1963

Built-in and programmed machine controls

Vito Petruzzelli

Built-in and Programmed Machine Controls

by Vito Petruzzelli
Consultant, Chicago Office

Presented before a special meeting of Haskins & Sells Chicago Office audit staff — November 1963

One of the thoughts that occurred to me after I was asked to speak on machine and programmed controls was, "How can I project my interest and enthusiasm to my audience?" Having considered this problem for a while, I thought the easiest way to tackle it would be to allow the subject matter to generate its own interest as it has for me. All of us, whether we care to admit it or not, are fascinated to some degree by the spectacular. Unfortunately, some are carried away by it. This morning we were exposed to the phenomenal capabilities of electronic data processing. It is a fact that the pace at which this technology is advancing is as remarkable as the technology itself. Our education in its elements and its use is a challenging task. The technology of electronic data processing creates an aura of the dramatic and sensational. Our job during this presentation will be to diminish any apprehensions you might have about the subject, solve some of the mysteries concerning how these machines actually function, and, hopefully, develop respect and interest for what these machines and the men and women who use them can accomplish.

The order of our presentation will be:

- A general discussion of controls built into the hardware.
- A general discussion of controls generally provided with most systems through automatic programming.
- A general discussion of controls specially provided in each program.

HARDWARE

The terms *computer, machine, computer* or *machine systems* are used synonymously. For those who are directly concerned with electronic data processing, this rather imprecise method of communicating does not present much of a problem; but for the uninitiated, these terms imply indistinguishably different concepts. To eliminate the possibility of confusion, we will speak about the various *components* of a computer system.

Simply, the functions of a computer system are that it accepts, manipulates, and records data. With each of the verbs in the

preceding sentence, we can associate a specific device. To "accept" data, any system must have an input device; to "manipulate" data, this system must have a processing unit; to "record" data, it must have an output unit.

The most mysterious aspect of these devices, and unfortunately the aspect that lends itself to our greatest concern, is how these functions of taking in, processing, and putting out data are accomplished.

Before explaining how these various devices function, we should review the basic concepts of data recording and representation which were explained earlier. The basic forms of data recording are: cards, magnetic tape and disk, paper tape, charged ferrite cores, and the printed document. The basic forms of data representation are: holes in the cards and paper tape, charged deposits on the oxide surface of the magnetic tape and disk, polarity of charge of the ferrite cores, and a meaningful character on the printed document. The machine controls associated with each of the above-mentioned forms of data recording are similar but not identical. It is important for us to know what these controls are and how they differ.

THE CARD READER AND CARD PUNCH UNITS

The most common input/output devices are the card reader and punch unit.

Most card input/output units are built with two *read* stations; that is, a card is read once and preliminary checks are made before the hole count in each column is stored. These preliminary checks consist of: a test to determine whether a valid character has been sensed by the brushes that have fallen through the card holes onto a charged surface, and a test to determine whether or not a card is properly aligned in the transport mechanism. After the validity check, the hole count is stored and the card passes under a second read station where it is re-read. The validity and hole count checks are performed again and compared to the results of the stored count. If there is any difference, or if an invalid character has been sensed, the device will stop and remedial action can be taken. Once the card has passed both read stations, it is assigned parity based on the specific machine system used. The character code is translated into bit configurations, which can be operated on in the processing unit. The transfer of data to or from the input/output unit is verified by means of parity. An example of parity checking will illustrate the machine

control exercised in data movement. Given are three facts in the example:—One, the data to be moved are the characters *A3;* two, the machine system is based on odd parity; and three, the bit representation within the central processing unit is Binary Coded Decimal. The card reader will recognize *A3* represented in the conventional Hollerith card code, transfer the characters into the Binary Coded Decimal mode, and total the number of bits for each character. The *A* is composed of the B, A, and 1 bits. The number of bits is 3, an odd number, and therefore, no modification of the bit structure is necessary. The 3, however, is composed of the 2 and 1 bits. The number of bits is two; consequently, an additional bit must be added to the character structure. The additional bit is called a check bit. In moving the data from the input device to the processing unit, if a bit were lost for any character, a machine-stop would occur indicating an error in data transfer.

