SCIENCE BULLETIN
CHIAO TUNG UNIVERSITY
Vol. 5, No. 1, APRIL 1971

A New FORTRAN Processing Program-FORTRAN Processor(19000)-for the IBM 1620 Computer.

by

Jong-Chuang Tsay

National Chiao Tung University

Abstract

The existing FORTRAN processor of IBM 1620 digital computer at National Chiao Tung University uses a two-stage translation process to execute a FORTRAN program. Considerable time and paper tape are wasted in this process. This paper describes an one-stage translation process that avoids these disadvantages. This new translation program called FORTRAN processor (19000) is now used in the computer center of National Chiao Tung University.

1. Introduction

The two-stage translation process of a FORTRAN program is accomplished by two programs-compiler and loader & subroutines. The first stage is referred to as compilation and the second stage is referred to as loading. Compilation consists of translation from a FORTRAN source program to an intermediate language, while loading consists of translation from the intermediate language to a machine language object program that is executable in the machine (Fig. 1). The two-stage translation process was used in the computer center of NCTU before July 1970. It is accomplished by two programs called compiler (19000) and loader & subroutines (19000). The procedure for processing FORTRAN programs by the compiler (19000) and the loader & subroutines (19000) is shown in Figs. 2 and 3. In Fig. 2, FORTRAN programs are compiled in batch. When the compiler (19000) is read in by the machine instruction 360000000300, an initialization program in the compiler (19000) is automatically executed. During this initialization phase, the main compiler program is moved in two copies to memory locations 20000-37144 and 00401-17545. A special 10-digit field (00000000‡‡) is also placed in the symbol table area from memory location 59879 to 40499. A single 10-character field, consisting of 9 zeros and a single record mark, is then located at 40489. This special symbol (000000000‡) signifies the end of

the symbol table. The first 12 symbols in the symbol table are reserved for six special function names of subroutines. When the initialization program has been executed, the computer will halt in the manual mode and the following message will be typed:

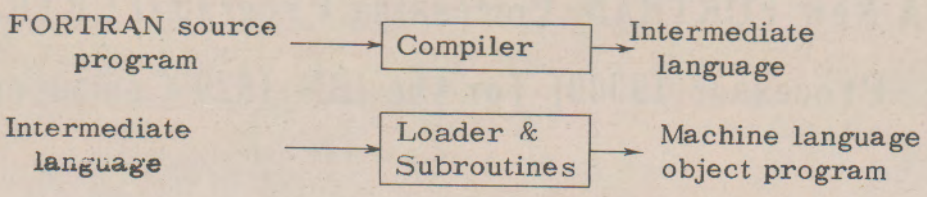

Fig. 1. Two-stage translation process.

