XVI SOFTWARE

GENERAL DESCRIPTION

The term software applies to all programs written for a particular computer system and distributed by the manufacturer or a user's group. The programs are thoroughly debugged, can be used over and over with many applications, and therefore save the programmer a great deal of time and effort. Most of the software programs are combined with the user's program so that the operator is not necessarily aware of their use. Some such programs must be physically added to the user's program by the operator—for example, by adding one or more cards (resetters and loaders) to the beginning of a program at the time of card input. When that is necessary, the programmer will specify in his instructions to the operator which programs must be added.

Software is produced by the manufacturer of the computer system, and the various programs are distributed to the customers. Computer users also originate programs which they share with each other. The General Electric Computer Department provides facilities for distribution of such customer-originated programs through what is called the "General Electric Two Twenty Five Computer Users' Association" (GET organization). Descriptions of the GET programs are distributed with the programs themselves, and must be filed for operator reference at each site where the programs are used.

UTILITY ROUTINES

One category of software consists of numerous programs or portions of programs which are available on cards for incorporation into the main program of any application for which they are suitable. They may also be on tape. These programs are called "Utility Routines" or "Service Routines." The majority of them were originally described in the GE-225 Utility Routines Manual. However, current distribution is by individual titles, and they are designated merely as GE-225 Programming Routines. Each routine performs a specific function such as clearing all of memory, changing binary coded decimal (BCD) information to binary, or performing some arithmetic operation. The writeups contain descriptions of each routine—explain its purpose, the method of its incorporation into the main program, and its use.

Most of the information in the writeup is written for the programmer, and the programmer repeats in his run book or other operator instructions, any portions of the description required by the operator. However, the operator should have access to updated GE-225 Programming Routines for reference should he need them. Like many other publications, the descriptions of the routines provide background information which gives a better understanding of what the operator is doing and the reason he is doing it. Since there are occasional changes and/or corrections to both the routines and their writeups, it is necessary that they be kept up to date.

Instead of generalizing about operating procedures for utility routines, procedures for a few of those most frequently used will be described in detail. From the examples, the operator can obtain an understanding of the types of procedures involved. No effort will be made here to summarize or even mention all of the types of routines available.

The operator will keep a set of most-used routines, such as loaders and tape dump cards, neatly labeled and filed in a handy cabinet near the computer console. An extra supply of the most-used cards can be punched ahead of time and kept in the card file. This will save much time when a card becomes worn or bent and will not go through the reader during a run.

Memory Resetters

Most programs used assume that extraneous information remains from previous programs. To make sure memory is cleared out, a utility routine (the program punched on a card) is used to clear each word location in memory, resetting it to zero (zero memory) or all ones (minus memory) or to some known instruction such as "Branch to Location $(40)_8$."

Figure XVI-1 is an illustration of a memory resetter which changes all memory locations to zeros. Operating procedures are the same regardless of how memory is cleared. There are separate routines for the various sizes of memory, e.g., 4K, 8K and 16K.

Following are the operating procedures for "Memory Resetter" routine number CD225B1.0 which sets all memory locations above 40 to either minus one or to zero. The operator will have two separate memory resetter cards in his file of utility routines. One is called "Minus Memory" and the other is called "Zero Memory." The operating procedures are the same for both.

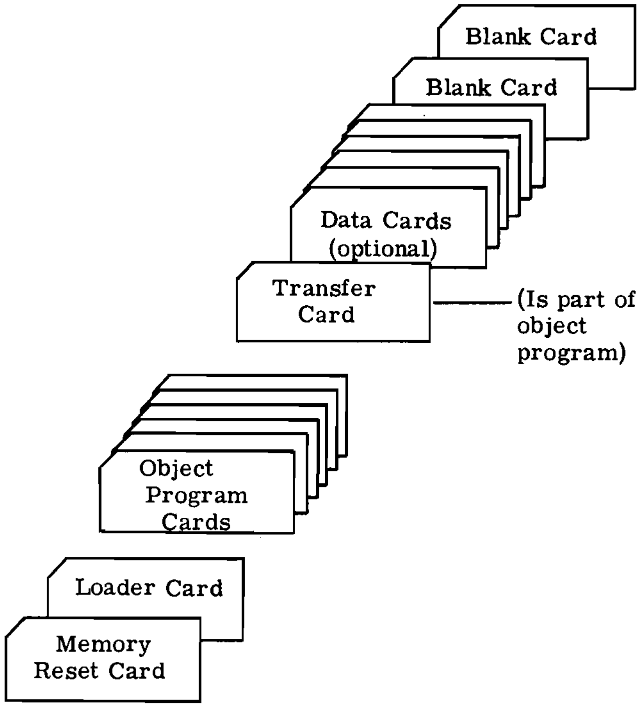
EXAMPLE: Operating Procedures for a Memory Resetter Routine.

