Summer 1970 US /GRBK)

INTRODUCTION TO OFF-LINE COMPUTATION AT NRAO

Dave Ehnebuske

I. The Machine

At NRAO we have an IBM System 360 Model 50. It is a general purpose high-speed stored program digital computer. The system can be viewed as having two basically different and fundamental parts to it. First is the obvious "hardware" or collection of machines, wires, lights and what not that you can see even when the machine is shut down. The second part is of equal importance but is not visible except in its effect. This part is called the "Executive Software". The purpose of the Executive Software is to give the machine the coordination and control necessary to make the hardware do something.

A) Hardware

1) Central Processing Unit (CPU): The Central Processing Unit is the part of the system in which the work is done. The rest of the system merely provides information to the CPU and takes information from the CPU. The CPU is composed of 3 parts: Channels, the Arithmetic and Logical Unit, and Storage.

a) Channels: The channels are that part of the CPU that connects the various input - output (I-0) devices (card reader, punch, printer, tape-drives, etc.) to the Central Processor.

b) Arithmetic and Logical Unit: The Arithmetic and Logical Unit (ALU) is that part of the CPU that performs manipulations on data.

c) Storage: Storage is that part of the CPU that "holds" data and instructions for the ALU. It is a contiguous string of bits (the binary equivalent of the decimal digits). Each group of 8 bits (called a byte) has an "address" or way to find it. The address of a byte is the number of bytes you must skip in order to get to the one you are interested in. Larger groupings of storage are half words, words, and double words. The address of a half word must be evenly divisible by two, that of a word by four and that of a double word by eight.

2) Unit Record Equipment: Unit Record Equipment is the class of input-output devices that handles punched cards or paper. At NRAO we have the following Unit Record Equipment: Printer, Card Reader-Punch, and Plotter.

a) Card Reader-Punch: The Card Reader-Punch is attached to the CPU by one of the channels. It can read cards at 800 cards per minute or punch cards at 200 cards per minute.

b) On the same channel is the Printer. The Printer prints characters line by line at the rate of 1100 lines per minute, 132 characters per line.

c) The last piece of Unit Record Equipment is the Calcomp Plotter. It is a digital incremental drum type and can draw most anything.

3) Direct Access: Direct Access devices are input-output devices which store information in such a way that the order in which information is retrieved is not necessarily related to the order in which it was stored. This is different than say magnetic tape where one must, to get the last piece of information stored, read each piece of information sequentially before it.

On our system we have disk storage for direct access input-output. There are 8 packs of disks on the system. Each pack is 11 disks stacked up like records. Except for the top surface of the top disk and the bottom surface of the bottom disk, both the top and bottom of each of the disks are used to store information. Information is recorded magnetically in concentric circles each of which is called a track. There are 200 tracks on a surface.

4) Magnetic Tape: Tape is the most common medium for storing sequential data in computer readable form. There are several methods for recording tape and we can read and write most of them.

a) $7^{\underline{tk}}$ Seven track tape is, as its name implies, recorded by dividing the tape across its width into seven parts, each part called a track. The data for one character is recorded across its width as seven bits, one bit per track.

b) $9^{\underline{tk}}$ Nine track tapes are much the same as $7^{\underline{tk}}$ ones but instead of dividing the tape into seven tracks, the tape is divided into 9 tracks. Thus one can record a complete byte across the width (8 bits of data and one bit called a parity bit that the system uses to check for errors).

B) Software

1) MFT: MFT for multiprogramming with a fixed number of tasks is the executive system for our computer. It coordinates the work being done with the computer's resources.

2) HASP: HASP for Houston Automatic Spooling Priority System is a companion to MFT. Specifically it handles Unit Record Equipment. HASP and MFT work together to take care of the computer's resources. They read the cards, punch, print, plot, handle storage and CPU, and the many other things to make the computer function efficiently.

II. Support System

Support Software is the set of programs other than Executive Software that allows the completion of many useful tasks. Support Software ranges from programs as simple as reproduction of decks of cards to such complex things as Language Processors (compilers and assemblers).

A) <u>Language Processors</u>

Language Processors are the support software programs that process programs written in a symbolic form that is (more or less) intelligible to people. Language Processors serve the function of translators, translating from symbolic form to machine language which is utter gibbarish. At NRAO we have three Language Processors: Assembler (or BAL for Basic Assembly Language), FORTRAN, and PL/1.

