UNIVAC®

SOLID-STATE SYSTEMS

TECHNICAL BULLETIN

S-4 ASSEMBLY SYSTEM

90 Card Configurations

Programmers Reference

First Edition

SYSTEMS PROGRAMMING
LIBRARY SERVICES

SPLS

UP 1774.7

SYSTEMS PROGRAMMING
LIBRARY SERVICES

SPLS

UNIVAC®
SOLID-STATE SYSTEMS

ADDENDA

ADDENDA to "*S-4 ASSEMBLY SYSTEM,*
90 Card Configurations" *Manual*
UP 1774.7

1. Recovery from Possible Main Memory Error

Because of certain internal program conditions, a particular main-memory (parity) error may occur (under very rare circumstances) when S-4C10 is used.

S-4C10 is built to operate without index registers. However, the routine, for its own purposes, carries certain instructions negatively. In effect, this is equivalent to modifying the instructions by IR_2, and for this reason, if IR_2 is present in the computer, it must be cleared to zeros. S-4C10 tests to see whether or not IR_2, if it is present, contains anything other than zeros. It does this by loading a constant into rA, and then loading the contents of the same location, modified by a IR_2, into rL. If rA and rL test equal, then either IR_2 is not present or it is set to zeros, either of which is acceptable, and S-4C10 proceeds to normal processing. If rA and rL test unequal, then IR_2 is present and is set to some value other than zeros, in which case S-4C10 proceeds to load IR_2 with zeros prior to normal processing.

If the address fabricated is invalid, when the"30" (load rL) instruction is modified by IR_2, the computer will halt with -30 1660 2663 displayed in rC, along with a main-memory error indication. Should this occur, recovery can be easily effected by selecting "c" and depressing the RUN button.

2. Modifications for Effecting Double-Spacing on the Printer

In order to effect double-spacing on the Printer, the following modifications should be made to S-4C10:

LOCATION	NOW IS	SHOULD BE
4589	30 4704 4420	30 4079 4420
4097	30 4704 4556	30 4079 4556

P R E F A C E

This manual is intended as a programmer's reference manual to familiarize the programmer with the basic elements of the S-4 90-Card Assembly System. The S-4 assembler is designed to facilitate the coding of data-processing applications by reducing both coding time and error frequency.

The S-4 90-card system operates on a basic 5000-word system which must contain as a minimum configuration, a Card-Reader, a Read-Punch Unit, and a Printer.

TABLE OF CONTENTS

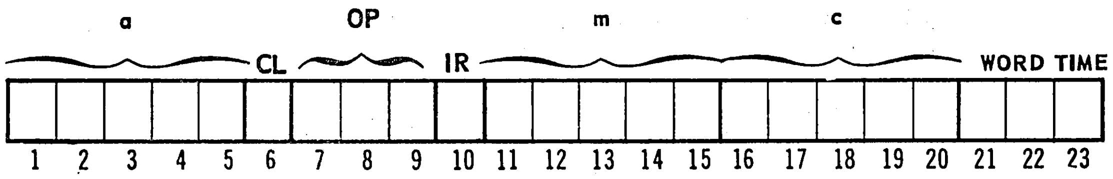
I. INTRODUCTION

The S-4 Assembly System is a one-pass assembly-language translator. Input is a source program coded in S-4 assembly language. Output is an object program in UNIVAC® Solid-State machine code acceptable to a standard loading-routine. For documentation and desk-checking purposes, printed listings of both the source and object programs, and printouts of the various tables of the assembly system are provided.

The source program for an S-4 assembly is written on the S-4 coding form (see Appendix A). Each line consists of 49 character positions, and is equated to a particular columnar position on the punched card. The four leftmost character positions are primarily for the convenience of the programmer and are used to specify program line-sequence. This information, though punched on the card, is ignored by the assembler since it sets up its own line or card sequence.

A. INSTRUCTION FORMAT

The basic S-4 instruction is specified in the following format:



It consists of a symbolic a, m, and c field; a symbolic OP code field; a class designation, an index register specification and a word-time field. The instruction is contained in columnar positions 46-68. Explanation of each element in the instruction format follows.

1. Symbolic "a" Field

   This field, specified in columns 46-50 contains the address of an instruction or constant and may be any one of the following:

   a. Absolute machine address
   b. Permanent tag
   c. Temporary tag
   d. Local reference point
   e. Blank address
   f. Interlace address
   g. Overflow address
   h. Register address
   i. Regional address

   The various types of specifications listed above are discussed in chapter 3.

2. Class Field

   This field, specified in column 51, is used when the information in the remaining fields, columns 52-90, is to be treated in a special way by the assembly program. Any one of eight specifications may be made in this field. The following is a list of the specifications and the actions they cause the assembler to perform:

Character	Assembler Action
C	The data in columns 52-90 is to be treated as a comment and, therefore, carried over to final output unchanged.
P	If the line represents a constant, the primed portion of the contents of columns 56-65 will be stored in the object program.
U	If the line represents a constant, the unprimed portion of the contents of columns 56-65 will be stored in the object program.

Character	Assembler Action
N	If the line represents a constant, the contents of columns 56-65 will be translated and the numeric portion stored in the object program.[1]
Z	If the line represents a constant, the contents of columns 56-65 will be translated and the zone portion stored in the object program.[1]
I	The tens complement of the data produced from 56-65 will be stored in the object program.
Δ	No special action imposed.
*	The line is treated as a title line and is printed on the assembly record at the beginning of a new page.

3. Symbolic "OP" Field

The 3-digit symbolic operation-code field, columns 52-54 may contain any of the following:

(a) ΔΔΔ (blanks) to indicate that the content of the m and c fields, columns 56-65 is a constant and should be treated as column 51 directs.

(b) A 3-digit symbolic OP code or an S-4 control operator. If a symbolic OP code, it will be translated to its machine code equivalent during assembly.

(c) A 2-digit machine coded instruction in form Δnn. It will appear in final output unchanged. If this option is used, the word-time field must also be employed.

[1]In normal 90-card assemblies, this specification will not apply; however, the S-4 Assembly System provides the ability to assemble 90-Tape programs on the 90-card computer.

4. Index Register Field

The index register field, column 55 contains a numeric specification when index-register modification is indicated for the m address of the associated instruction:

Entry	Meaning
1	Use index register 1.
2	Use index register 2; load negative.
3	Use index register 3.

5. Symbolic "m" Field

The symbolic m field, columns 56-60, may contain any of the entries specified for the symbolic a field. In addition, it may also be part of a constant entry.

6. Symbolic "c" Field

The symbolic c field, columns 61-65, specifies the next instruction and may contain any of the entries indicated for the symbolic a field. It may also be part of a constant entry.

7. Word-Time Field

The word-time field, columns 66-68, is analyzed during assembly. If it contains a numeric specification, the numeric value will be assigned as the maximum number of word-times for the operation specified in the OP field. If the field is not entirely numeric, it is treated as part of the remarks field. When the entry in the symbolic OP field is a machine-coded instruction, the number of word-times from a to c is specified by the programmer in the word-time field. With this, an additional specification is entered in the most significant position of the field, column 66 to direct the assembly of the instruction line. The format of the entry in the word-time field is;

xnn

where; x is a Δ or 5 bit if c specifies the next instruction.