We have gone on at some length to describe parity checking because it is a fundamental control, and one that is a part of almost every phase of electronic data processing.

The card punch employs the same checking features. Of course, the direction is reversed. Instead of the parity and hole count being checked as it goes to the central processor, it is checked as it comes from it. The characters are re-read and the holes counted immediately after the card has passed the punch dies. This is accomplished by a read station positioned after the punch station.

MAGNETIC TAPE UNITS

Magnetic tape units are second to card devices in popularity. These units are commonly referred to as "tape drives" or "tape stands." The outstanding advantages of magnetic tape over cards are speed, reusability, unlimited record format, and file capacity.

The machine controls associated with magnetic tape vary with each manufacturer. The most popular control shared by most is that of vertical parity. A vertical parity check consists of testing for validity the bit configuration of each character on tape. This check takes place in reading and in writing onto magnetic tape. The read-after-write feature is provided by a dual gap read/write head on each drive.

In conjunction with the vertical parity check, there is another control called the horizontal parity check. The horizontal check works like this: As a predetermined number of characters or records, commonly referred to as a block of data, is being written, the processing

unit compiles a total of the bits recorded in each of the seven channels on tape. Based on the parity of the system, even or odd, an additional bit may be placed in that channel. The result is a parity character written at the end of a block of data.

The horizontal and vertical parity checks have been combined by Minneapolis Honeywell to provide automatic error correction. This combination of checks is called "Orthotronic Control." Simply, it means the "downfooting" of each character or "frame" and the crossfooting of each row or channel of bits. Once a vertical parity error occurs in a character a flag or signal is noted. After the block is read the corresponding error in crossfooting should indicate the bit errors in each channel.

One manufacturer has a tape drive that will read a wider than conventional tape. The tape is electronically divided into upper and lower halves. Each half has identical data recorded upon or read from it. This feature is in addition to the parity and validity checks found in other equipment.

Another control built into the hardware is dual-level sensing. This test discloses that the signal strength of a record or group of characters written onto tape is too weak, or that the condition of the tape precludes a successful write operation. The tape is back-spaced and another attempt is made, executing all of the tests performed previously. If the condition of the tape is not suitable after a specific number of attempts, the drive will skip the tape until it has found a suitable recording surface. The number of tape skips is recorded on the exterior label of a tape. As soon as a predetermined number of skips has been recorded, the tape reel is retired.

RANDOM ACCESS UNITS

There is a present trend toward processing data concurrently with the completion of a transaction. This generally requires the transmission of data from a terminal to a central processing unit. This concurrent processing is called "real-time processing." The most striking example of its popularity has been its acceptance as the solution to the problem of airline reservations. Real-time processing is based on the concept of random access. One definition of random access is the system of data processing that provides the availability of any data in a system without searching for the data serially. It is obvious that in order to provide direct access to any data within a system, that system must have a device with massive storage capabilities. Two terms prevalent in most discussions of Mass

Random Access Storage are MARS and RAMAC. Both terms convey the same idea.

These devices employ the same parity and validity checking features existing in the other units. The most distinctive characteristics about these units are the read/write checks and the capability to address any segment of the disk file. Because of the design of the read/write mechanism in most units, it is not possible to read automatically after writing.

This departure from what we have been accustomed to gives rise to the question, "How is the usual read-after-write test provided?" A solution to this problem is not difficult because the position of the read/write mechanism is on the same track on which the data have been recorded. Since there is no movement of the arm immediately after the programmed instruction to write a record, there should follow the command "write check," i.e., re-read the record written on the disk and compare it to the record the program called for to be written. With this device we have become exposed to controls that must be provided by commands rather than by the hardware.

The ability to address any record in the disk file is something requiring the closest supervision. Very often the specific identification of a record such as account or part number is equated to the address of the record within the file. Another method of controlling data handling would be to provide two identification elements. In other words, once the record is accessed, the program would test some independent element within the record other than the address.

PRINTER UNITS

The printing devices incorporate parity and validity checking in the data movement from the central processing unit to the buffer. An additional check called an "echo check" is executed when the position of the print head on a chain, wheel, or type bar is sensed at the moment printing occurs and is compared to the character in storage.