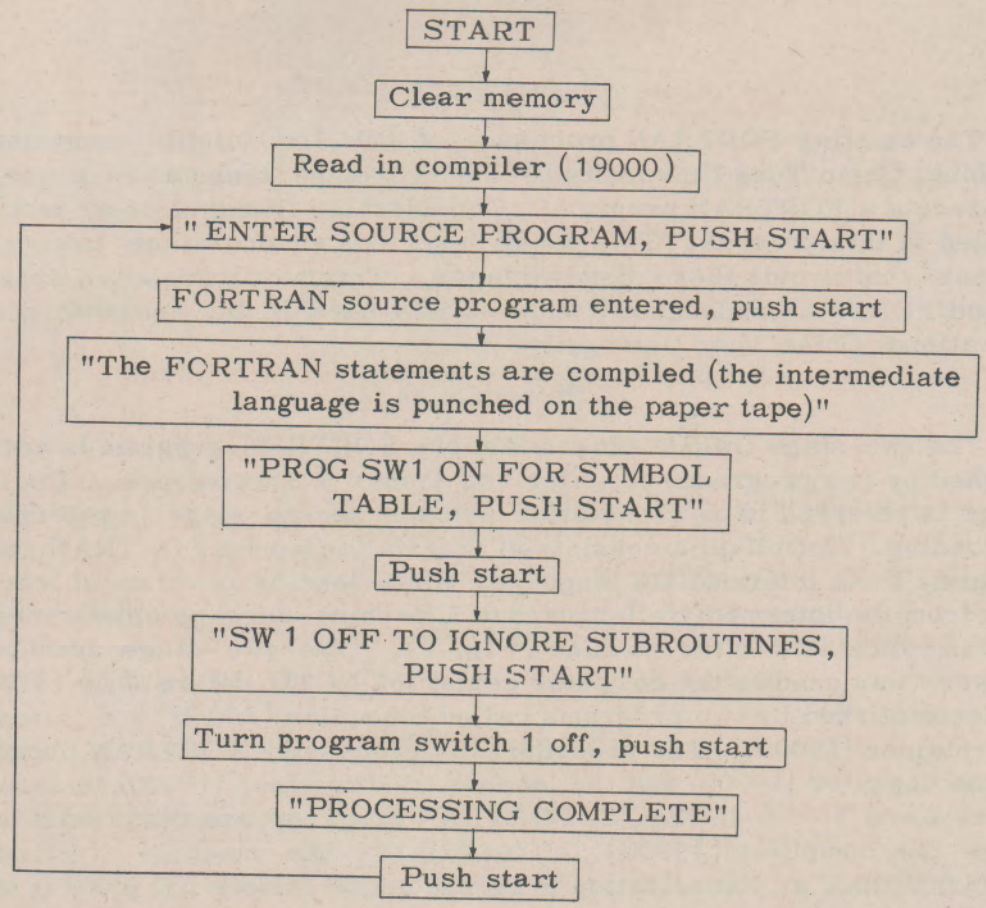

Fig. 2. First-stage translation process (compiling).

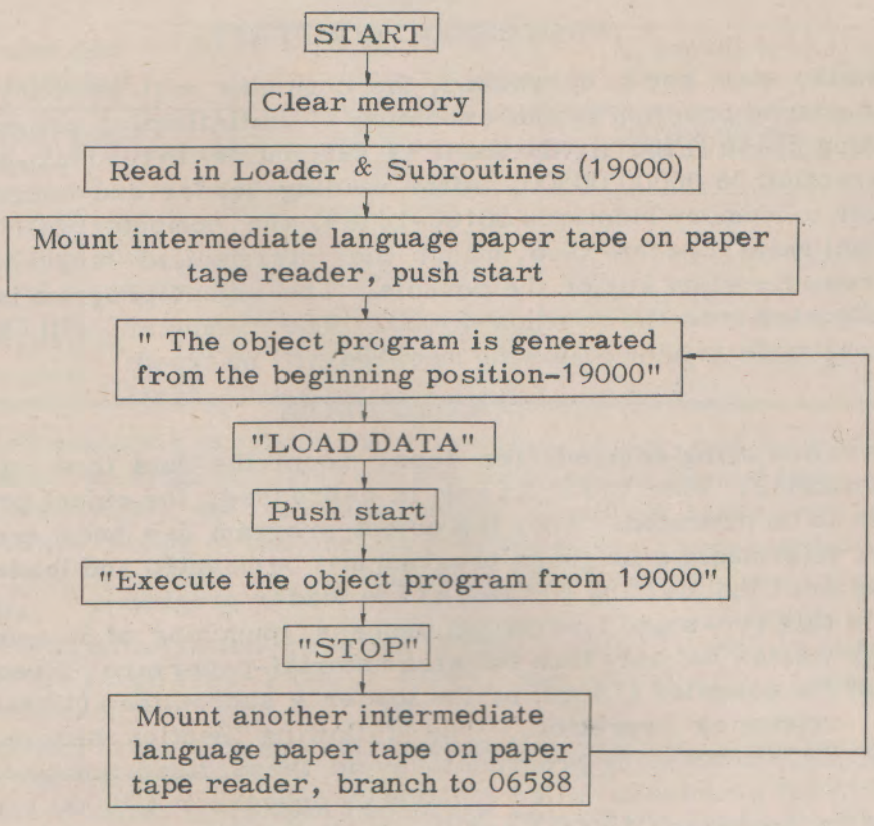

Fig. 3. Second-stage translation process (loading).