1) Assembly Language: Assembly Language is the closest to machine language of the three languages we support. It is merely machine language directly translated into mnemonic form. For the technically minded, Assembly Language is mnemonic machine instruction processor with relative symbolic addressing and macro-instruction capability. In short, if you don't program well don't try Assembly Language.

2) FORTRAN: FORTRAN was the first machine independent language developed. That is FORTRAN (ideally) does not depend on a particular machine. Despite its antiquity FORTRAN is still used in many applications. It is relatively easy to learn and use. Its best feature is its easy ability to efficiently do complex calculations.

3) PL/1 (for Programming Language/1): PL/1 is a relative newcomer to computing. It has many of the features that made FORTRAN popular but is easier to learn and apply and has a host of abilities that FORTRAN simply does not have.

B) <u>Standard Reduction Programs</u>

There are several programs that routinely reduce the data taken from the telescopes. They have been tested and optimized and work quite well. If you need to use them there are books published describing how they work.

C) <u>Other Libraries</u>

Besides the Language Processors there are other groups (libraries) of programs and pieces (subroutines) of programs:

1) Utilities: The Utilities' library has many mundane but useful programs in it. For instance there is LIST,a program to print on paper what is punched on cards; Repro, which makes copies of card decks; and Sequence, which punches sequence numbers onto card decks. There are also maintenance programs that are used to move, create and delete data and otherwise patch up the system.

2) The Subroutine Libraries: Subroutine libraries are groups of commonly used pieces of programs. These pieces can be hooked up and included in programs to avoid coding the same thing over and over.

a) Scientific Subroutine Packages (SSP): PL/1 & FORTRAN both have subroutine libraries of common types of scientific routines. These may be included in programs.

b) NRAO Subroutine Library: NRAO also has a limited selection of subroutine developed and maintained here. These are mostly astronomy oriented routines and can be very useful.

III. Job Control Language

There is one more piece of software that is very important. It is important because anyone attempting to use the computer system must use it. This piece of software is called Job Control Language (JCL). JCL is the means to outline to the computer what you intend to do, how you intend to do it and with what data the work is to be done. JCL allows you to divide the work into separate logical complete units called jobs. A job is a unit of work which is self-contained. It is the first level in the outline of the work to be done. For example, a job may be to compile (translate) a FORTRAN program and do it.

The second level in the JCL outline of the work to be done is called a step. A job consists of one or more steps. In the preceding example the job would consist of two steps: first compile the FORTRAN program and second do (execute) it.

The third level in the outline defines for the computer what data will be used in each step and where the data may be found. Each step of a job may use one or more of these data definitions.

Each of these three levels has unique JCL statement type. The job definition is done with a JCL JOB card. Each step is defined with a JCL EXEC (for execute which is computer jargon for <u>do</u>). Each step's data definition is done with a JCL DD statement (for data definition).

A) Syntax for JCL

Like other languages JCL has syntactic conventions. Unlike most other languages the syntax for JCL is quite simple. The syntax for JCL is shown below:

```
//name ↑ operator ↑ operand ↑ comment
```

Each of the parts are separated from the others by at least one blank at the position of the arrows. The two slashes must be in the first two positions in the card (cols. 1 & 2) and the comment must end before col. 72. None of the first three fields may contain blanks unless they are enclosed in apostrophes. The name field names the card and is required on most of the cards. The operator is always required and must be one of the following: JOB, EXEC, or DD. The operand is always required. It is composed of parameters which are separated from each other by commas. The comment field is the only field that is always optional. It is also the only field which may have blanks.

B) Types and descriptions

1) JOB card: The JOB card looks like this:

```
//jobname  JOB  (account number), CLASS=x,MSGLEVEL=1
```

Example:

```
//SAMPLE JOB (141,P,1,1,15),CLASS=B,MSGLEVEL=1
```

The accounting numbers are posted in the keypunch room in Charlottesville. The classes for jobs are based on running times and you should check to see what class your job falls into when making a JOB card.