4 bit if m specifies the next instruction.

7 bit when m is not to be used to update the latency counter (the clock),

1 bit if x is considered part of nn field.

lnn is the number of word-times between a and c. If three digits are needed to express the number of word-times, the 1 bit in the x position may be employed.

B. REMARKS

The remarks field, columns 66-90 if the word-times field is not used, contains comments concerning the source program. All data is this field will be part of the side-by-side, object and source-code listing printed by the assembler.

C. DATA CONSTANT CCDING

The following considerations apply when data constants are specified in the source program:

1. The OP portion of the instruction will contain blanks (ΔΔΔ).

2. The IR field indicates the sign of the constant;

   Δ if positive

   - if negative

3. The constant is entered in the symbolic m and c fields.

4. If the constant contains undigits, the class field must be blank (Δ).

5. A blank in the constant field will be interpreted as a zero.

6. The class field may contain any of the values I, N, Z, U, P, or Δ.

7. The following designations are used for undigits:

Bit-Pattern	Character
0101	A
0110	B
0111	C
1101	F
1110	G
1111	H

II. S-4 ADDRESSING

The S-4 Assembly System provides for many types of addressing to ensure a high degree of programming flexibility. That is, programmers are able to select from a variety of addressing forms that format most suited to a particular need in a computer application. This capability increases the power of the system as a programming tool and provides a versitility not present in ordinary machine coding. The following is a listing of the various addressing forms provided by the S-4 System:

1. Blank Addressing
2. Tag Addressing
3. Absolute Addressing
4. Register Addressing
5. Regional Addressing
6. Interlace Addressing

A. BLANK ADDRESSING

If the generation of absolute addresses in the object program is to be left to the assembler, the a, m, or c portion of the instruction involved may be left blank. If either the m or c field of an instruction is left blank, it will be an indication to the assembler that a reference is being made to the next consecutive line of coding; therefore, the a field of the next instruction must also be left blank to allow the assembler to assign the same absolute address as the m or c of the previous instruction which made the reference. For example:

a	OP	IR	m	c
X	LDA			Y
			00000	00005
Y	STA		Z	
	LDL		P	Q

Assuming that X, Y, Z, P, and Q are some form of S-4 address specification, the converted machine-coded version might appear in final output as;

a	OP	m	c
0404	25	0406	0408
0406	00	0000	0005
0408	60	0410	0412
0412	30	0414	0557

When both the m and c symbolic address fields are left blank, the symbolic a fields of the next two instructions in sequence must also be left blank. The blank m field will reference the next line in sequence; the blank c address will reference the second line down. For example;

a	OP	IR	m	c
X	LDA			
			00000	00001
	ADD		Y	Z
Z	STA			

Assuming that X, Y, and Z are some form of S-4 address specification, the converted machine-coded version might appear in the final output as;

a	OP	m	c
0200	25	0202	0204
0202	00	0000	0000
0204	70	0207	0209
0209	60	....	....

It should be noted that absolute addresses will be assigned in the object program only if the blank m or c portion is normally specified in the machine-coded instruction. That is, certain instructions require no specification in either the m or c field; for example, a 26 (CLA) or 06 (CLX) in which the c portion is not specified, or the 77 (ATL) in which the m address is not specified.

B. TAG ADDRESSING

A tag is a symbolic specification or address that relates nonconsecutive lines of coding. Tags provide connecting links between operations by relating the m or c portion of an instruction with the a portion of another instruction that has been, or is yet to be specified. They may be used to denote the entrance and exit lines of a common subroutine; to transfer from one operation to another; to reference lines that are to be modified; or to transfer control to a common line at the end of a branching chain of instructions.

Provision is made in S-4 for three types of tag specification:

1. Permanent tags

2. Temporary tags

3. LRP (Local Reference Point) tags

Permanent tags are employed to preserve relationships that will be maintained throughout the program. That is, since programs are normally subdivided into logical units or sections, the permanent tag provides a method of referencing either across or within these program sections. Temporary tags are generally employed to establish relationships between lines of coding within a logical section of the program and are generally not referenced by lines of coding from another section. The LRP tag is a special form of temporary tag. It is generally used within comparatively short coding segments and allows a relationship to be established without exhausting the combined total of 300 temporary and permanent tags permitted in a program.

As each tag is specified, it is entered in a tag table along with its assigned absolute address. Temporary tags may be cleared from the tag table at any time (but usually at the end of a logical program section) to permit their reassignment in another portion of the source program. Permanent tag entries are maintained in the tag table throughout the program. However, should a permanent tag become inactive (that is, no reference made to it during the remainder of the program), it may be cleared from the symbol table. Clearing of the tag table will be discussed when the S-4 control operators are considered.

Program subdivision is left entirely to the discretion of the programmer since no formal method is provided by the S-4 System. It should be noted, however, that sections assembled first will receive preferential treatment as far as optimization is concerned. Therefore, it is in the interest of the programmer to assemble the most important sections first.

1. Permanent tags

   The permanent tag is specified in the a, m, or c field in the following format:

   x n n n m

   Here; x is any alphabetic or special character.

   nnn is any combination of alphabetic, alphanumeric and/or special characters.

   m is the area in storage to which the tag is to be assigned and should contain one of the following:

   a. A blank for standard-access memory assignment.

   b. Any character for high-speed-access memory assignment except an O or P.

   c. An O or P for overflow (c+1) condition. (See "Overflow Addressing.")

   Ideally, a permanent tag specification should, in some way, be indicative of the function performed by the tagged procedure or should conform to some meaningful tag coding scheme. For example, the tag

   G R O S S

   might specify the location at which the result of a gross-pay compution is stored in high-speed-access memory.

2. Temporary Tags

A temporary tag is specified in the a, m, or c field in the following format:

x n n b y

Here; x n n is the tag identifier.

x may be any alphabetic, numeric or special character except Δ.

nn may be any two-digit numeric if the tag is to be assigned to standard-access memory; it must be blank if the tag is to be assigned to high-speed-access memory.

b is blank if fast-access memory is to be assigned; it is numeric if high-speed-access memory is to be assigned.

y must be a numeric if b is numeric. If b is blank, y may be one of the following:

(a) blank

(b) O or P for an overflow condition.

When employing a temporary or permanent tag specification, the following should be observed:

a. Absolute locations may be assigned to temporary tags by the programmer.[2]

b. Individual tags may be cleared from the tag table (released for reassignment) at any time during an assembly.[3]

---

[2] See SYN Control Operator.

[3] See EQU Control Operator.

c. The tag table may be entirely cleared of temporary tags at any time during an assembly and new temporary tags initiated.[4]

d. Once a tag has been cleared from the table, any further reference to the tag is treated as if no previous reference had appeared, consequently, a new absolute address will be assigned.