SETUP PROCEDURE

1. Prepare the card reader for input.

OPERATING PROCEDURE

1. Set the AUTO/MANUAL switch to MANUAL.
2. Load the card deck, arranged as follows:
3. Depress the LOAD CARD switch to move the first card.
4. Depress the RESET ALARM switch.
5. Depress the LOAD CARD switch to read the first card into memory.
6. Depress the RESET P switch.
7. Return the AUTO/MANUAL switch to AUTO.
8. Depress the START switch to begin automatic feeding of cards.



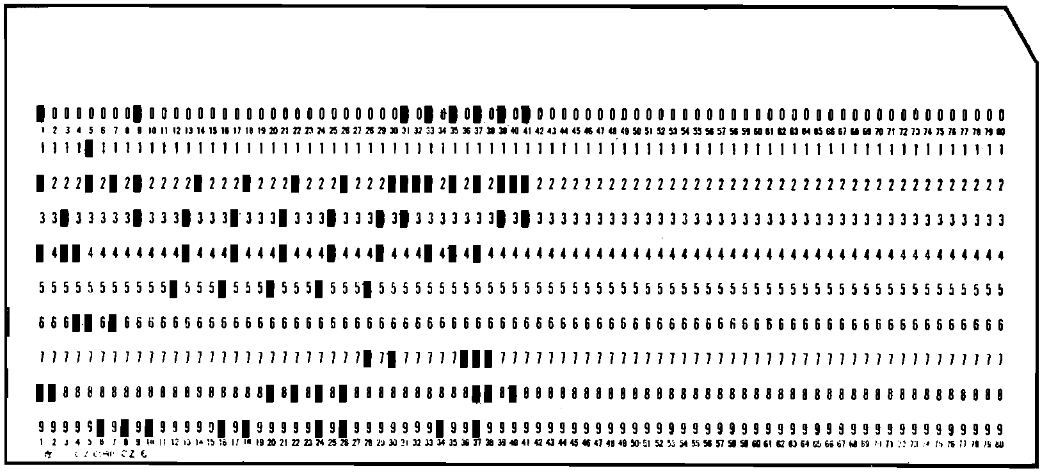


Figure XVI-1. Memory Resetter to "Zero" Memory

Memory Loaders

A loading routine is a short program which, when entered into memory, brings other information (loads it) into memory from cards or tape. Figure XVI-2 is an example of a binary loader for nonrelocatable GAP cards. Each loader must be used only with the type of cards it is designed to load. For example, some programs are punched in relocatable format. (These may be differentiated by noticing that they do not have a zero punch in the first column.) Many language programs, such as GECOM use their own loaders. Unless the various routines specify that they are self-loading, they must be preceded by an appropriate loader.

The following are the operating procedures for a special type of memory loader, number CD225B1.4. It is one which not only loads binary cards into the computer, but computes and compares checksums, and saves information in the first 512 memory locations for a memory dump. Included are the procedures to follow when there is a checksum error. It is not necessary that a memory resetter be used with this loader.

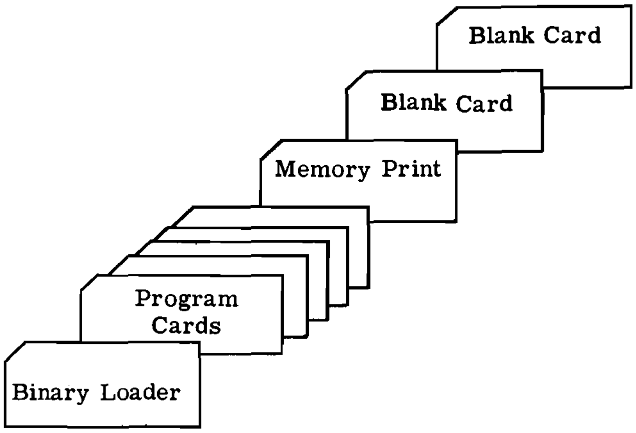
EXAMPLE: Operating Procedures for a Binary Loader with Memory Print Option Routine.