2) The EXEC card: The steps in a job are defined by the EXEC card and in its simplest form it looks like this:

```
//stepname  EXEC  procname
```

Example

```
//PL1   EXEC   PL1LFCG
```

The stepname is optional but the procname is not. The procname tells the computer to search on the disk for a procedure with the name PL1LFCG (which is the procedure for compiling and executing a PL/1 program). Commonly used procedures are given below:

Name	Purpose
ASMFC	assemble a program
ASMFCG	assemble & execute a program
FORTC	compile a FORTRAN program
FORTCG	compile & execute a FORTRAN program
PL1LFC	compile a PL/1 program
PL1LFCG	compile & execute a PL/1 program
ASMFCL ASMFCLG FORTCL FORTCLG PL1LFCL PL1LFCLG	special purpose procedures that are no longer required except for special cases. They all take longer to do than the above.

3) The DD Card: The DD or data definition cards can be unbelievably complicated because they tell the computer where to get the data from and how the program is to use it. But it is nonetheless simple in its concept. The idea is that it has two parts, the first part hooks to the program and the second part hooks to the data:

```
//ddname DD DSNAME=datasetname
```

The ddname is the first part, it is what the program calls the data. The DSNAME= is the second part and the datasetname is the name of the data. This description is not complete for there are many more parts to the operand that must be specified in order for the data to be found (e.g. one must tell on which I-O device the dataset is, whether it is being created now or if it already exists, what to do with the data when the job ends, when the step ends, etc.). By far the simplest DD card is also the most common. It says that the data is in the form of cards and that these cards follow the DD card:

```
//procstepname.ddname  DD *
```