3. Overflow (c+1) Addressing

Overflow and c+1 conditions can result from either an arithmetic operation or an abnormal condition in an input or output unit. In an arithmetic operation, it is caused by the generation of a numerical quantity beyond the digit capacity of the register that is to receive it. In an input or output unit, it might be the result of any of a number of mechanical conditions (Printer out of paper, RPU card jam, for example). In either case, the instruction that will be executed next is determined by the addition of 1 to the c address of the instruction in which the overflow or c+1 condition occurs or is detected.

There are eight instruction codes that can result in overflow or c+1 conditions:

S-4 Code	Machine Code
ADD	70
SUB	75
DIV	55
PRN	11
PFD	16
HCC	72
RCC	81
TBU	F6

Whenever one of these codes is used, a subroutine should be coded that will handle the possible overflow or c+1 condition. In S-4 coding this is accomplished by the use of temporary or permanent tags with an O, or P in the LSD of the tag.

If there is no overflow, control will be sent to the instruction coded with the O tag in the a address portion. If overflow occurs, control is sent to the instruction containing the P tag in the a address portion.

[4]See HED C Control Operator.

When coding for overflow and c+1 conditions, the following should be observed:

a. The c address portion of the line in which overflow may occur must be in the O form of a permanent or temporary tag.

b. O and P tags do not have to follow the line in which overflow may occur but may be placed at any point during the assembly. The only restriction is that when temporary tag form is used, all program references must be made before a HED C control operator is introduced.

c. Overflow tags must be counted as part of the tag limits. Each set (O and P) is counted as one tag.

d. Unless a HED F control operator is in effect, O and P tags will be assigned to fast-access memory.

Examples of overflow (c+1) coding:

a	OP	m	c
	LDA	X12△△	
	ADD	X13△△	L01△0
L01△0	STA	X14△△	N1△△△
L01△P	JMP		

The temporary tag L01 contains an O in the LSD. The c+1 tag contains a P in the LSD.

4. LRP (Local Reference Point) Tags

LRP tags permit relationships to be established between nonconsecutive lines of coding without using permanent or temporary tags. The LRP tags a line with which communication is to be made, through an m or c portion of a prior, a succeeding, or the same instruction.

The LRP tag is specified in the following format:

n△△△x

Here; n is the LRP identifier and must be a numeric in the range 0-9

ΔΔΔ is always blank.

x is the storage-allocation position and may be:

Δ (blank) if standard-access memory is to be assigned.

H if high-speed access memory is to be assigned.

An LRP tag is referenced in the following manner. Note that the reference must indicate the direction or relation of the tagged line to the line referencing it; that is, whether the tagged line is a previous, a succeeding, or the same instruction. The format is:

nyΔΔx

Here; n is the identifier assigned to the LRP tag (0-9)

y is the direction indicator when the LRP being referenced is assigned by the assembler to standard-access storage; otherwise, this position is blank (Δ).

x is the direction indicator when the LRP being referenced is assigned by the assembler to high-speed access storage; otherwise, this position is blank (Δ).

ΔΔ is always blank.

Note that either x or y may be specified at one time and never both. One or the other will always be blank depending on the storage assignment of the LRP. The following may be entered in either x or y:

B if the LRP-tagged line, with which communication is made, exists in a backward direction from the referencing line. B may be either in y or x and is the only reference not dependent on storage assignment.

F if the LRP-tagged line, with which communication is made, exists in a foreward direction from the referencing line.

H if the LRP-tagged line, with which communication is made, is the same as the line in which the reference occurs. If this specification is employed but there is no LRP tag in the a address, the assigned address of the current entry in the a field is established in the LRP table as the address of that LRP tag.

LRP tags are assigned in a two-part tag table, based on their order of specification. The two parts of the table are designated B(backward) and F(forward). When an LRP is specified in a field, its absolute address is entered in B. For example, the LRP 1△△△H indicates that a high-speed storage assignment is to be made for LRP 1. The table entry is made in the following manner:

LRP	B	F
0		
1	4211	
2		
3		
4		
5		
6		
7		
8		
9		

From this point on, all backward references to LRP 1 will address 4211. The address will remain valid until a new LRP 1 appears in a succeeding instruction. When a new LRP 1 appears, 4211 will be cleared from the table and a new assignment will be made.

If a forward reference to an LRP is made from an m or c field, its absolute value is assigned in F. For example, 1FΔΔ indicates that the next LRP 1 encountered is in standard-access memory. The table entry is then made in the following manner.

LRP	B	F
0		
1		1342
2		
3		
4		
5		
6		
7		
8		
9		

When the LRP 1 is encountered in some succeeding instruction, it shifts from F to its corresponding position in B clearing any entry in that position. Until this shift occurs, any entry made for a previous LRP 1 in the B column will remain valid.

LRP	B	F
0		
1	1342	
2		
3		
4		
5		
6		
7		
8		
9		

Examples of LRP coding:

a.
```
a         OP        m         c
1         LDA       GIN       SIN
```
LRP 1 is assigned in standard-access storage.

b.
```
GIN       LDA       1B        SIN
```
1B refers to previous LRP 1.

c.
```
GIN       STA       SIN       2    F
```
Next LRP 2 encountered is in high-speed access storage.

d.
```
4    H    LDL       GIN       SIN
```
LRP 4 is assigned in high-speed access storage.

e.
```
GIN       LDX       4F        SIN
```
Next LRP 4 encountered is in standard-access storage.

f.
```
SIN       LDX       5H        GIN
```
LRP 5 is assigned the address given to SIN.

g.
```
GIN       LDL       SIN       2    B
```
2   B refers to previous LRP 2.

h.
```
GIN       LDA       2    F    SIN
```
Next LRP 2 encountered is in high-speed access storage.

C. ABSOLUTE ADDRESSING

Fixed computer locations (absolute addresses) are referenced in the following manner:

Δ n n n n

Here; Δ must be blank.

nnnn must be the specific memory address and must be within 0000 and 4999.

D. REGISTER ADDRESSING

When it is necessary to address a register, the address is coded by using the two most significant digits of the desired symbolic address field.

R n △ △ △

Here; R must be R.

n may be A, X, or L depending upon the register to be referenced.

△△△ must be blank.

Whenever program control passes to a register address, the contents of the register should be displayed on the next line with the appropriate register address in the symbolic a field.

Register addresses always produce the following absolute addresses:

RA△△△	000A
RX△△△	000C
RL△△△	000B

E. REGIONAL ADDRESSING

Data, or instructions stored as data, are placed in reserved areas of memory known as regions.[4] An entry in any of these areas may be referenced by a regional address. The format of a regional address is:

a n n n n

Here; a is any alphabetic or non-blank special character.

nnnn is the entry number within the region area. This will be in the range 0001 through the highest entry number reserved. Thus, if 200 locations, 0200 through 0399, have been reserved for region B, the B regional addresses would be B0001 through B0200. B0001 would be location 0200; B0009 would be 0208, etc. If 50 locations four word-times apart have been reserved in band 10 for region C, the C regional addresses would be C0001 through C0050. C0001 would be location 1000; C0002, 1004, etc.

[4] See REG Control Operator.

F. INTERLACE ADDRESSING

Reference to an input or an output interlace is accomplished by use of symbolic interlace address.[5] The format of this address is:

k i m n z

Here; k represents the input/output device and must be one of the following:

H - Card Reader Interlace.