ENTER SOURCE PROGRAM, PUSH START

Mount the source tape and push start button. The FORTRAN source program will be compiled and the intermediate language will be punched on the paper tape. Sensing of the END statement signals the compiler that the entry of source statements is completed. The computer halts in the manual mode and the following message will be typed:

PROG SW1 ON FOR SYMBOL TABLE, PUSH START

If it is desired to get a listing of the symbol table, set program switch 1 on. If the listing is not desired, turn program switch 1 off. In either case, the start key on the console must be depressed. If program switch 1 is on, a listing of the symbol table is typed. The computer will halt in the manual mode and the following message will be typed:

SW 1 OFF TO IGNORE SUBROUTINES, PUSH START

Turn switch 1 off and push start button. The computer will halt in the manual mode and the following message will be typed:

PROCESSING COMPLETE

When the start key is depressed, the computer will be ready to compile next source program by the execution of initialization program. In the loading phase (Fig. 3), the loader & subroutines is read in by the machine instruction 36 00000 00300. After reading loader and subroutine programs to memory locations 00100-16383, the computer will halt in the manual mode. In this time, mount the intermediate language tape and depress the start key on the console. The object program is generated and located from the position-19000. The computer will halt in the manual mode and the following message will be typed:

LOAD DATA

If data are being entered from tape, mount the data tape on the paper tape reader. When the start key is depressed, the object program will begin to be executed. When the object program has been executed, another intermediate language program may be mounted and loaded by branching to 06588(starting position of the loader program).

In this two-stage translation process, punching of intermediate language wastes not only time but also material-paper tape. Frequent reading of the compiler (19000) or the loader & subroutines (19000) is also a time consuming operation. The following section describes a new FORTRAN processing program to avoid these disadvantages. The new FORTRAN processor called FORTRAN processor (19000) is obtained from the compiler (19000) and the loader & subroutines (19000) with some necessary modifications.

2. FORTRAN Processor (19000)

To achieve a one-stage translation process, the FORTRAN processor (19000) should contain arithmetic table, initialization program, main compiler, loader, and subroutines. In addition to store these programs, the memory should reserve locations for symbol table, intermediate language, and object program. The memory is occupied by these programs and tables in the manner of Table 1.

Table 1 :

Name	Required memory locations	Actual located addresses in memory
Arithmetic table	301	00100-00400
Initialization program	589	17800-18388
Main compiler (in two copies)	17145×2 = 34290	00401-17545 24500-41644

Loader and subroutines	16284	41646-57929
Symbol table area	1810	58190-59999
Intermediate language and object program	5500	19000-24499
Correction program	~1000	Remaining locations

Since the initialization program, main compiler, loader, and subroutines are all obtained in original form from the compiler (19000) and the loader & subroutines (19000), the following points should be taken into consideration in the design of the one-stage translation process and all the correction programs made for the following modifications are listed in the Appendix.

(1). Since the initialization program of the compiler (19000) is executable only in the range from 40000 to 40215, the initialization program should be corrected.

(2). The symbol table area is contracted and its position is changed from 40480-59999 to 58190-59999.

(3). The intermediate language is moved to memory locations beginning at 19000 instead of punched on the paper tape. All the "punch on tape" instructions of the main compiler should be corrected by branching to a subroutine that moves the intermediate language to consecutive memory locations beginning at 19000. This subroutine also detects the error condition when the total intermediate language exceeds 5000 memory locations.

(4). Since the loader and subroutines are executable only in the range 00100-16383 instead of 41646-57929, there should have a program that moves the loader and subroutines to the correct positions before loading.