SETUP PROCEDURE

1. Mount tape for output on handler 0, plug 1.
2. Prepare printer for output.
3. Prepare card reader for input.

OPERATION

1. Set the AUTO/MANUAL switch to MANUAL.
2. Load the card deck as follows: (The Binary Loader and Memory Print cards are in the operator's file of utility routines. The program deck is provided by the programmer, and it contains the required transfer card.)



3. Depress the LOAD CARD switch to move the first card.
4. Depress the RESET ALARM switch.
5. Depress the LOAD CARD switch.
6. Depress the RESET P switch.
7. Return the AUTO/MANUAL switch to AUTO.

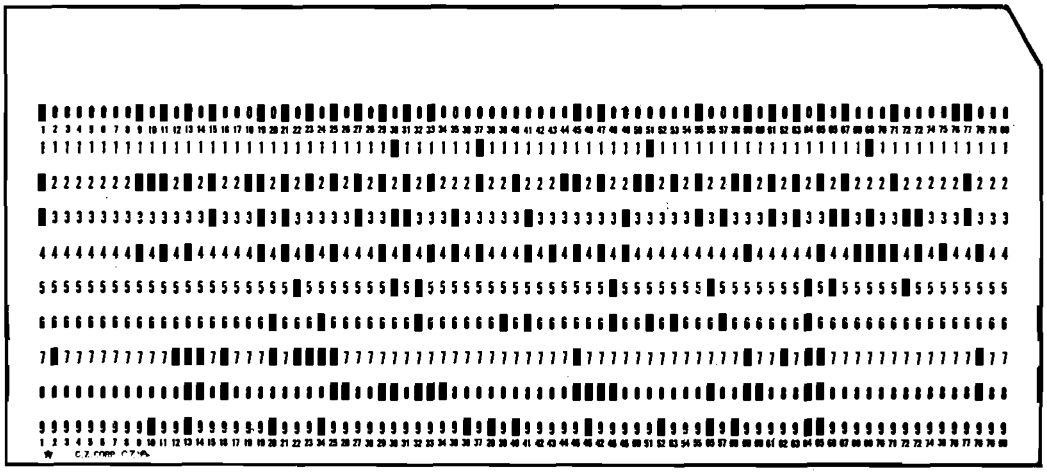


Figure XVI-2. Binary Loader for Nonrelocatable GAP Cards

8. Depress the START switch to begin automatic feeding of cards.

9. When the computer halts after execution of the program, enter $(2600040)_8$ by means of option switches, using the following steps: (This causes the first 512 locations to be written on tape handler 0.)

   a. Set the AUTO/MANUAL switch to MANUAL.

   b. Toggle the RESET A switch.

   c. Raise necessary option switches to enter a octal 2600040.

   d. Depress the A to I switch.

   e. Set the AUTO/MANUAL switch to the AUTO position.

   f. Depress the START switch.

ERROR CONDITIONS

The card reader procedure listed here can be used to correct any type of card reader error.

Indication	Operator Procedure
1. The card reader halts because of a checksum error.	1. Set the AUTO/MANUAL switch to MANUAL.
2. $(37)_8$ appears in the P register of the console.	2. Remove from the output hopper the last card read. (This is the card which caused the checksum error.)
	3. Empty the input hopper.
	4. Return cards to the input hopper with last card read at the bottom of the deck. (Card on the sensing platform must also be returned to its proper position in the deck.)
	5. Depress the LOAD CARD switch to move the first card.
	6. Depress the RESET ALARM switch.
	7. Depress the LOAD CARD switch to read the first card into memory.
	8. Depress the RESET P switch.
	9. Return the AUTO/MANUAL switch to AUTO.
	10. Depress the START switch.
	11. If steps 1 - 11 did not correct the error, continue as follows.
	12. Remove card with checksum error.
	13. Repunch the faulty card by use of the "Checksum Corrector" routine.
	14. Continue with the newely punched card (steps 3-11).
	15. If cards still don't read, the error is in either the program or the equipment. Reproduce the faulty card and give it to the service engineer. Return the program to the programmer with notice of the checksum error.