R - Read-Punch Unit Read Interlace.

O - Read-Punch Unit Output (Punch) Interlace.

P - Printer Interlace.

T - Tape Interlace.[6]

D - RANDEX Drum Interlace. (Also for first or second tape interlace if desired.)[6]

i represents the number of the interlace and must be a numeric in the range 0 through 9. S-4 provides ten interlace patterns for each input/output unit.

m refers to the particular level of an interlace word or to the band of the interlace. It must be one of the following:

U - Unprimed portion of a word

P - Primed portion of a word

N - Numeric portion of a word

Z - Zone portion of a word[6]

B - Entire interlace (used with buffer load and unload instructions)

nz specifies the word referenced. This specification depends upon the input or output unit designated by k. When k is H, R, or O:

n is 1 when the word being addressed is located in the <u>first read interlace</u> of the Card Reader or RPU; or the <u>punch interlace</u> of the RPU.

---

[5] See INT Control Operator.
[6] Applicable only for USS 90 Tape assemblies.

n is 2 when the word being address is located in second read interlace of the Card Reader or RPU.

z is the card word; 0-9.

When k is T:[7]

nz is a numeric in the range 00 through 99.

When k is P:

nz is a numeric in the range 01 through 13.

When k is D:

nz is a numeric in the range 00 through 47. If a number greater than 47 is specified, it will be treated as a tape interlace. This capability permits 20 interlace patterns to be specified for tape as opposed to ten for all other units.[7]

When an entire interlace is specified (m is B):

nz is 00 for the untranslated interlace.

01 for the translated interlace.

nz is the number of lines to advance on the printer if k is P.

[7]See footnote 6.

III. S-4 CONTROL OPERATORS

Command of the S-4 Assembly process is exercised by the use of Control Operators. Their function is to provide:

1. Memory allocation controls.
2. Access to, and control of, tag table content.
3. The use of a Constant Library.
4. Development of program testing aids and their inclusion in an object program.

Control operators are coded in the Symbolic Operation Field as three digit mnemonics.

A. ASSEMBLY CONTROL OPERATOR

RST - Initialize For Assembly

The RST Operator sets conditions for an assembly. By using the RST Operator between programs, a series of programs can be assembled in one computer run. The functions performed by the RST Operator are:

1. Sets HSP listing page number to 1.
2. Clears the Availability Table.
3. Clears the Symbol Table.
4. Clears the Interlace Table.
5. Clears the Region Table.
6. Stores the new program title from the RST Operator (see RST Operator format).
7. Clears the Card Number Counter.
8. Clears the Word-Time Clock.
9. Initializes the assembly program to non-forward search mode.

10. Initializes modes of some Control Operators

HED D

HED K

HED N

HED P

HED Y

The RST format is:

OP	m	c
RST	ppppp	ppppp

RST is the mnemonic used in the symbolic OP field.

pppppppppp	is the ten-digit positions for the alphanumeric and/or special characters that identify the source program to be assembled. Unused positions are coded as blanks (Δ). The specification is punched in columns 1-10 of the output cards and is printed in columns 121-130 of the page header-line of the printed listing.

B. STORAGE ALLOCATION OPERATORS

In any program, certain areas and/or locations must be restricted from assignment during the assembly process (data-storage locations, interlaces, tables, packaged subroutine locations, etc.). In S-4 coding, this is accomplished by the use of the following Control Operators.

1. BLR - Block Reservation

The BLR Operator is used to reserve a given number of locations at a fixed increment from each other beginning and ending at specified addresses.

The format is:

OP	m	c	w/t
BLR	bbbbb	eeeee	iii

BLR is the mnemonic for Block Reservation.

bbbbb is the absolute address or defined symbol at which block reservation is to begin.

eeeee is the absolute address or defined symbol at which reservation is to end.

iii is the increment between locations. If this field is blank or 000 the increment is considered to be 001. Increments less than 200 are modulo drum size. Increments greater than 200 are modulo 200 and will remain within the band specified.

Examples of BLR coding:

a.

OP	m	c	w/t
BLR	0403	0793	005

This would reserve every fifth location, beginning with 0403, through 0793.

b. BLR 4400 4599 257

This would reserve every fifty-seventh location within band 44.

c. BLR GET START

This would reserve every location between the previously defined tags GET and START.

2. BLA - Block Availability

The BLA operator makes available a given number of locations at a fixed increment from each other beginning and ending at specified addresses. It is the reverse of the BLR operator.

The BLA coding is in the same format as that of the BLR.

Examples of BLA coding:

a. BLA 0403 0798 005

This would make available for S-4 Assembly assignment every fifth location, beginning with 0403, through 0798.

b. BLA 4400 4599 257

This would make available for S-4 Assembly assignment every fifty-seventh location within band 44.

3. REG - Regional Specification

The REG Operator defines a region composed of a specified number of elements beginning at a certain location and separated by a given increment. REG coding is in the following format:

OP	m	c	w/t
REG	xnnnn	yyyyy	iii

REG is the mnemonic for Regional Specification.

x is an alphabetic or non-blank special character.

nnnn is the absolute address at which the region is to begin.

yyyyy is the absolute address or defined symbol at which the region is to end. If yyyyy is blank, the region will be defined but the elements of the region will not be restricted in the memory table.

iii as defined under BLR.

Examples of REG coding:

a. REG A1700 Δ1842 ΔΔΔ

This would reserve every location from 1700 through 1842 for Region A. ΔΔΔ could also have been coded 000 or 001.

b. REG B1200 Δ1350 010

This would reserve every tenth location from 1200 through 1350 for Region B.

c. REG S4600 ΔΔΔΔΔ 203

This would set up every third location within band 46 (modulo 200) as region S. The region will not be restricted because yyyyy is blank.

4. INT - Interlace Pattern Reserve

The INT Operator reserves an interlace for the input/output unit specified in the m field. This interlace will be located in the memory area specified in the symbolic c field:

OP	m	c
INT	xy△△z	△nnnn

INT is the mnemonic for Interlace Pattern Reservation.

x is the particular input or output unit interlace:

H - Card Reader Interlace.

R - Read-Punch Unit Read Interlace.

O - Read-Punch Unit Output (Punch) Interlace.

P - Printer Output Interlace.

T - Tape-Synchronizer Interlace.[8]

D - RANDEX® Drum Interlace (also for Tape Interlace if desired).[8]

y is the number of the interlace and must be a decimal digit in the range 0 through 9. This allows up to ten interlaces for each input/output unit (twenty for tape since both T and D may be used).

△△ these digits are always blank.

z is 0 if automatic translation is not to be used (unless x = D).

is 1 if automatic translation is to be used (unless x = D).[8]

is △ (blank) if the input/output unit involved does not use translation (such as Tape and RANDEX units).[8]

[8]Applicable for USS 90 Tape assemblies only.

If x = D, z = 0 if a RANDEX input interlace is desired (200 locations). z = 1 if a RANDEX output interlace is desired (48 locations).