(5). All the "read from tape" instructions of loader program should be corrected by branching to a subroutine that loads record from memory locations beginning at 19000.

(6). To save time in the translation process, all the messages, except error messages and START message (that is an abbreviation of ENTER SOURCE PROGRAM, PUSH START), are omitted. The listing of symbol table area is controlled by the bit of memory location 00007 instead of the program switch 1.

(7). In the two-stage translation process, the contents of symbol table should be punched on the intermediate language tape because the memory is cleared before loading. This operation is omitted in the one-stage translation process because the symbol table is not destroyed.

(8). For a program that has a DIMENSION statement, there is another point that should be taken into consideration. Since a dimensional entry only occupies two entries of the symbol table during compiling phase, it must be expanded during loading phase. These kinds of operation destroy the original symbol table. Thus for a program without a DIMENSION statement seven types of correction program are satisfactory for an object program less than 5000 digits. For a program that has a DIMENSION statement, another correction program should be added. However, the object program and the occupied symbol table just before loading should be less than 5000 digits.

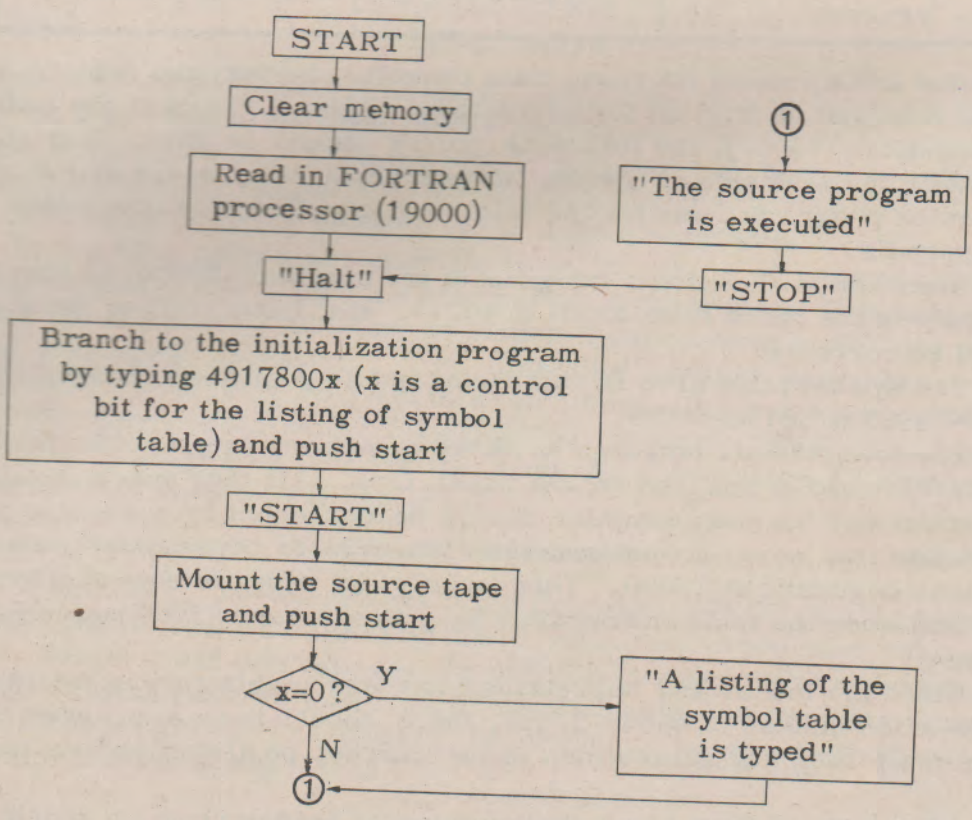

Fig. 4. One-stage translation process (processing).

