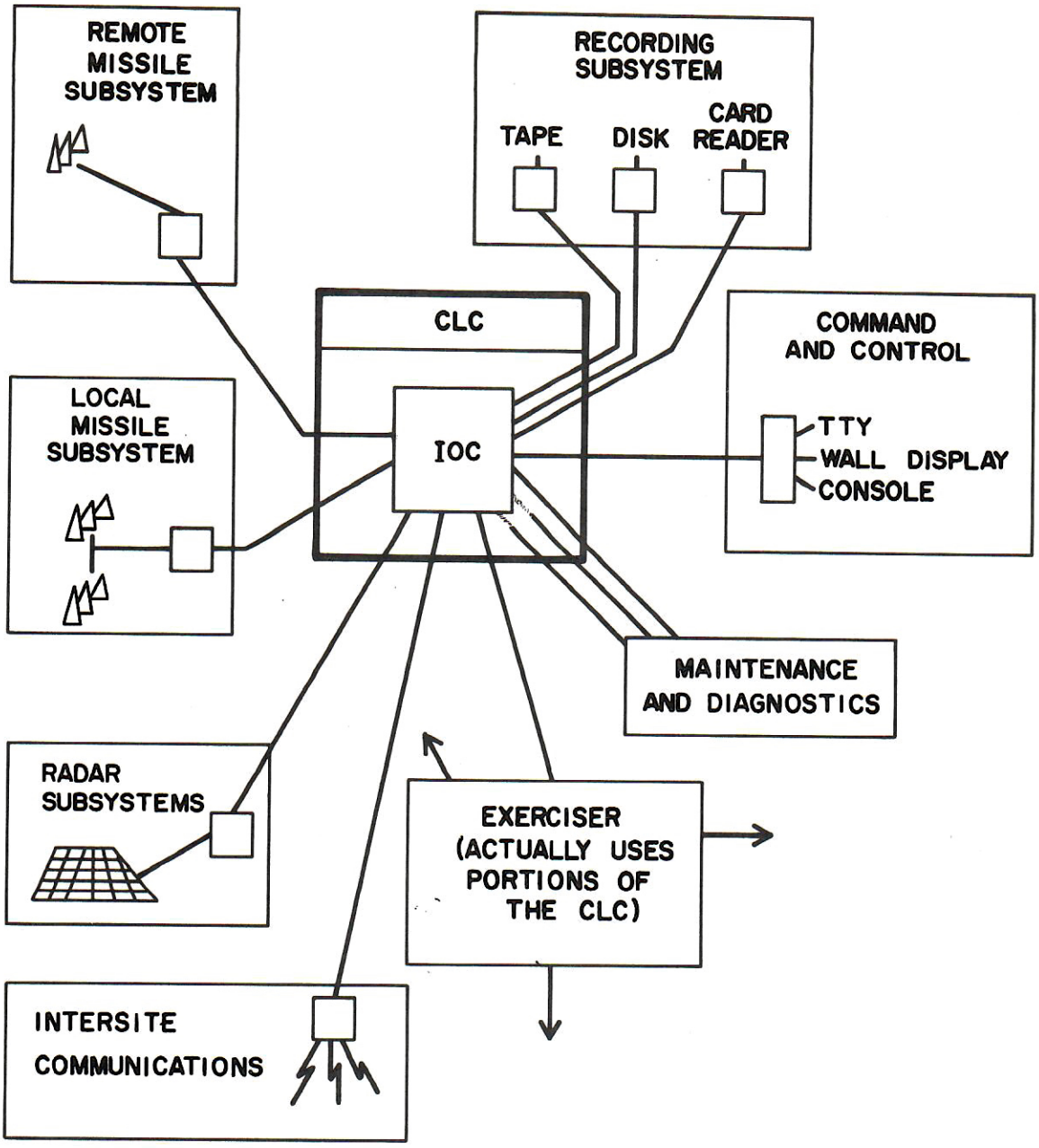


FIGURE 6. GENERAL STRUCTURE OF SAFEGUARD SUBSYSTEMS

4.1.3 Subsystems Covered

Several peripheral subsystems are introduced in this chapter (see Figure 6):

1. Recording Subsystem
2. Command and Control
3. Maintenance and Diagnostics
4. Radar Subsystems
5. Intersite Communications
6. Local Missile Subsystem
7. Remote Missile Subsystem
8. Exerciser

In each case it is to be borne in mind that some portion of each subsystem is connected by an I/O channel to the CLC.