Memory Dumps

As the word dump implies, a memory dump is a program which produces a representation of what is in memory at a specific time. The dump may be to the typewriter, the printer, to paper tape, or magnetic tape. Many dump routines dump all of memory, beginning at a location immediately after the dump program itself which usually is read into memory starting at location zero. With most dump routines, the operator must manually obtain a printer dump of the first few (0 through about 250) locations of memory, for the routine cannot dump the area it overlays. The manual dump of the lower 250 locations must be made before the dump routine is read in, because the routine itself destroys the information in lower memory. Selective dump routines dump only specified portions of memory. There are separate dump routines for various sizes of memory.

The following are the operating procedures for an 8K memory print routine, number CD225B2.12, which prints the contents of an 8K memory in octal on the high speed printer. The routine is used in conjunction with the binary loader routine just described. The loader saves (on magnetic tape) the information in the first 512 locations.

EXAMPLE: Operating Procedures for an 8K Memory Print Routine

Since this routine actually extends the dump initiated by the "Binary Loader with Memory Print Option", the setup procedure and operation for that routine apply to this one. To continue from operations described in the loader routine, take the following steps:

1. When the program stops in a loop, toggle option switch zero (the sign switch) to read in the "Memory Print" routine.
2. When the program again stops in a loop, enter the printer plug number by use of option switches 17, 18 and 19. For example, if the plug number is 6, lower option switches 17 and 18 to enter a binary 110.
3. Toggle option switch zero.

All of 8K memory will then be printed out on the high speed printer. The program rewinds the tape on handler 0, plug 1. The words END OF MEMORY DUMP will be typed out on the console typewriter to notify the operator when the dump has been completed.

Tape Dumps

Another kind of dump is a transfer of information on magnetic tape to the high-speed printer where it can be easily read, or to paper tape for special uses by the paper tape reader and punch.

The routine "Tape Printer 1", number CD225B1.4, prints a binary or BCD tape in octal or Hollerith format on the high-speed printer. (It will not read special 18-bit binary.)

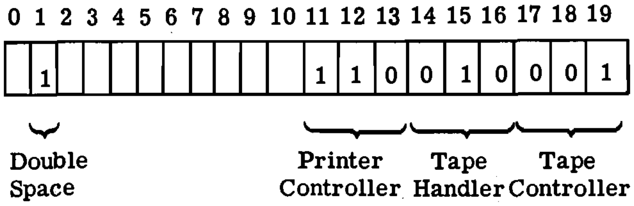
EXAMPLE: Operating Procedures for the Tape to Printer Routine.

SETUP PROCEDURE

1. Mount the tape to be dumped on any handler.
2. Ready the printer for output.
3. Ready the card reader for input.

OPERATING PROCEDURE

1. Load the deck of cards for the routine. (It is a self-loading binary deck.)
2. When the program stops in a loop, enter the printer plug number into the A register positions 11, 12 and 13 by lowering appropriate option switches. Figure XVI-3 illustrates the setting for plug number 6.
3. Enter the tape handler number into positions 14, 15 and 16 by lowering appropriate switches. Figure XVI-3 illustrates the setting for handler number 2.
4. Enter the tape controller plug number into positions 17, 18 and 19 by lowering appropriate switches. Figure XVI-3 illustrates the setting for plug number 1.



0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
	1										1	1	0	0	1	0	0	0	1

Figure XVI-3. Example of Switch Setting, (A 1 means the switch is down.)

5. If the programmer designates a printer format of double spaces between the records, lower option switch 1. (Otherwise, records will be single spaced.)
6. Toggle switch zero to cause the program to continue and the printout to begin.
7. When the tape end-of-file is reached, the program stops with $(15030)_8$ in the P register. The operator may either continue or rewind the tape, as designated by the programmer.
    a. To print past the end-off-file, lower option switch 19 and toggle switch 0.
    b. To discontinue printing and rewind the tape, lower switch 0.

SPECIAL PROCEDURE

The operator can cause the program to skip printing a specified number of records in either a forward or backward direction. The operator can also cause the program to continue printing after an end-of-file halt or cause it to rewind the tape and terminate the program.

1. Space Backward

    If the program should hang in a loop because of an error condition such as a parity error, the operator can make the program backtrack and repeat the part which had the error. The operator may also want to stop the program and make it backtrack because there has been faulty printing – for example, the printer could run out of paper. To do this, the operator must:

    a. Lower option switch 4 to cause the program to stop in a loop after printing the current record. (This step is to be omitted if the program has already stopped.)

b. Enter into the A register positions 7 through 19 the number of records to be skipped. For example, to skip 20 records lower option switches 15 and 17 to read in a binary 20.

c. Lower option switch number 1 to designate a backwards skip.

d. Toggle switch 0 to resume running the program.