Fig. 4 illustrates the one-stage translation process. When the FORTRAN processor (19000) is read in by the instruction 36 00000 00300, the initialization program, main compiler, loader, subroutines, and correction program are transferred to memory locations as shown in Table 1. When the computer halts in the manual mode, the initialization program will be executed by branching to 17800. The initialization program initializes a symbol table area of 181 entries and moves a copy of the main compiler from 24500-41644 to 00401-17545. When the initialization program has been executed, the computer will halt in the manual mode and

the START message will be typed. The source tape should be mounted at this time and the start key should be depressed. Now, the computer enters into compiling phase by execution of the main compiler from 00402. The source tape is read in and compiled statement by statement. Generated intermediate language is moved consecutively to memory locations beginning at 19000. The compiling phase is terminated by sensing of the END statement in the source tape. Since the loader and subroutines are executable only in the range 00100-16383, the loader and subroutines should be moved from 41646-57929 to 00100-16383 before entering into the loading phase. In the loading phase, the loader program accepts input of intermediate language from core locations beginning at 19000. Object codes are generated beginning at the same location 19000. Object codes may be overlapped on the intermediate language because the record of intermediate language accepted is always longer than the object code generated. Sensing of the end control code in the intermediate language signals the end of the loading phase. At this time, the object time address of a symbol or subroutine name is determined and stored in the symbol table area. Now the computer branches to 19000 for the execution of the program. When the program has been executed, the computer will halt in the manual mode, another source program may be translated by branching to the initialization program located at 17800.

3. Discussion and Conclusion

There are two constraints in the use of the FORTRAN processor (19000). Firstly the intermediate language generated should be less than 5000 locations. Secondly, numbers of different variables or constants in the source program should be less then 181. This is not a serious problem, since it is enough to translate a moderate size FORTRAN program. When the first constraint is not satisfied, the message PROGRAM TOO LARGE will be typed and the computer will halt in the manual mode. When the second constraint is not satisfied, the message ERROR NO. 4 will be typed and the computer will halt in the manual mode.

Assume there is a program that generates 4000 characters of intermediate language, it wastes a time of 4.5 minutes to punch these characters on tape (punch speed 15 characters/sec.) by the two-stage translation method. By the one-stage translation process, the intermediate language will be generated in a few seconds. This is a large save of CPU time. In addition, there are a large amount of paper tape wasted in the punch of intermediate language. However, The FORTRAN processor (19000) has also the capabilities of punching intermediate language and loading from intermediate language paper tape by adding a small correcting tape. All these advantages make the FORTRAN processor (19000) preferable to be used in the Computer Center of National Chiao Tung University after July, 1970.

Acknowledgement

The author wishes to acknowledge professor Chi-Chang Lee for his many valuable suggestions during this work.

Appendix

Correction programs-It can be read in by the instruction 360000000300. For convenience, the symbol E is used to represent an end of line symbol E/L.

Tape loader program

360006200300260005900071260007800071160007203002

250001100000360007200011E

250001100011490000CE

Correct errors (1) and (2).

178001815 0E

41000000000025178101770125178111770141000000000

3159880180281617878581992600000178111117878000101

41787859880471787201300155819800000310040124500 16

17962181562500000177011217962000051417962183834 61

795601100492433600000000000000000062585963466258 59

63005356474600535647000045675746004567570000416 34

1554641634155004356624600435662000062495546006249550000E

182671827 2E07512E

177011770 2LE

243362436 7E

161848618900016185861900049080 68E

Correct error (3).