CENTRAL PROCESSING UNITS

The last piece of hardware we shall investigate is the central processing unit. The names associated with this device are "mainframe" and "CPU," the latter standing for Central Processing Unit, and sometimes humorously, "the brain." A major function of the central processing unit is data movement, which includes the storage of different record elements for formating of records and the storage of different values in arithmetic operations. This function is controlled

by parity, discussed earlier. Double or complement arithmetic has been built into the arithmetic circuitry of some central processing units. Interlock circuits inhibit input/output operations while data are being transferred from the unaddressable parts of storage called "buffers."

In order to assure the user that all of these hardware controls are in good working order, most manufacturers provide scheduled preventive maintenance.

AUTOMATIC PROGRAMMING

The second general discussion focuses on the controls to be obtained through automatic programming. By automatic programming we mean the set of computer commands generally provided by the manufacturer to execute certain common operations. Such a set is provided with most languages, enabling the user to communicate with the computer system. A significant automatic programming package from our point of view is the one that controls the input/output functions.

The most important file-handling check that can be performed is the identification of that file. In other words, we should only operate on the files called for in a specific job. We know, for example, that we would not use the "master" payroll-personnel file for an accounts receivable update. Likewise, we would not use last month's loan-payment file for this month's updating of the master loan file. An analogy has been made that writing improperly on a master file is comparable to burning the general ledger. This is not just a truism—especially when the general ledger, itself, is a reel of tape.

Recently there were three examples of this type of disaster that occurred with clients' data. Each analysis of these unfortunate occurrences revealed how carelessness, not malfunctioning, was the cause. The automatic programming check existed but was suppressed or ignored.

How can we be certain that a given reel of tape is the file we want? An external label is one method; however, a reel of tape without an external label is much like a can without a label. The cook may be looking for pumpkin-pie filling but wind up with beef stew instead.

THE HEADER LABEL

Most manufacturers have provided an indelible record at the beginning of a file called a "header label." The main purpose of this record is to identify positively the file as the one sought.

Not only is it the first record on tape, but is is physically separated from the data by a special symbol—a gap of non-recorded oxide surface. This label has the following elements:

- The letters or symbols identifying this as a header label.
- The serial number of this reel of tape. This is the number assigned to this reel as it is received from the vendor and before it is used. For example, 00092 means this is the 92nd tape we have purchased.
- The file serial number, which is the specific identification of the type of file this is. For example, 53729 is the Accounts Receivable Master file.
- The reel sequence number, which is the position of this reel in a multi-reel file. For example, —019 means that this is the 19th reel in this file.
- The name of this file—for example, Payroll.
- The creation date. For example, 63327 is today's date, the 327th day of 1963.
- The retention cycle. For example, —005 means this file must be kept active for five days before it can be written upon.

After this definition of what constitutes a header label, one can easily see that checking this label can provide a high degree of confidence in file handling. There are those who criticize the use of the label-checking feature in automatic programming. Their criticism is based on the fact that it is too time-consuming and thus too costly. Taken in perspective, the relative cost of label checking is extremely low. Consider the effect of the loss of the general ledger file.

It takes about a minute to type the label check, and substantially less if it is printed or punched. This time includes the machine stop and the verification by the operator. The more efficiently run computer facilities are so handled; a client of ours who has a computer system renting for about $400 an hour does so and considers this time well spent. It is difficult, if not impossible, to disagree.

THE TRAILER LABEL

Just as automatic programming provides us with identification information, so too does it provide us with a check to determine that the number of records we had written on a tape was actually the number processed. This is accomplished by means of a trailer label. Like the header label, its format is different from that of the data records and is physically separate from the data also.

It consists of:

- The letters or symbols identifying this as the end of a reel in a multi-reel file, or as the end of the file.
- A count of the number of blocks of information written.
- A count of the number of records written.