2. Space Forward

The operator may want to skip a specified number of records at a particular place in a program. For example, the programmer may have specified that the 40 records after the first programmed halt need not be printed. To skip forward, the operator must:

a. Lower option switch 4 to cause the program to stop in a loop after printing the current record. (This step is to be omitted if the program has already stopped.)

b. Enter into the A register positions 7 through 19 the number of records to be skipped. For example, to skip 40 records, lower option switches 14 and 16.

c. Toggle switch 0 to resume running the program.

Checksum Corrector

Another card program which the operator uses frequently is the checksum corrector which permits the operator to repunch binary cards to correct their checksums. All General Assembly Program (GAP) produced binary cards contain a checksum which is a computed figure based on the number of 1 bits in all columns of the card. This computed figure is punched in the last two columns punched (columns 79 and 80 on full cards).

When loading cards with a binary loader, the checksum is recalculated as each card is read, and compared with the checksum in the last two columns. If an incorrect comparison is obtained, the computer stops in a loop. An incorrect checksum may have been recomputed because the card had either a tear or an oil spot on it, causing the reader to indicate a punch which was not on the cards. It is also possible that the computer failed to calculate properly or the original binary card was mispunched. Incorrect checksums also occur when the programmer makes a correction in a card but neglects to correct the checksum. It is usually necessary that this card be corrected before the program can be continued. After the card is corrected, program operation must be returned to either the beginning or to a known restart point.

Figure XVI-4 is an illustration of a GAP card which has a checksum in the last two columns. The illustration explains how such a card can be read. However, it is not necessary that the operator know how to do this.

The following are the operating procedures for the checksum corrector routine with which the operator can make new cards having corrected checksums. The routine, number CD225B2.8, can be used to correct one card or many cards having checksum errors. It can also be used with a deck of cards having only a few cards with incorrect checksums in the deck. However, the procedure of using the corrector routine on correct as well as incorrect cards is wasteful of computer time.

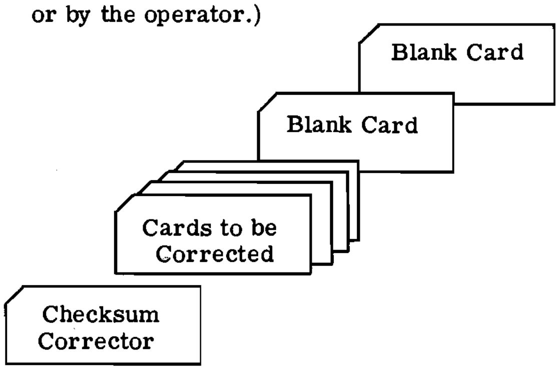
EXAMPLE: Operating Procedures for the Checksum Corrector Routine

SETUP PROCEDURE

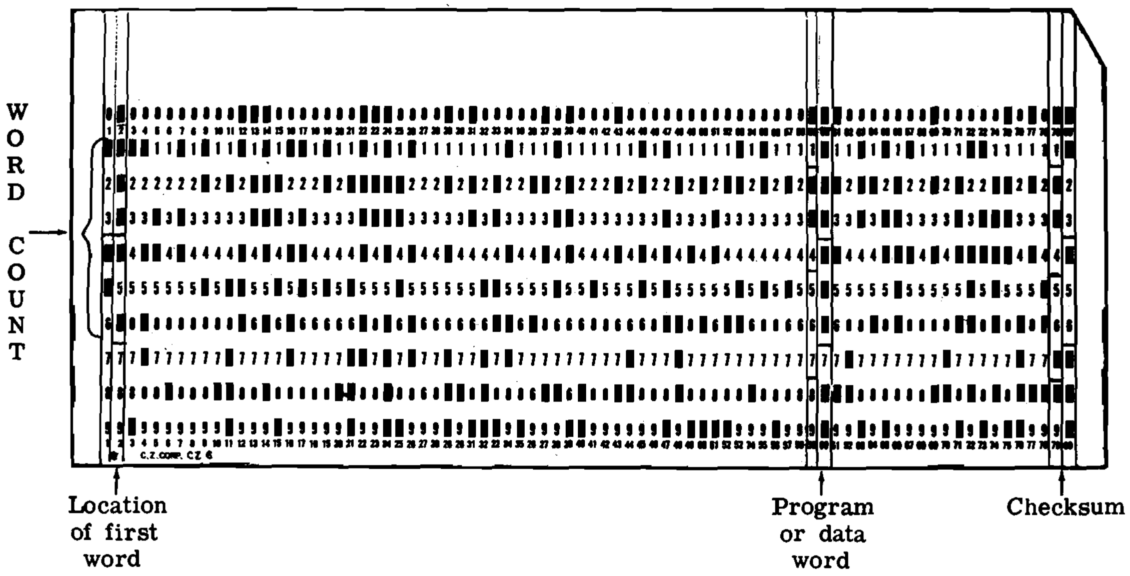
1. Ready the card reader for input.
2. Ready the card punch for output.