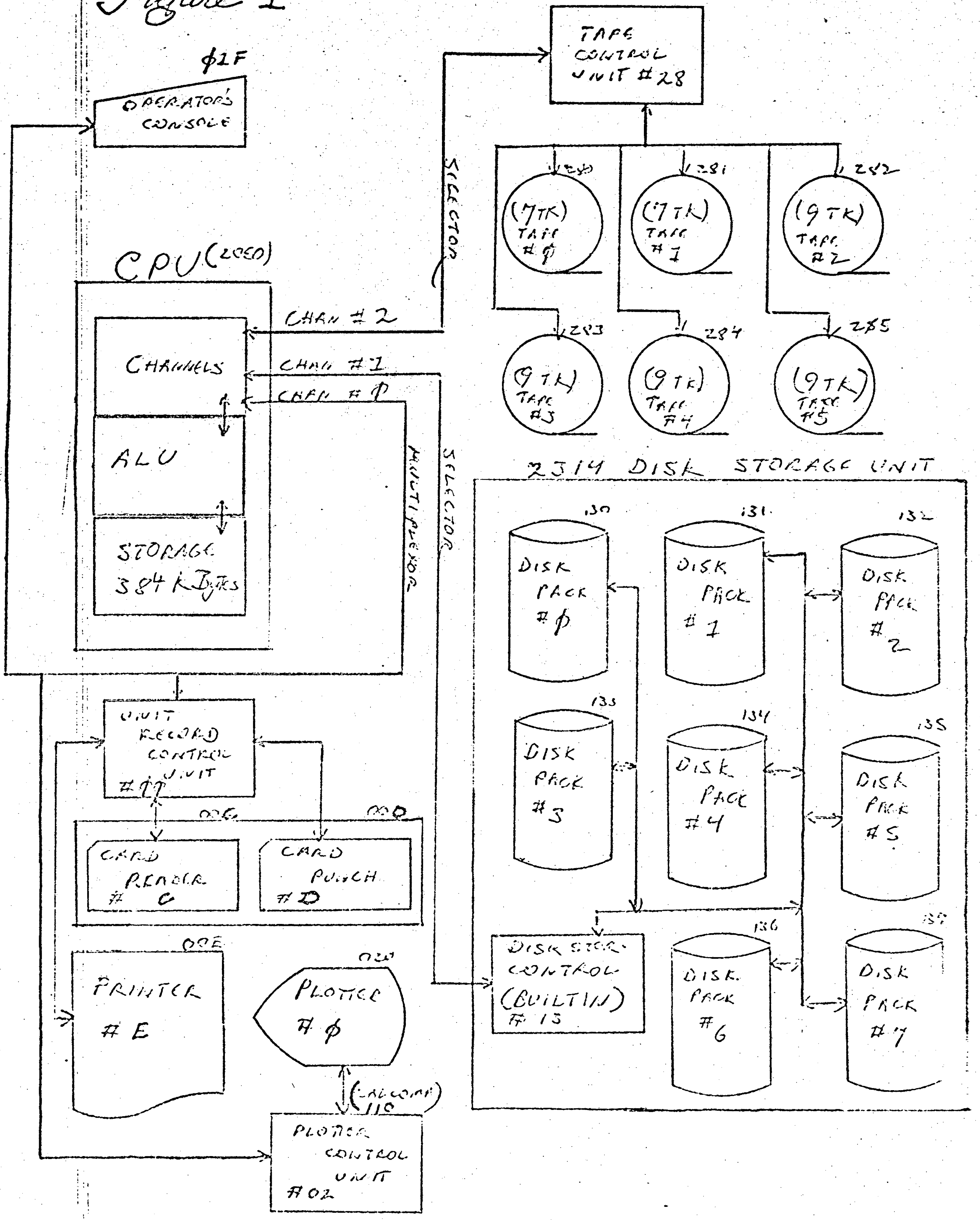
Example:

```
//FORT.SYSIN  DD *
```

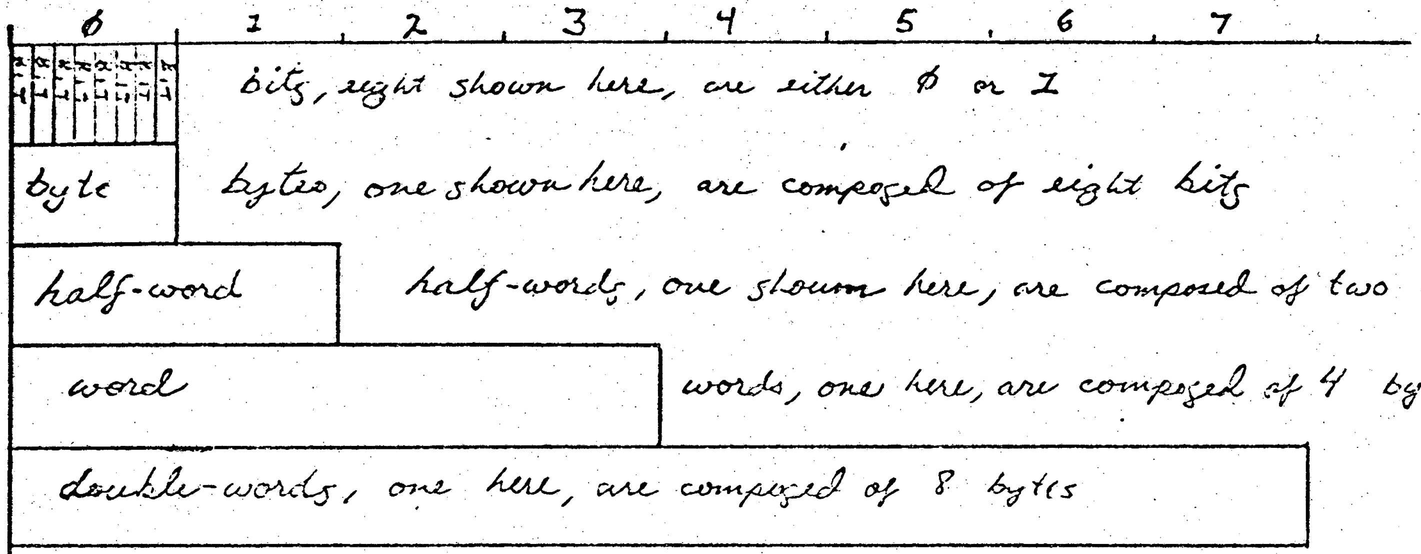
This card says that the cards following are cards for input to the FORTRAN compiler.

NRAO 360/50 HARDWARE CONFIG.

Figure 1



Storage Organization
Figure 2


0
1
2
3
4
5
6
7
bits, eight shown here, are either ∅ or 1
byte
bytes, one shown here, are composed of eight bits
half-word
half-words, one shown here, are composed of two bytes
word
words, one here, are composed of 4 bytes
double-words, one here, are composed of 8 bytes

Alignment:
Because of hardware design, larger storage groupings than bytes are required to start on specific boundaries:
1) the address of a half-word must be evenly divisable by 2.
2) word addresses must be evenly divisable by 4
3) double-word addresses must be evenly divisable by 8.

According to Ken Brady storage has the following additional catagories:

1) Nibble: 4 bits (half a byte)
2) Mouthful: 16 bytes