△nnnn is space followed by the memory area in which the interlace is to be reserved. nnnn must be an even band number (0200, 0400, 1000, 4200, etc.).

It should be noted that while overlapping of interlaces is permissible, the condition must be kept in mind when coding the source program.

Examples of INT coding:

a. INT H1△△1 2000

This would reserve the locations for an HSR translated interlace on band 20.

b. INT D1△△0 0800

This would reserve the locations for a RANDEX input interlace on band 08.

5. SYN - Synonym

The SYN Operator will reserve a single location. It may be used for the following purposes:

a. To equate a tag to specific memory location.

b. To provide a time relationship between two tags.

The coding of a SYN Operator is:

OP	m	c	w/t
SYN	xxxxx	yyyyy	iii

SYN is the mnemonic for Synonym.

xxxxx is the symbolic address which is to be equated to the content of the symbolic c field.

yyyyy may be (1) a previously defined symbol, the location of which is assigned.

(2) an absolute address to be assigned.

If yyyyy is an undefined symbol the SYN Operator will be bypassed and an error note 5 will be printed.

iii is the word-time increment to be added to yyyyy before assigning the xxxxx address.

Examples of SYN coding:

```
a.  SYN     STOP△      △0674
```

This will cause the tag STOP to be assigned 0674 as its address. If tag STOP had already been assigned an address, this would establish 0674 as the address.

```
b.  SYN     JOE△△      SAM△△      015
```

The tag JOE will be assigned an address fifteen word-times greater than that assigned to tag SAM.

```
c.  SYN     ENT△△      EXIT△      030
```

The tag ENT will be assigned an address thirty word-times greater than that assigned to tag EXIT.

C. ALLOCATION-CONTROL OPERATORS

Certain control operators included in the S-4 Assembly System permit control of the allocation processes governing the address assignments of instructions, constants, and tag addresses without source program revision. Thus, latency needs, discernable only from an over-all understanding of a source program, can be met.

These Allocation Control Operators are coded in the symbolic OP field and the most significant digit position of the symbolic m field:

OP	m
HED	a△△△△

HED is the mnemonic used in the Symbolic OP Field.

a△△△△ is the alphabetic designating the desired Allocation Control Operator followed by four blanks.

The Allocation Control Operators are:

1. HED B - Initiate Forward Search

   The use of a HED B Operator will cause a scanning of lines ahead of the line which an undefined symbolic specification is encountered. This scanning will proceed until:

   a. A "C" is found in column 51 (46).

   b. A constant is detected.

   c. Ten lines have been scanned.

   d. Any operator is encountered (this stops it temporarily).

   e. A HED A Operator is encountered (this stops it permanently).

   When any one of these conditions is met, forward search is terminated for that sequence and allocation is made on a reverse direction; that is, from the line on which forward search is stopped, back to the line in which it was begun. When forward search has been initiated, it will continue to operate until a HED A control cperator is encountered. That is, when any of conditions a through d has been met, allocation is made. Normal allocation is then resumed until an undefined symbolic specification is encountered; forward search again takes effect. This process will continue until a HED A control operator is found.

   Under certain conditions a HED B Operator will have no effect:

   a. the memory table is filled.

   b. the symbol table is filled.

2. HED A - End Forward Search

   The HED A Control Operator terminates the forward search initiated by a HED B Control Operator, if any.

3. HED D - Extend to High-Speed Memory (when necessary)

   The HED D Control Operator, in effect, extends allocation from standard-access to high-speed memory for minimum latency address assignment. That is, if a HED D Control Operator is in effect, and an unassigned address cannot be optimally assigned in standard-access memory, high-speed-access memory is examined for an optimum location. If such a location is found, the assignment is made. If no such assignment is possible in the high-speed area, the standard-access area is searched for the next best location. If not found, high-speed memory is searched. This process continues until assignment is made.

4. HED E - Terminate HED D

   The HED E Control Operator eliminates the allocation modes initiated by a HED D Control Operator.

5. HED F - Assign High-Speed Storage

   The HED F Operator will cause all succeeding unassigned symbolic and blank addresses to be assigned in high-speed-access memory.

6. HED N - Resume Normal Allocation

   The HED N Operator eliminates the assembly modes initiated by an HED F Control Operator.

7. HED Z - Allocate in Standard-Access; Execute in High-Speed Storage.

   The HED Z Operator is only applicable for source programs that will operate on tape configurations. The instruction lines following a HED Z Operator will have addresses allocated in standard-access memory. The m and c addresses, however, will be allocated in high-speed memory. The address assignment is such that when a band-to-band transfer through the tape buffer is made the instructions will occupy the correct locations for minimal latency.

Before a HED Z Operator is used, all of memory must be reserved except the area in which the instructions are to be executed. Besides this memory reservation, the m and c fields of the HED Z line must contain the band specification of the standard-access band in which allocation is to begin and the high-speed band in which execution will take place.

For example:

OP	m	c
HED	Z2000	△4600

This would cause succeeding instructions to be allocated to locations beginning in band 20 for execution in band 46.

8. HED Y - Terminate HED Z Control

The HED Y Operator returns the assembly process to the normal allocation mode.

9. WDT - Word-Time Control

The WDT Control Operator is used to modify the word-time clock; setting, resetting, and/or adding to it for the next instruction or a portion of the next instruction. The information concerning the desired modification to the word-time clock is coded in the m, c and word-time fields.

OP	m	c	w/t
WDT	sssss	△△xyz	iii

WDT	is the mnemonic for word-time clock control operator.
sssss	is a tag or a word-time level or an absolute address if x is S or blank.
△△	these columns are always blank.
x	is A if an increment is to be added to the word-time clock.
	is S if the word-time clock is to be set to a particular level.

y is A if next a address is to be modified.

is M if next m address is to be modified.

is C if next c address is to be modified.

z is △ (blank) if the word-time clock is not to be reset after the action specified by x.

is R if the word-time clock is to be reset to its previous level plus normal incrementation.

NOTE: If xyz are blank, it will be interpreted as SA△.

Examples of WDT coding:

a. WDT SAM△△ △△△△△ 015

Would result in the word-time clock being set to the level of SAM plus 15 for the next a address.

b. WDT △0013 △△△SA△ 000

Would set the a address of the next line to be assembled to level 013.

c. WDT SAM△△ △△SAR 010

Would set the next assembled a address to the level of SAM plus 10 then reset the word-time clock to its prior setting plus normal incrementation.

d. WDT △△△△△ △△AA△ 015

Would add 15 to the word-time clock before assigning the next a address.

e. WDT △△△△△ △△AMR 015

Would add 15 plus normal incrementation to the word-time clock before assigning the next m address and then reset the word-time clock to its previous setting plus normal incrementation.

f. WDT △1345 △△SC△ 000

Would set the word-time clock to level 145 (band relative address of 1345) before assigning the next c address.

g. WDT △0015 △△AM△ 025

Would add 25 plus normal incrementation to the word-time clock before assigning the next m address and 15 plus normal incrementation after assigning the next m address.

h. WDT △0006 △△AAR 007

Would add 7 to the word-time clock before assigning the next a address. The clock would then be reset to its reading before the a address assignment and 6 in addition to normal incrementation added to it before assigning the next m address.