4.2 RECORDING SUBSYSTEM

4.2.1 Controllers and Units

A Recording Subsystem (RSS) is included as part of the system for two reasons:

1. It is needed to get information in and out of the CLC (for instance, there is no capability in the CLC for printing the results of a calculation).

2. It is needed because the storage capability within the CLC is far from infinite.

The RSS is composed of three types[2] of controllers (each of which is plugged into an IOC):

1. Magnetic tape transport controller (MTTC)
2. Multiple disk drive controller (MDDC)
3. Multiplex controller (MXC)

The MTTC controls four magnetic tape drives; tape is considered an intermediate speed unit. The MDDC controls four disk drives (or two disk drive cabinets); disk is considered a high-speed unit capable of sending or recording large quantities of data. And the MXC controls two printers and one card reader, each of which is a slow unit. The relationship between the controllers and their units is shown in Figure 7.

Each controller is "smart" enough so that it can direct its units to perform different kinds of jobs simultaneously. For example, of the tapes connected to an MTTC, one might be rewinding while another is sending its data to the CLC.

4.2.2 Uses by the CLC

The uses of the Recording Subsystem by the CLC are these:

1. Programs (i.e., PU or IOC instructions) can be picked up by the IOC from RSS devices and stored in VS or can be shipped to the STU for storage in PS.

---

[2] There will be more than one of each type available to the system.

2. Data can be picked up by the IOC from the RSS devices and stored in VS, or, conversely, data (such as the result of a PU calculation) can be picked up from VS and sent to the RSS.

3. Copies of programs can be sent from PS to the STU to an RSS device (via the IOC).

4.3 COMMAND AND CONTROL

4.3.1 General

The function of the Command and Control Display Subsystem (CCDSS) is to permit operating personnel to communicate with the system. For example, an operator may wish to "talk" to the CLC by using a teletypewriter (TTY); it is the CCDSS which helps him accomplish this task. The units which make up the CCDSS include (but are not limited to) TTY's and cathode ray tube (CRT) display units. Like the RSS, CCDSS units are wired to a controller, which, in turn, is connected to the IOC and the status unit.

4.3.2 CCDSS Equipment

The CCDSS "controller" is called the logic control buffer (LCB).[3] Because of differences in data transfer speeds between the IOC and CCDSS teletype units, the LCB holds material in a buffer memory before shipping it to

---

[3] Actually, there are two independent LCB's in a single rack and each is connected to a different IOC. One IOC/LCB combination is in the Green System, and the second is held in the Orange System as backup for the tactical LCB.

CCDSS units. The LCB, then, controls the transfer of information between its internal buffer memory and each unit.

These are the CCDSS units:

1. Teletyprinters (TTY's) - provide two-way communication between an operator and the CLC.
2. Consoles in any of three configurations:
   a. Cathode ray tube (CRT) display unit (and light pen).
   b. CRT and control-indicator unit (the control-indicator unit provides switch indicators for the console operator in addition to those related to the CRT).
   c. Non-CRT consoles (contain switch indicators only).
3. Wall displays - provide system status displays.

4.3.3 Interfaces With the CLC

Each LCB is connected to an IOC by an I/O channel. Orders (such as "clear buffer memory") are passed to the CCDSS via the IOC; data transfers are made in a similar fashion. The CCDSS can send data to the IOC, or it may transmit a peripheral command (such as "I want to talk to you") to the IOC. In addition, an LCB may communicate with the status unit whenever there is an error or a change in status.