183921857 6E

1418486240004718492013003400000001023918443001 00

485759564759415400635656005341594745 0‡00001900 00

000031184861839145185281848649185520000011184860

000149185040000011184860000149000000000 0E

316613166 8E4907766E

277492776 1E490757400000E

$\bar{3}1673\bar{3}1704E$

$1618570\bar{0}3662161839\bar{1}\bar{1}68034918392E$

$\bar{2}9141\bar{2}9153E490760600000E$

$\bar{3}1705\bar{3}1736E$

$1618570\bar{0}5054161839\bar{1}\bar{1}68034918392E$

$\bar{2}9665\bar{2}9677E490763800000E$

$\bar{3}1737\bar{3}1768E$

$1618570\bar{0}5578161839\bar{1}\bar{1}69274918392E$

$\bar{2}9701\bar{2}9713E490767000000E$

$\bar{3}1769\bar{3}1800E$

$1618570\bar{0}5614161839\bar{1}\bar{1}68034918392E$

$\bar{3}0113\bar{3}0125E490770200000E$

$\bar{3}1801\bar{3}1832E$

$1618570\bar{0}6026161839\bar{1}\bar{1}68034918392E$

$\bar{3}0137\bar{3}0149E490773400000E$

$\bar{3}1833\bar{3}1864E$

$1618570\bar{0}6050161839\bar{1}\bar{0}65574918392E$

$\bar{3}0185\bar{3}0197E49581.2400000E$

$\bar{5}8124\bar{5}8155E$

$1618570\bar{0}6098161839\bar{1}\bar{1}70434918392E$

$\bar{3}0241\bar{3}0253E495815600000E$

$\bar{5}8156\bar{5}8187E$

$1618570\bar{0}6154161839\bar{1}\bar{1}70434918392E$

$\bar{3}0297\bar{3}0309E492410000000E$

$\bar{2}4100\bar{2}4131E$

$1618570\bar{0}6210161839\bar{1}\bar{1}68034918392E$

$\bar{3}1649\bar{3}1661E492413200000E$

$\bar{2}4132\bar{2}4163E$

$1618570\bar{0}7562161839\bar{1}\bar{0}04024918392E$

$\bar{3}5051\bar{3}5063E492416400000E$

$\bar{2}4164\bar{2}4195E$

1618570$\bar{1}$09641618391$\bar{1}$20354918392E

$\bar{3}$8765$\bar{3}$8777E492419600000E

$\bar{2}$4196$\bar{2}$4227E

1618570$\bar{1}$46781618391$\bar{1}$68034918392E

$\bar{4}$0373$\bar{4}$0385E492422800000E

$\bar{2}$4228$\bar{2}$4259E

1618570$\bar{1}$62861618391$\bar{1}$68034918392E

Correct error (4)

$\bar{3}$1901$\bar{3}$1913E491868000000E

$\bar{2}$4300$\bar{2}$4336E

1618933$\bar{0}$00001618940$\bar{0}$00004906588000000E

$\bar{1}$8680$\bar{1}$8891E

2618710$\bar{1}$75501118710$\bar{4}$154615$\bar{0}$000000000

1118691$\bar{0}$00051418691$\bar{1}$77054718680012 00

3100100416462618782$\bar{1}$75502$5\bar{0}$000017701

1118775$\bar{0}$00051418775$\bar{1}$77054718764012 00

1618691$\bar{1}$75501618775$\bar{1}$75501618486$\bar{1}$9000

1618586$\bar{1}$90001618916$\bar{5}$99394924300E

$\bar{1}$7546$\bar{1}$7702E

$\bar{0}$0401$\bar{0}$0555$\bar{0}$1171$\bar{0}$1281$\bar{0}$2437$\bar{0}$3909$\bar{0}$4643

$\bar{0}$5019$\bar{0}$5020$\bar{0}$5021$\bar{0}$5070$\bar{0}$5828$\bar{0}$6573$\bar{0}$7728

$\bar{0}$7737$\bar{0}$8935$\bar{0}$8946$\bar{0}$8967$\bar{0}$8977$\bar{0}$8987$\bar{0}$8997

$\bar{0}$9007$\bar{0}$9017$\bar{0}$9027$\bar{0}$9831$\bar{0}$9842$\bar{0}$9862$\bar{1}$0438

12292$\bar{1}$2445$\bar{1}$2592E

Correct error (5)

$\bar{1}$8580$\bar{1}$8676E

00$\bar{1}$900000000311859$\bar{1}$1858$\bar{6}$451862818586

491865200000111858$\bar{6}$00001491860400000

1118586$\bar{0}$00014200000000000E

4832$6\bar{4}$8338E1718592$\bar{0}$8957E

$\bar{4}$8410$\bar{4}$8422E1718592$\bar{1}$2375E

Correct error (6)