D. TAG TABLE CONTROL OPERATORS

Specific control of the Tag Table content is provided through the use of two Control Operators:

1. EQU - Equivalence

The EQU Control Operator can equate a tag to a specific value or location or clear the symbolic tag from the Tag Table so that the tag may be reused. It is similar, though not identical to the SYN Control Operator (see page 3-6). An EQU will not restrict a location in the memory table.

The coding of an EQU Operator is:

OP	m	c
EQU	xxxxx	yyyyy

EQU	is the mnemonic for Equivalence.
xxxxx	is the symbolic address to be equated to yyyyy.
yyyyy	is the defined actual value or symbolic address or spaces if xxxxx is to be erased from the Tag Table.

Examples of EQU coding:

a. EQU CAA△△ A12△△

It is assumed that A12 has been defined in the assembly process. This EQU will cause CAA to be permanently stored in the symbol table whereas A12 will be erased by the next HED C card.

b. EQU INCRA △0002

This will relate INCRA to an increment of 0002 when used in the symbolic m field of a LIR or IIR instruction line.

c. EQU GROSS H1U21

H1UN1 was previously defined by an INT entry. The processing of this EQU will equate GROSS to card word 1, unprimed, HSR, second read station.

d. EQU BED△△ △△△△△

Since the Symbolic c Field is blank, permanent tag BED is erased from the Tag Table and is available for redefinition.

2. HED C - Clear Temporary Tag Table

The HED C Control Operator clears the temporary tag table. It is usually used to mark the end of a particular section or segment of coding within the source program.

E. CONSTANT LIBRARY OPERATORS

A constant library may be established for an installation using the S-4 Assembly System. This is done by assigning each constant a tag. Thus, a source program may be prepared using the specific tags for the constants that are desired. Through the use of HED L and HED K control operators, the entire constant library may be introduced as part of the source program. Only those constants whose tags have been referenced during the previous assembly of the source program coding will be used. Constants not referenced will be ignored.

1. HED L - Process Constant Library

   The HED L Control Operator is used just before the constant library deck is introduced. It will cause only those constants whose tags have been referenced by the previously assembled source program to be accepted from the constant library.

2. HED K - End Constant Library Processing

   The HED K Control Operator indicates that the processing of the constant library is finished. Normal assembly mode will be resumed.

F. PROGRAM TESTING OPERATORS

The S-4 Assembly System includes a number of control operators designed to aid in program testing and to permit assembly around previously allocated portions of memory.

1. HED X - Printer Output

   The HED X Control Operator eliminates the assembly output on the RPU. Printer output continues.

   To eliminate the assembly mode initiated by a HED X Control Operator, HED P Control Operator must be used.

2. HED P - Resume RPU Output

   The HED P Control Operator causes assembly output to be resumed on the Read-Punch Unit. Printer output continues.

3. PPA - Print and Punch Availability Table

   The PPA Control Operator provides a printed listing of locations not used and a punched card deck as a reloadable record of the memory availability table. The deck produced is in multiple word-per-card format (8 words per 90-column card). By loading such a deck in another assembly run, the memory availability table would be restored and further assembly could be initiated without any overlay of previous assembly allocation.

The format of the printed listing caused by a PPA Control Operator is:

```
0000  nnnnn  0800  1000  nnnnn  1800  ....  4000  nnnnn  4800
0001  nnnnn  0801  1001  nnnnn  1801  ....  4001  nnnnn  4801
0002  nnnnn  0802  1002  nnnnn  1802  ....  4002  nnnnn  4802
  '     '      '     '     '      '     '     '     '      '
  '     '      '     '     '      '     '     '     '      '
  '     '      '     '     '      '     '     '     '      '
  '     '      '     '     '      '     '     '     '      '

0199  nnnnn  0999  1199  nnnnn  1999  ....  4199  nnnnn  4999
```

The first n after 0000 refers to location 0000; the second n after 0000 refers to location 0200; the third n to 0400; the fourth n to 0600; the fifth n to 0800; and so on for the balance of the printout.

Each n will contain a utilization key the interpretation of which is:

0 for an unused location.

1 for a location used in an a address.

2 for a location used as a data address in the m address field.

3 for a location used as a next instruction address in the m or c address field.

4 for a location used as a next instruction address and data address.

5 for a location used as an a address and next instruction address.

6 for a location used as an a address, data address, and next instruction address.

8 for a location reserved by a BLR or INT operator.

9 for a location reserved by a REG or SYN operator.

4. PAT - Print Availability Table

The PAT Control Operator will provide only a Printer listing of the Memory Availability Table. No punched card deck will be produced.

5. SYP - Print Symbol Table

The contents of the Symbol Table may be listed on the Printer at any time during the S-4 Assembly by use of an SYP Control Operator. Each line of the listing consists of five symbol sets each of which is in the following format:

```
ttttt   f   aaaa
```

ttttt is the tag.

f is the utilization key (the interpretation is the same as for the PPA utilization keys).

aaaa is the address allocated to the tag.

IV. ASSEMBLY FEATURES

A. ASSEMBLY OUTPUT

1. Punched Card Output

The output deck of an S-4 Assembly is the machine-coded object program in a one-instruction-per-card format plus the symbolic coding and remarks. This format is acceptable to the standard loading routine of the object program.

The output of the S-4 90 Card System conforms to PTA01-PTA02 format. All fields on the card are punched in card code except the assembled instruction which is in the USS code and located in the unprimed portion of columns 21-30. Positive instructions have a key of 3; negative a key of 4. A four bit, when necessitated by IR modification, is automatically buffed onto the instruction.

2. Printer Output

The S-4 Assembly Printer output is a side-by-side listing of the assembled and the symbolically coded lines (including the Word-Time and Remarks Field) and codes to indicate error detected during the assembly processing. The latter, when necessary, are printed on the extreme right of the listing.

3. Printer Error Codes

The error codes that may appear on the Printer listing are:

Code	Error
△ (Blank)	No error detected.
1	Region or interlace not defined.
2	A non USS form of machine (absolute) address was encountered.
3	An input error in blank-address linkage was discovered.
4	Incorrect Class or Symbolic OP code was used.

Code	Error
5	This line was bypassed because of an error condition. Error condition was one of the following: a. SYN Operator - (1) c address symbol undefined. (2) an address is unavailable. b. EQU Operator - Spaces in m and c c. HED Operator - Invalid HED designation. d. INT Operator - Invalid INT designation. e. BLR and BLA Operators - Invalid address in m f. REG Operator - (1) Invalid address in m (2) Invalid REG designation
6	Symbolic specification has already been defined in a previous a address.
7	Two consecutive locations are not available for assignment to an overflow (c+1) set.
8	Memory full indicator. May be result of one of the following: 1. All memory was depleted and no assignment could be made.

b. A band-relative address required by a SYN Operator was unavailable. The next best address level was assigned.

If several errors appear on a line they will be noted in the same order as detected (class, OP, a, m, c).

For example, if both the class and OP were erroneous and the m address was a defined regional reference, 4, 4, 1 would appear as the error note.