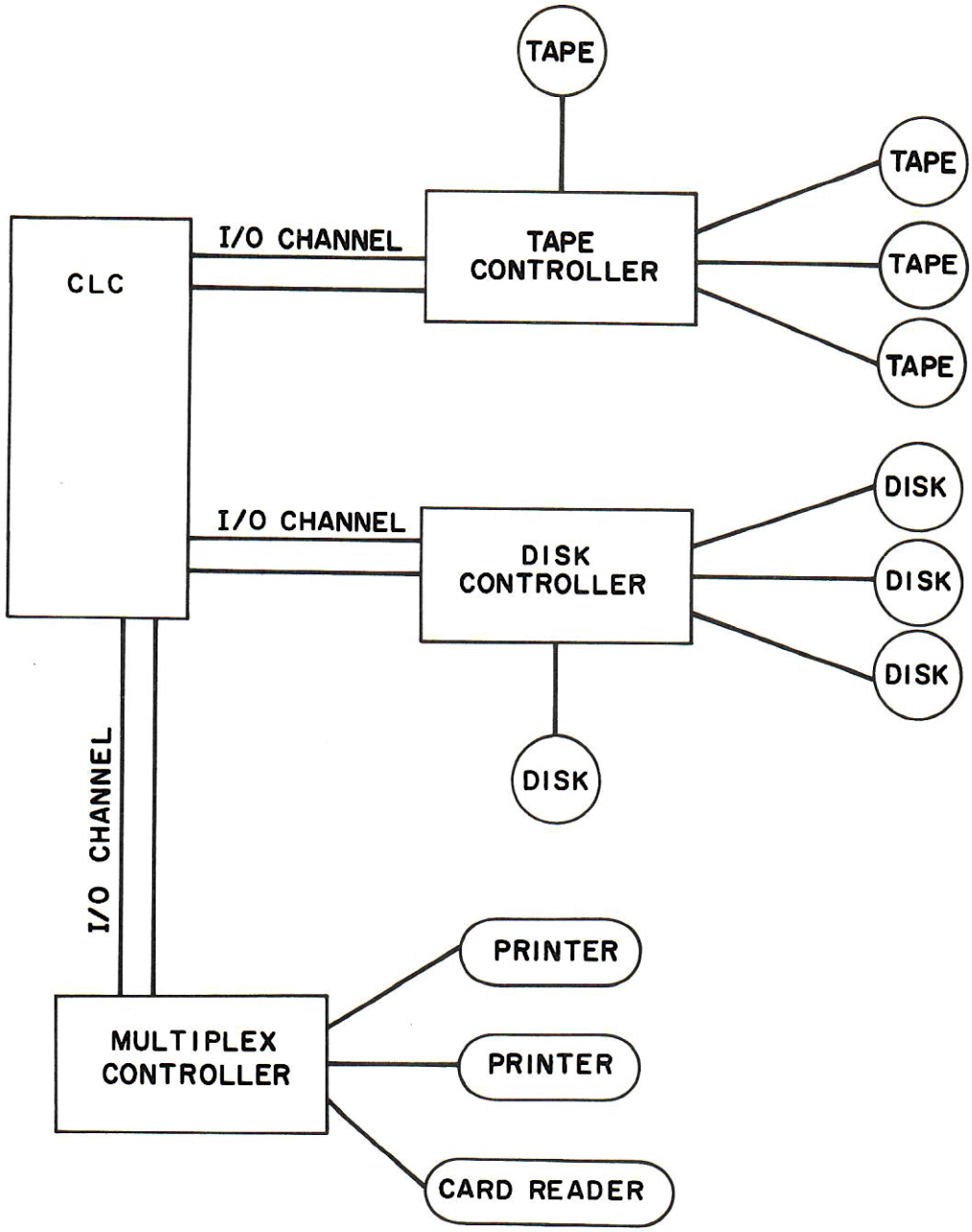


FIGURE 7. RECORDING SUBSYSTEM

4.4 MAINTENANCE AND DIAGNOSTICS

When a piece of CLC hardware fails, bits in the status unit are set to indicate the condition. However, the faulty piece of CLC hardware must be isolated from the rest of the system and replaced by a backup unit, and the unit must be tested and checked for probably error patterns until the problem is located. It is the function of Maintenance and Diagnostics (M&D) to accomplish that checkout on equipment that has been isolated.