OPERATION

1. Set the AUTO/MANUAL switch to MANUAL.
2. Load the card deck as follows: (The Checksum Corrector card is in the operator's file of utility routines. The deck to be corrected may be provided either by the programmer or by the operator.)



3. Depress the LOAD CARD switch to move the first card.
4. Depress the RESET ALARM switch.
5. Depress the LOAD CARD switch to read the first card into memory.



The octal representation of the word count is 46

The octal representation of the location of the first word in memory is 1750

The octal representation of the word in columns 59 and 60 is 2601773 which is the machine language for the instruction to branch unconditionally to location $(1773)_8$

The figure representing the checksum is 2615447

Figure XVI-4. 10-Row Binary GAP Card with Checksum

6. Depress the RESET P switch.
7. Return the AUTO/MANUAL switch to AUTO.
8. Depress the START switch to begin automatic feeding of cards.

Newly-punched cards in the card punch have corrected checksums. These cards should replace the faulty cards in the output hopper of the card reader.

Card Reproducer

The operator frequently uses the routine "Card Reproducer." The need for reproducing cards via the computer arises during card loading when a card is damaged and needs to be replaced quickly to avoid a delay in the running of a program. The need also arises during debugging when one or more card punches is changed to make a correction. (The more economical way to reproduce cards when time is not vitally important is to use regular card reproducing equipment, providing it is available.) The reproducer routine may not be used unless the first $(20)_{10}$ memory locations are available. The operator must make sure that reading the routine into the computer would not destroy required program information.

The "Card Reproducer" routine, number CD225B6.1, can be used to reproduce binary, special binary, or decimal cards.

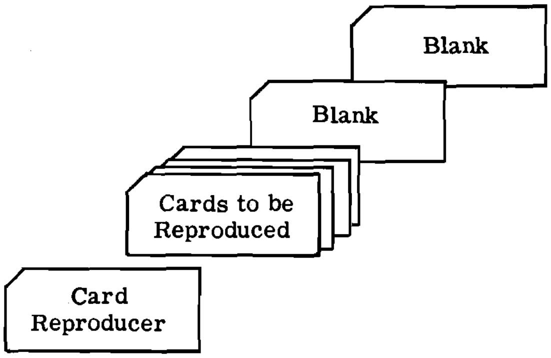
EXAMPLE: Operating Procedures for the Card Reproducer Routine

SETUP PROCEDURE

1. Ready the card reader for card input.
2. Ready the card punch for card output.

OPERATING PROCEDURE

1. Set the AUTO/MANUAL switch to MANUAL.
2. Load the card deck as follows:



3. Depress the LOAD CARD switch to move the first card.

4. Depress the RESET ALARM switch.
5. Depress the LOAD CARD switch to read the first card into memory.
6. Depress the RESET P switch.
7. Return the AUTO/MANUAL switch to AUTO.
8. Depress the START switch to begin automatic feeding of cards.

The newly-punched cards in the card punch must now be substituted for the faulty cards.

GE-225 BRIDGE Service System

As was stated earlier, the majority of the utility routines were originally described in the GE-225 Utility Routines Manual. A few extra-long routines are described in individual publications. One of these is the "BRIDGE Service System." The routine is a tape maintenance and run sequencing program which performs such functions as correcting program tapes and updating or correcting master tapes or transaction tapes.

GE-225 Forward Sort/Merge Generator

This is another of the larger programs which is described in a separate manual. Its sort function arranges records in a sequential order, and is used, for example, in sequencing data files, master files, and transaction records. Its merge function combines several sequential input files into a single sequential output file. Sections of the GE-225 Forward Sort/Merge Generator which particularly apply to operating procedures are:

Section VII which contains general operating instructions

Section VIII which contains information on error detection.