B. PREPARATION FOR ASSEMBLY

After the program has been coded it is punched, one instruction line per card. The resultant deck is the input to the S-4 Assembly program. The sequence of the input cards is:

a. RST Card: The RST card contains the name of the program in the m and c field of the card.

b. Specials: If desired, PPA cards may be entered at this point, in order to restore the memory allocation as it was at the time of punching the PPA cards so that further assembly will be an area other than used in the prior program. In addition, cards controlling package routine changes, interlace changes and so on, should be entered at this point.

c. BLR: Block Reservation cards.

d. REGion: Regional Reservation cards.

e. INTerlace: Interlace Reservation cards.

f. SYNonym: Synonym Cards.

g. EQUivalence: Equivalence Cards.

h. HED: HED cards as needed prior to initiation of main program detail card assembly.

i. Detail: Main program detail cards plus Control Operator cards as required.

j. HED L: If the constant assembly option (Paragraph 2.06) is utilized, a HED L card followed by the constant pool followed by a HED K card is required at this point.

k. SYP: Symbol Table printout (recommended but not required).

l. PPA/PAT: Print-Availability Table or Print and Punch-Availability Table.

m. RST: If further compilation to re-initialize.

C. THE FORWARD SEARCH OPERATION

The Forward Search function of S-4 is primarily intended to optimize latency where a branching operation is involved. This function enables the assembler to scan up to 10 lines ahead when a previously undefined symbolic specification is encountered and then make assignments in a backward direction.

A HED B Control Operator will initiate the function. It sets a switch that causes S-4 to enter Phase 1 (Forward Scanning) of Forward Search when the first undefined symbolic tag (LRP not included) is encountered. The data on this card and on subsequent cards will be stored in a special table that will be accessed when Backward Assignment (Phase 2) commences.

Forward Scanning will normally continue for up to 10 lines. However, if certain conditions are encountered, the assembler will enter Phase 2 prior to encountering this limit. Further, depending upon the condition, Backward assignment will commence either with the line in which the condition occurs or with the line preceding. Upon execution of Phase 2, the normal processing mode resumes, but only until another undefined symbolic tag is encountered, at which point the Forward Search cycle will be repeated.

If it is desired to permanently eliminate the Forward Search mode, a HED A card must be introduced. If, at this point, Phase 1 is in effect, Phase 2 of the current cycle will be executed before termination.

The conditions that suspend Phase 1 of Forward Search are as follows:

1. Where backward assignment will begin with the preceding line:

    a. A constant has been encountered.

    b. The card contains a "C" in the class field.

    c. Any control operator is encountered.

    d. Any LRP if the suffix is:

        (1) blank

        (2) H and the a address did not contain this LRP.

2. Backward assignment will begin with the same line when the 10th line has been scanned.

The "C" class field comments cards is used as a control for Forward Search processing. Those sections of coding containing branches to a common exit point should be preceded and followed by "C" cards. The HED A and HED B are also required to initiate and terminate the function.

V. OPERATING PROCEDURES

The S-4 Card 90 Assembly deck is composed of two sections:

a. The load section; Cards are numbered 1 through 44 in columns 89-90.

b. The multi instruction-card section; Each card contains up to 8 instructions in machine code in words 0, 1, 2, 3, 5, 6, 7 and 8 (unprimed portion), column 86 contains the number of instructions on the card. Cards 87-90 contain the first location to be loaded; the remaining instructions are loaded n+1.

A. LOADING S-4 ASSEMBLY

1. Read-Punch Unit

   a. Fill input magazine with blank cards.

   b. Depress FEED ONE CARD button three times.

   c. Depress RESET button.

2. Printer

   a. Determine adequacy of paper supply.

   b. Advance paper until six holes are above holding clamps.

3. Console and Card Reader

   a. Place S-4 self-loading deck in Reader input magazine.

   b. Depress GENERAL CLEAR.

   c. Depress ONE INSTRUCTION button.

   d. Key 72 0000 0000 into Register C.

   e. Depress RUN button (HSR should feed one card).

   f. Key 96 0000 0011 into Register C.

   g. Depress CONTINUOUS and RUN buttons.

   h. Upon successful load, the computer will stop on 67 3500 3500.

4. Optional Punch-Check Section

A punch-check section may be included by either manual key-in or by punching two cards with the following information:

Location	Instruction
3515	05 4917 4619
4901	26 4354 0000

B. EXECUTING S-4 ASSEMBLY

1. Place program deck in the input magazine of the Card Reader

2. Release 96 button.

3. Depress GENERAL CLEAR and RUN button.

C. LOAD ERROR STOPS

67 [all bits in m] 0025

A misload; reload deck again.

67 6YY6 0150

A read error in the Card Reader. Recommit the cards in stacker 1 and depress GENERAL CLEAR and RUN buttons.

67 0149 0150

The input hopper is empty. To continue loading, put cards in input hopper and depress GENERAL CLEAR and RUN buttons.

67 YTTY 0164

The card count on the sentinel card does not agree with the number of cards loaded. Locate the missing card and reload.

D. ERROR STOPS DURING EXECUTION

67 0444 cccc

Punch malfunction (c+1):
If card jam occurs, remove cards from output hopper. Clean out punch, reposition cards, depress ONE CARD three times, depress RESET, GENERAL CLEAR and RUN buttons.

67 0888 cccc

Punch malfunction:
Six attempts have been made to punch read check card with no success. Repair punch unit. To omit bad card and continue, depress RUN button (manually create lost card from printer listing).

67 0111 cccc

Empty input magazine indication in Card Reader: Either the Reader input magazine is empty or the picker knife has failed to feed a card. Take remedial action, select c, depress GENERAL CLEAR and RUN buttons to continue.

APPENDIX A – CODING FORM

The coding form shown on the following page can be used for the S-4 90 Card Assembly.

Remington Rand Univac
DIVISION OF SPERRY RAND CORPORATION

S4 ASSEMBLY FORM
UNIVAC SOLID-STATE SYMBOLIC SYSTEM

90

PROGRAM: ______________________ DATE: ____________ PAGE: ____ OF ____

LINE	LINE NO.	SYMBOLIC "A"	OPERATION		IR	SYMBOLIC "M"	SYMBOLIC "C"	REMARKS	
			CL	Symbolic				WORD TIME	
	13 14 15 16	46 47 48 49 50	51	52 53 54	55	56 57 58 59 60	61 62 63 64 65	66 67 68	69 70 75 80 85 90
1									
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									
21									
22									
23									
24									
25									