  The counts of the number of blocks of data and the number of records are conducted by the automatic programming package independently during program execution, and compared to those written in the trailer label. If there is any difference, a label check occurs. This situation requires an analysis of this program or the program that prepared this file.
- An area that can be used to accumulate independent totals to ensure a valid record count. An example could be adding the first four digits of account number and storing the sum in this area. Such a sum is called a hash total. When this file is used again this hash total can be accumulated independently and compared to the hash total in the trailer label.
- A blank area to be used by the programmer for whatever additional file-handling controls he wishes to impose.

A comment about automatic programming seems appropriate. While such packages cannot supply the degree of control we desire, they nevertheless go a long way in supplying us with some very important tools. We recognize with regret that many installations choose either to ignore or to suppress these controls.

SPECIAL CONTROL BY INDIVIDUAL PROGRAMMING

The third general subject to be discussed is that of control provided in each program. Programmed control such as this can be categorized as linkage, arithmetic, or editing control.

Linkage

The control between programs is generally referred to as linkage. We have already been exposed to header and trailer labels as a means to obtain this control. Additional methods use techniques that are traditional to auditing and accounting in general. They are: balancing dollar amounts and crossfooting; both of these can be easily programmed. The use of test data is perhaps the most effective method of testing linkage. That is, if a deck of test data were to pass successfully through a series of programs called upon to process it by

various methods, one could be reasonably certain that the controls were in effect.

Computational Control

Arithmetic control can be accomplished by several methods. Limit or tolerance testing is a popular technique. One client is preparing a system to audit airline tickets. The system is complex and involves considerably more than just the audit of the sale and use of tickets. It includes the determination of earned revenue, which is a substantial job. The client is relying rather heavily on this technique. Essentially it is this: The price *paid* for a ticket is compared against the price *calculated by the system.* If there is a difference exceeding a predetermined amount stored in the program, an exception record is written. Limit control is used in aging accounts-receivable files and analyzing expense accounts.

Sign checking is another method of arithmetic control. An example of sign checking is: If a given budget account balance becomes negative at any time, write an exception record.

Double, or as one client puts it, double-reverse arithmetic, is a third arithmetic control. This type of control is comparable to what is commonly known as complement arithmetic. An example of this is: Add the multipliers, multiplicands, and products in a series of multiplications and, at the end of the job, divide the multiplier into the product to yield the multiplicand.

These three methods of arithmetic control are representative of a multitude of checks, any of which can be programmed.

Editing

Probably the least understood operations are those generally categorized as editing operations. After listening to the preceding remarks concerning the controls built into the hardware and those included in automatic programming, one might reasonably ask, "I appreciate what you have explained so far, but how can I be assured of the quality of the data?"

We have already heard about test checks and later on we shall hear more about controls that are documented and included in systems design. There are, however, programming checks that are fundamental and invariably found in a well-designed program.

Serial processing has a self-generating control. If a given file is constructed in a given order, there is no reason why that order cannot be tested. This test, called a "sequence test", controls, quite obviously,

out-of-sequence conditions, matching file against file, duplicate records, and missing records.

Very often we find that the execution of a given set of logical commands depends on the codes found in a record. A table of valid codes can be stored so that whenever a code enters a system, it will not only determine a certain logic but will also be checked for validity. For example, the letter "A" in a specific record location is matched against a valid code table. If the code is valid, a given freight rate must be used in the calculation of the transportation charges.

Code checking is extended to include combination or exclusion checks. Using the example mentioned above, not only must there be the letter "A" in a specific record position, but also the order must originate in the State of Illinois. The only limit to this type of checking is a practical one—the size of the program written.

Finally, definition is an extremely important control. By definition here we mean not only a definition of the job that must be done and the definition of the steps to accomplish the job; we mean also what some refer to as the pick-and-shovel work of defining every element in a record. Every item in a record should be immediately familiar to the programmer. Record definition is transcribed into a language understandable to the computer system. Once the transcription is complete, the system will automatically reject any record not presented in the format described.

SUMMARY

Briefly, what have we said about machine and programmed controls?

- Controls exist in the hardware to monitor a given mechanized function.
- In automatic programming, controls exist to assure correct file handling and linkage between programs.
- Individual programs can include multitudes of controls, the degrees and types of which are direct functions of the system being implemented.