M&D, which has access to all portions of the CLC, is comprised of its own computer (a CDC 1700), an M&D Console for manual control of testing, and a variety of associated equipment, such as tape, disk, and TTY. The functions of all these pieces of hardware are these:

1. Initiating and loading the CLC (and reinitializing--for instance, after a hardware unit in the Green System fails and has to be replaced by one that works).

2. Assisting in DPS recovery (for example, the CLC's operating system might find that there have been too many hardware errors, in which case the operating system will signal that DPS recovery should begin).

3. Testing and diagnosing the hardware (some of the diagnostics are automatic, while others are entered manually). These tests may be conducted after a fault occurs in the hardware or as a matter of routine checking.

4.5 BRIEF PAUSE

Up to this point, the "CPU" for the SAFEGUARD System has been described from the point of view of its individual components and how they communicate with each other. Further, CLC communication with the "outside world" via I/O channels and interface wires has been touched upon. Finally, three portions of that "outside world" have been mentioned, and it is the function of those three subsystems to:

1. Extend the storage capacity of the CLC and get information in and out of the CLC.
2. Provide man/machine communication with the CLC.
3. Diagnose hardware problems.

The remainder of this chapter will briefly describe the subsystems comprising the raison d'etre for all the hardware and communication paths: the radar and missile farm subsystems, as well as a subsystem that exercises the system under simulated battle conditions.

4.6 RADAR SUBSYSTEMS

4.6.1 Sites and Equipment at the Sites

There are three types of "sites" in the SAFEGUARD System:[4]

---

[4] This is aside from the initial proving ground and continuous maintenance site for the system's software, the TSCS.

1. Ballistic Missile Defense Center (BMDC)
2. Perimeter Acquisition Radar (PAR)
3. Missile Site Radar (MSR)

The BMDC is an interface between the system and the military personnel who must make the decisions about the use of SAFEGUARD capabilities; the BMDC site contains Data Processing System (DPS) equipment but no radar or missiles. PAR sites were developed to perform long-range surveillance and initial detection and tracking of incoming missiles; PAR sites contain DPS equipment and a type of radar (the PAR) but no missiles. MSR sites are used to track the incoming threats and to guide the defensive missiles; MSR sites contain the whole bag: DPS equipment, a type of radar (the MSR), and defensive missiles (SPARTAN for long range and SPRINT for short range).

4.6.2 Radar/CLC Relationship

To compute the angular position of a target, the range of a target, etc., the radar (whether PAR or MSR) must rely upon the support of the CLC, which performs the actual computation. Further, it is the CLC which must direct the radar to perform tracking or surveillance, determine what time the radar is to perform a particular task, and decide what angle the radar beam should be pointed toward.

As an interface device between the radar and the CLC there is a "controller" (connected via an I/O channel) which acts as a central clearing house for data passed back and forth. The "controller" for the PAR or MSR receives data from the CLC and distributes that data to its units, where a digital to analog conversion eventually takes place.

4.7 INTERSITE COMMUNICATIONS

Since it is required that there be communication between sites (between, for example, an MSR site and a PAR site), the Data Transmission Controller (DTC) has been developed. The function of this hardware group is to transmit data to and from sites. The DTC interfaces with the CLC via an I/O channel connected to the IOC and interface wires "plugged into" the T&S rack.

4.8 LOCAL AND REMOTE MISSILE SUBSYSTEMS

A "local" missile subsystem serves as a pre-launch communications link between the CLC and the launch preparation equipment at a missile farm. Equipment in this subsystem receives two launch orders: The first causes it to provide the proper voltage to the launch preparation equipment to start the launch sequence, and the second causes it to transmit information to the launch preparation equipment to complete the sequence.

A "remote" missile subsystem performs a similar function, but it is used for remotely located missile farms (that is, those located 10-25 miles from the Missile Site Control Building). Whether "local" or "remote", the equipment interfaces with the IOC and the T&S rack.

4.9 EXERCISER

4.9.1 General

The Exercise Subsystem can check out the system under simulated tactical conditions. A threat trajectory file is available to the subsystem, and this file contains a description of target ranges, velocities, etc., for use in tactical system simulation. In addition, the subsystem can simulate missile status and intersite communications traffic.