S-4 ASSEMBLER	OP CD	m	c	DESCRIPTION	WORD TIMES
ARITHMETIC					
ADD	70	m	c	(m) + (rA) → rA, if Overflow c=c +1.	5
SUB	75	m	c	(rA) - (m) → rA, if Overflow c=c +1.	5
MUL	85	m	c	(rL) x (m) → rA MSD, rX LSD.	5 + ND + SD
DIV	55	m	c	(m) ÷ (rL) → rA Quot, rX Rem. if Overflow c=c + 1.	20 + SOD + STCED
TRANSFER					
LDA	25	m	c	(m) → rA	4
LDX	05	m	c	(m) → rX	4
LDL	30	m	c	(m) → rL	4
STA	60	m	c	(rA) → m; m may not be a register	4
STX	65	m	c	(rX) → m; m may not be a register	4
STL	50	m	c	(rL) → m; m may not be a register	4
ATL	77	-	c	(rA) → rL	3
CTA	23	m	-	(rC) → rA	3
CLX	06	m	-	Zeros → rX; sign +	3
CLA	26	m	-	Zeros → rA; sign +	3
CLL	31	m	-	Zeros → rL; sign +	3
CAA	36	m	-	Zeros → rA; retain original sign	3
CAX	86	m	-	Zeros → rA, rX; sign of rL → rA, rX	14
LOGICAL					
BUF	20	m	c	Superimpose (m) on (rA) → rA	4
ERS	35	m	c	Extract (m) from (rA) → rA	4
SHR	32	0n00	c	Shift right n places. (rA) → (rX) → rA	3 + n
SHL	37	0n00	c	Shift left n places. Zeros → rA LSD	3 + n
ZUP	62	-	c	Suppress Zeros, Commas, RR in rA, rX.	4
JMP	00	m	-	Skip	2
HLT	67	-	-	Halt	-
COMPARISON					
TEQ	82	=	≠	(rA) : (rL)	3
TGR	87	>	≤	(rA) : (rL)	3

S-4 ASSEMBLER	OP CD	m	c	DESCRIPTION	WORD TIMES
TBU[2]	F6	m[3]	c	(B) → Tape Interlace on n band. L 198; NI 003. If AOT (rC) → rA, NI in C+1.	205
TRD	G2	Oxyz	c	Read 1 blk. from tape → B	17
TWR	H2	OxyO	c	Write 1 blk. from B → Tape	17
RANDEX[6]					
LSR	40	m	c	(m) → SIR	4
DPT	43	OnOO	c	Test Unit N; if head in position set H.P.F.F.[4]	No=3 Yes=4
DBT	92	Yes	No	Test H.P.F.F.[4]; if set (rC) → rA; N1 → m	3
TBU[2]	F6	m[3]	c	(B) → Tape Interlace on n band. L 198; NI 003. If AOR (rC) → rA, N1 in C+1.	205
TBL	C6	m[3]	c	Tape Interlace on m band. B. L 048; N1 053	205
TBT	C7	Yes	No	Buffer Test: Yes, (rC), → rA, Error FF → rL	No=3 Yes=4
TST	C2	Yes	No	Synchronizer Test: Yes, (rC) → rA	No=3 Yes=4
PDH	18	OUDSSTTB[5]		Position Read-Write Head. 125-550 ms.	125 m.s. (min)
DWT	28	OUDSSTTB[5]		Write RANDEX Blk (B) → Blk. specified	35 m.s. (min)
DRD	38	OUDSSTTB[5]		Read RANDEX Blk. Blk specified → B	105 m.s. + Lat.
DWC	48	OUDSSTTB[5]		Write/Check RANDEX Blk. (B) → Blk, specified and check.	105 m.s. + Lat.
DSW	58	OUDSSTTB[5]		Search Write. (B) → Blk. identified by Search.	35 m.s. + Lat.
DSR	68	OUDSSTTB[5]		Search Read. (Blk.) identified by Search → B.	35 m.s.
PAPER TAPE[6]					
RPT	A1	m	c	Read paper tape. If interlock rC → rA; NI at m.	3 if c; 4 if m
PBU	A2	0000	c	(B) → rA & rX. Numeric → rA; Zone → rX. If parity error c=c +1.	3
TTR	A3	m	c	Input Buffer-Loaded Test: Yes, (rC) → rA NI at m.	3 if c; 4 if m
PPT	A7	-	c	(rA) & (rX) → B: Initiate output punching.	3
TPB	A8	m	c	Output Buffer Free: Yes, (rC) → rA. NI at m.	3 if c; 4 if m

[1]Add 1 word

[1] Add 1 Word Time to instructions employing IR modification.

[2] If not executed, (rC) → rA, next instruction → c+1.

[3] m = bb00 if drum: where bb is band address.

[4] H.P.F.F. = Head Position Flip Flop.

[5] Instruction executed in SIR. 0 = unused digit position; U = RANDEX Unit D = Drum Half; SS = Sector; T = Track; B = Block.

[6] For use when assembling on USS 90 Card configuration for USS 90 Tape configuration.

S-4 ASSEMBLER	OP CD	m	c	DESCRIPTION	WORD TIMES
TRANSLATE					
CTM	12	-	c	RR (rA and rX) → MC-4 (rA). Zeros → rX	3
MTC	17	-	c	MC-4 (rA) → RR (rA and rX)	3
MTX[6]	C1	-	c	MC-4 (rA) → XS-3 (rA)	3
XTM[6]	C3	-	c	XS-3 (rA) → MC-4 (rA)	3
INDEX REGISTER					
LIR	02	m	c[1]	m of instruction word → IRi	3
IIR	07	m	c[1]	m of instruction word + (IRi) → IRi, and m of rA. Zeros → balance of rA	4
PRINTER					
PRN[2]	11	bbnn	c	Advance nn lines, print bb band. (rA), (rX) destroyed L197 : NI 189	592
PFD[2]	16	00nn	c	Advance nn lines	4
PBT	27	Yes	No	Printer Test: Yes (rC) → rA	No=3 Yes=4
CARD READER					
HBT	42	Yes	No	Buffer Test: Yes (rC) → rA	No=3 Yes=4
HBU	96	bb00	c	(B) → J interlace on bb and: L 198; NI 001	203
HBU[6]	96	bb01	c	(B) → MC-6 → J_T interlace on bb band. L 198; NI 013	215
HCC[2]	72	m	c	Card Cycle. Interlock (rC) → rA. NI → m	3 if c; 4 if m
HSS	47	0n00	c	Select Stacker n (n = 0, 1, 2)	3
READ-PUNCH					
RBT	22	Yes	No	Buffer Test: Yes, (rC) → rA	No=3 Yes=4
RBU	46	bb00	c	(B) → Ir Interlace on bb band. L 098; NI 101	203
RBU[6]	46	bb01	c	(B) → MC-6 → Ir interlace on bb band → B. L 098; NI 113	215
RCC[2]	81	bb00	c	Card Cycle. 0 interlace on bb band → B. L 098; NI 001	103
RCC[2,6]	81	bb01	c	Card Cycle. MC-6 in Or interlace on bb band → CC → B. L 098; NI 108	210
RSS	57	-	c	Select Stacker 1 (sort)	3
MAGNETIC TAPE[6]					
TST	C2	Yes	No	Synchronizer Test: Yes, (rC) → rA	No=3 Yes=4
TBL	C6	m[3]	c	Tape Interlace on m band → B. L 048; NI 053	205
TBT	C7	Yes	No	Buffer Test: Yes, (RC) → rA, Error FF → rL	No=3 Yes=5
TRW	F2	0xy0	c	Rewind UNISERVO x. (x=0-9) y=0, no interlock. y=2, interlock	600 ms.