Section IX which contains operating instructions for the forward sort object program.

Section X which contains operating instructions for the forward merge object program.

The Offline Operations Simulator (OOPS)

This is a special type of program which permits simultaneous operation of peripherals under program control. The program is described in the Offline Operations Simulator manual. Section VII of the manual contains information on program operation.

LANGUAGE PROGRAMS

One broad category of software consists of language programs. These facilitate the coding of a problem by permitting the programmer to use a shorter and more simplified form of coding rather than the absolute language of the machine. The program (compiler) translates the programmer's source program into the real machine language, producing what is called the object program. Numerous language programs are available for the GE-225. Most operators will use several, if not all of them. Since the programs are long and relatively complicated, each is explained in its own manual, and each manual has a section on operator instructions which must be referenced, if not learned. The operator must have a manual for each language program he is required to use.

By way of introduction to language programs presently available, they will be described very briefly under their type classifications.

Assemblers

Assemblers permit the programmer to write a program in meaningful symbolic codes, with the number of instructions generally matching machine codes one for one. The "GE-225 General Assembly Program" is the GE-225 program which, at most locations, is used more often than any other. It is usually referred to as GAP. Other language programs use GAP in their translation into absolute machine language. Two manuals on GAP are now available for the operator. They are the GE-225 GAP Operating Instructions and the GE-225 GAP Operating Instructions Using System Tape. The latter manual is for GAP systems which uses either tapes or cards for input.

The binary cards produced by the GAP program can be of two types. One is relocatable cards and the other is absolute.

Compilers

Compilers simplify the work of the programmer by permitting him to write in English language words, creating what is called the source program. The compiler then translates the source program into machine language, creating what is called the object program. The compiler interprets the instructions of the source program, calls in subroutines when necessary, and generates one or more machine instructions for each source language instruction. Several compilers are available for use with the GE-225 System.

The GE-225 General Compiler (GECOM) is widely used, for it is applicable for both business data processing and scientific problems. TABSOL, which stands for a Tabular Systems Oriented Language, is part of GECOM, and applies particularly to the use of decision or structure tables. The Report Writer is also part of GECOM. It permits the GE-225 computer user to rapidly produce reports in a format ready for use. GECOM operating instructions are found in the GE-225 GECOM II Operations Manual. Part of the information in the manual is for programmers, and part is for operators. Of particular interest to operators are:

Section III which contains compiler operating instructions.

Section VI which contains object program operating instructions.

Section VIII which describes use of object program subroutines.

The GE-225 WIZ System is a fast algebraic compiler which is used mainly for engineering and scientific mathematical problems. It translates the source program in one pass and produces machine instructions punched on cards or paper tape. Chapter IX of the GE-225 WIZ System manual contains operator instructions. GE-225 ZOOM is a compiler which permits the programmer to write the sourceprogram in a combination of English statements, algebraic expressions, and assembly mnemonics. Input is on punched cards and output may be on punched cards, magnetic tape, or printer listing. Section IV of the GE-225 ZOOM, a Marco Assembler manual contains operator instructions.

Systems Applications

Another category of software is concerned with solving problems for management. The GE-225 Critical Path Method is a program whichpermitsthe user to feed into the computer information on the scheduling of particular project. The GE-225 System then, by use of the Critical Path Method, produces a network analysis of the most efficient schedules, taking into consideration both cost and time elements. The manual GE-225 Application Critical Path Method Program contains two sections which are of interest to the operator:

Section 4.5 contains program operating characteristics.

Section 4.6 contains information on error stops in the program.

GE-225 Linear Programming permits the user to select the best alternative from a number of possible alternatives. It uses a mathematical technique which specifies how to use certain capacities to obtain the best results. For example, the program might be used to determine the scheduling of machine orders to give the least cost for a specific delivery date. This management tool may be applied to an infinite number of problems. Section IV of the GE-225 Linear Programming Manual contains instructions for the operator.

Industry-Wide Software Applications

Software is available for several industry-wide applications. An operator working for one of these industries could reference operating instructions in the various software writeups for that industry. For example, programs are available for the banking industry in what is called GE-225 Bankpac. Included in Bankpac are programs for the various banking systems such as personal trust, installment loan, demand deposit accounting, and transit accounting. Each run in each system has a run book which contains, among other things, operating instructions for that run.