$\bar{3}1565\bar{3}1603E$

$41000000000041000000000041000000000041E$

$\bar{3}2179\bar{3}2198E41000000000003907999E$

$\bar{3}1865\bar{3}1891E41000000000041000000000041E$

$\bar{4}8938\bar{4}8964E41000000000041000000000041E$

$\bar{3}0973\bar{3}1057E$

$41000000000041000000000041000000000$

$41000000000041000000000041000000000$

$430746600007E$

Correct error (7)

$\bar{4}8134\bar{4}8146E160705\bar{0}0001E$

$\bar{4}8446\bar{4}8458E171892\bar{0}8956E$

$\bar{1}8907\bar{1}8982E$

$\bar{0}0000\bar{5}993900000$

$161891\bar{1}\bar{0}000041\bar{0}000000000041000000000$

$11189160\bar{0}001491771000000E$

$\bar{1}7710\bar{1}7794E$

$1118911\bar{0}C0011118921\bar{0}0001251892\bar{1}18916$

$1418911\bar{0}0060471895801200111892\bar{1}\bar{0}0001495794200000E$

$\bar{5}7942\bar{5}8122E$

$2518921\bar{0}0401141894\bar{0}\bar{0}001475801401200161893\bar{0}0000$

$1618940\bar{0}00004200000000001218916\bar{0}0120241238918916$

$4706912013001618921\bar{0}8956261893318916492426000000$

$2618916123891118940\bar{0}\bar{0}001490691200000E$

$\bar{2}4260\bar{2}4291E$

$2218933123893318933000004958086E$

Correct error (8).

$\bar{3}1661\bar{3}1865E$

$2607645004161516870000002607621167\bar{9}3$

$160761\bar{6}1686925000000000120762100\bar{0}1$

1407621$\bar{0}$00004607730012001407616$\bar{1}$6810

460770601200120761600$\bar{0}$1490761000000

381681000200490759800000260774807616

380000000200490780200102E

$\bar{2}$7749$\bar{2}$7761E

495794200000E

$\bar{5}$7942$\bar{5}$7973E

161857$\bar{0}$03662161839$\bar{1}$68034918392E

$\bar{2}$9141$\bar{2}$9153E

495797400000E

$\bar{5}$7974$\bar{5}$8005E

161857$\bar{0}$05054161839$\bar{1}$68034918392E

$\bar{2}$9665$\bar{2}$9677E

49$\bar{5}$800600000E

$\bar{5}$8006$\bar{5}$8037E

161857$\bar{0}$05578161839$\bar{1}$69274918392E

$\bar{2}$9701$\bar{2}$9713E

495803800000E

$\bar{5}$8038$\bar{5}$8069E

161857$\bar{0}$05614161839$\bar{1}$68034918392E

$\bar{3}$0113$\bar{3}$0125E

495807000000E

$\bar{5}$8070$\bar{5}$8101E

161857$\bar{0}$06026161839$\bar{1}$68034918392E

$\bar{3}$0137$\bar{3}$0149E

491771000000E

$\bar{1}$7710$\bar{1}$7741E

161857$\bar{0}$06050161839$\bar{0}$65574918392E

$\bar{3}$1805$\bar{3}$1817E

491774200000E

$\bar{1}$7742$\bar{1}$7773E

16185700771816183911681049183 92E

31829318 53E

26077530761649189080000 0E

1890818939E

161857007754261839107753491 8392E

48446484 58E

171859208957E

1887218891E

4100000000004906588E

31637316 39E

41E

References

1. Peter Wegner, "Programming Languages, Information Structures, and Machine Organization," Central Book Co., Taipei, Taiwan, 1968.

2. W. Wesley Peterson, "Computing with the IBM 1620 Computer," Central Book Co., Taipei, Taiwan, 1965.