4.9.2 Equipment

In the Exerciser, the main piece of equipment differing from hardware already described is the Exercise Control Unit (ECU). Other than that, equipment already in existence is utilized. The ECU, for instance, is connected via I/O channels to IOC's and via interface wires to T&S racks. The IOC within the Exercise Subsystem (the Orange System IOC) can transmit simulated missile status information and orders for the ECU and can receive data from the ECU. In addition, the ECU interfaces with several other SAFEGUARD units and subsystems. For example, it can:

1. Connect portions of intersite communications in order to assist in simulating normal traffic to the tactical system.

2. Monitor transmit and receive traffic between the tactical system and the radar "controllers."

3. Intercept missile orders going to remote missile subsystems.

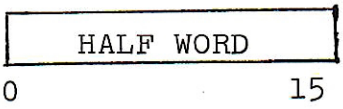
APPENDICES

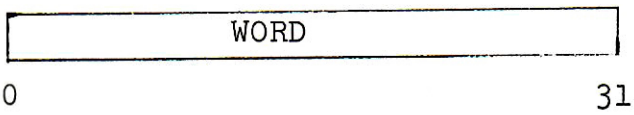
The miscellaneous collection of appendices on the following pages is intended to assist the systems programmer as he utilizes the succeeding volumes in this series of Systems Programmer's Guides. The appendices are presented in no particular order and may be added to from time to time. Such common items as binary-to-decimal conversion tables are not included here because they are available in plentiful supply in other manuals.

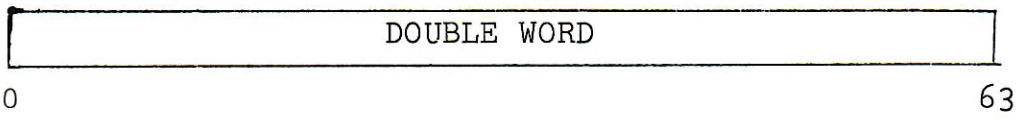
Appendix A

PROCESSOR UNIT DATA FORMATS

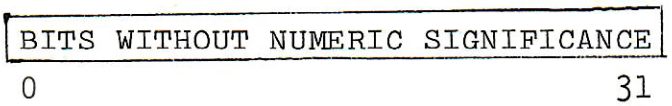
Each word operated upon by the CLC's processor unit consists of 32 bits, and each bit is identified by the numbers 0 through 31. The bits that make up a double word are identified by the numbers 0 through 63, and those that comprise a half word are identified by the numbers 0-15. Thus:



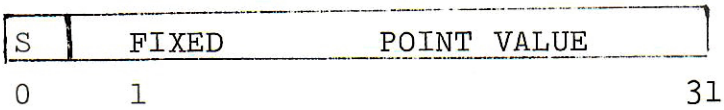




Each word may contain data in one of these formats: logical, fixed point, or floating point. Logical data is structured like this:

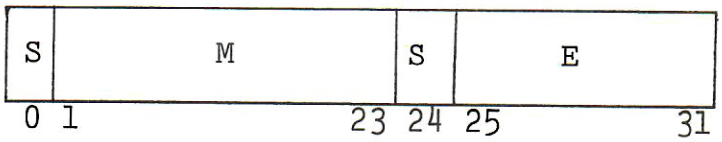


Fixed point data is structured like this:



If the sign bit (S) is zero, the value is positive, and if it is one, the value is negative. Negative values are stored in two's-complement form. (Note that the machine assumes that the binary point is to the left of the fixed point value but that the programmer can apply his own "scaling factor" - even to "pretending" that the binary point appears after bit 31. However, it is recommended that the programmer keep track of his "scaling factor," particularly for fixed point multiplication, division, and square root operations.)

Floating point data is structured like this:



where:

S is a sign bit (0=positive and 1=negative; negative values are stored in two's-complement form).

M is the mantissa portion

E is the exponent portion.

To perform a floating point operation, the programmer must normalize the mantissa portion so that bits 0 and 1 are not equal. Note that if the mantissa portion contains all zeros, the data cannot be normalized, since bits 0 and 1 will never be unequal.

Appendix B

CLC REGISTERS

Following are the CLC registers available to the programmer. The uses of these registers can be found in Volume 2 of this series of manuals.

PROCESSOR UNIT

1. 16 32-bit A-registers (A0-A15). A0 is an accumulator which receives the results of most arithmetic operations.
2. 1 32-bit R-register. The R-register is a 32-bit extension of the A0-register.
3. 16 32-bit B-registers (B0-B15).
4. 4 4-bit C-registers (C0-C3).
5. 16 32-bit Z-registers (Z0-Z15).

IOC

1. 32 64-bit CWS registers (identified by channel/port number-see Volume 2).
2. 2 64-bit History Registers (History Register 1 and History Register 2).
3. 1 32-bit Lock History Register.
4. 1 64-bit C2 Register.
5. 4 32-bit Snapshot Registers

STORE TRANSFER UNIT

1. 1 64-bit Read Register
2. 1 64-bit Write Register
3. 1 64-bit Control Register

TIMING GENERATOR

1. 1 64-bit TOD Register
2. 1 64-bit BCD Clock
3. 1 64-bit Timer Save Register
4. 1 64-bit Counter Reset Register
5. 2 8-bit Compare Masks
6. 2 8-bit Late Request Counter Masks
7. 2 5-bit PRP Masks

STATUS UNIT

1. 108 32-bit Status Words.
2. 12 32-bit Toggle Words.
3. 4 32-bit Partition Registers.
4. 4 32-bit Isolation Registers.
5. 6 18-bit Input Lock Registers.
6. 6 18-bit Output Lock Registers
7. 6 18-bit Notification Lock Registers

Appendix C

GLOSSSARY

Most of the terms used throughout the Systems Programmer's Guides are defined in the Software Glossary, DDXGLOSS@SMOOXXXX, and others can be understood by the context in which they appear. However, there are several terms used in a very specialized sense in the Systems Programmer's Guides (and in some cases, their meanings are unique to the Guides). These terms are defined below.

ADDRESS PORTION	Bits 12-31 of a word; used in reference to some PU instructions.
CHAIN	A collection of list or queue links in VS connected via pointers within each link. Chains are used by the IOC.
COMMAND	A string of bits sent from a PU or IOC across a data channel to another CLC unit; the string of bits triggers an action in the receiving unit. See also Peripheral Command.
DATA CHANNEL	An interface cable connected between ISU's of CLC racks, used for transmitting commands and data.

DOUBLE WORD	Two concatenated words, i.e., 64 bits (plus parity bits).
EXPONENT PORTION	Bits 24-31 of a word in floating point format.
HALF WORD	Sixteen bits.
INCREMENT PORTION	Bits 0-11 of a word, used in reference to some PU instructions.
INSTRUCTION	A string of bits that can be executed by a PU or IOC. See also Program.
INTERFACE CONNECTION	A wire between a portion of the status unit or timing generator and a peripheral device.
I/O CHANNEL	A cable connection between an IOC and a peripheral device, or between the IOC and a T&S rack, used to carry data and peripheral commands.
I/O JOB	A channel/port oriented IOC program which may transfer data and orders between a peripheral device and VS.
JOB	An IOC program and its associated control information.
LINK	An entry in a chain. See Chain.
MANTISSA PORTION	Bits 0-23 of a word in floating point format.

ORDER	A string of bits transmitted via the IOC across an I/O channel to a peripheral device, intended to generate an action by the peri- peripheral device or a T&S rack across an I/O channel to an IOC, intended to generate an action by the IOC. See also Command.
PERIPHERAL DEVICE	A control device connected to the IOC via an I/O channel and all the peripheral units the device controls.
PERIPHERAL UNIT	One unit (e.g., a card reader) of a peripheral device.
PORT JOB	A channel/port oriented IOC program which should not perform I/O commun- ication with a peripheral device.
PROGRAM	One or more PU or IOC instructions.
RACK	A physical hardware component, comprised of one or more units. See Unit.
SPECIAL IOC JOB	An IOC program which is not channel/ port oriented, initiated via an IIO special command.

UNIT	A logical portion of a CLC rack (e.g., the STU of the T&S rack). See also Rack.
WORD	Thirty-two bits (plus parity bits).