2227

manual

SUPERSOAP

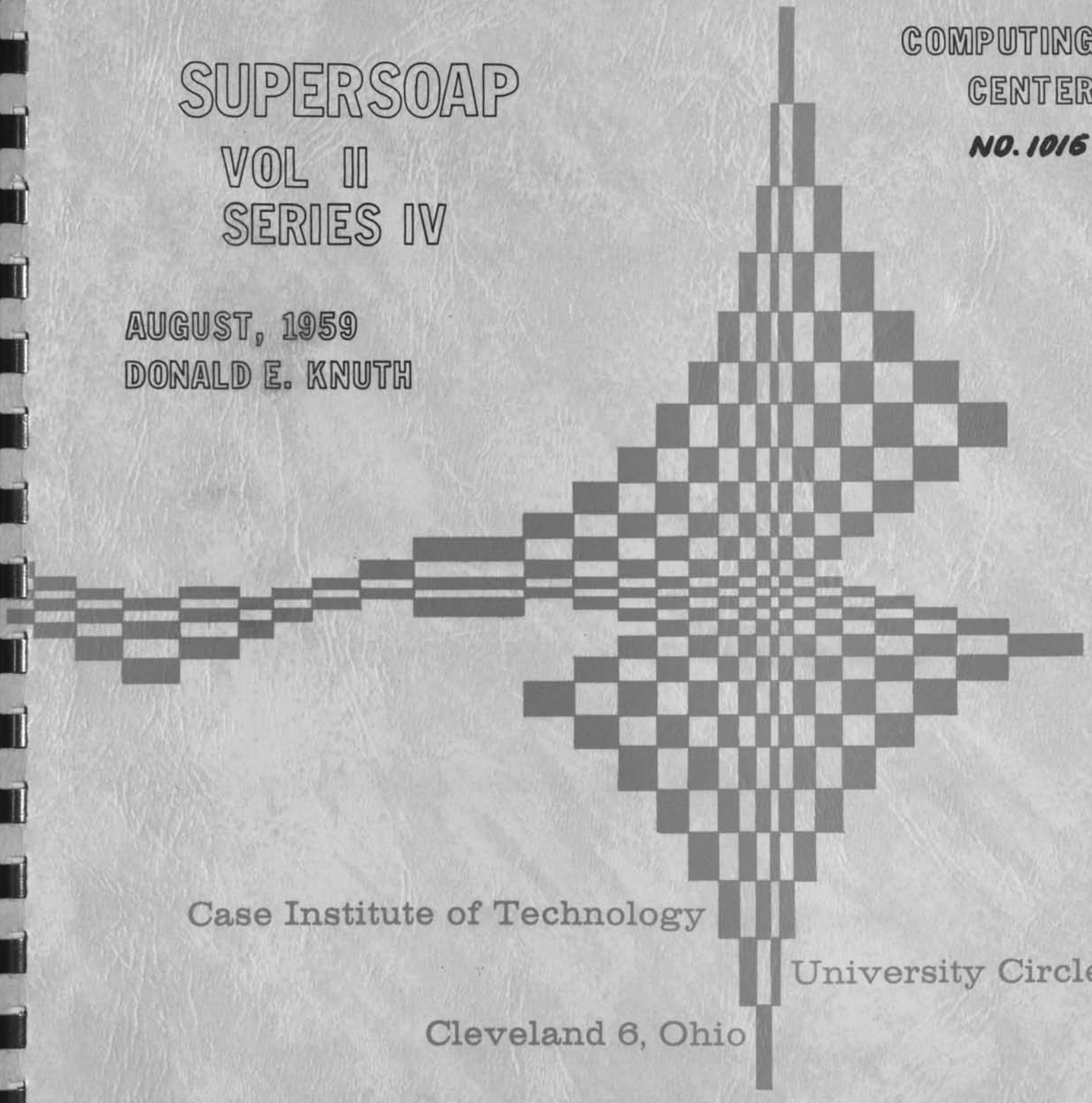
VOL II
SERIES IV

COMPUTING CENTER

NO. 1016

AUGUST, 1959

DONALD E. KNUTH



Case Institute of Technology

University Circle

Cleveland 6, Ohio

Super Soap

Assembly System for the

Augmented 650

Donald E. Knuth

August 20, 1959

S U P E R S O A P Assembly System for the Augmented 650

Abstract. SuperSoap is a major revision of Case SOAP III, designed to take advantage of a 650 with extra attachments. The present SuperSoap program requires index registers, core, a disk file unit, and the table-lookup-on-equal feature. No negative OP-codes are used anywhere by the program, so a machine with program register signs is not handicapped. A 12-word read buffer will not hinder SuperSoap either. A modification in which three tape units replace the disk file unit could be made with a moderate amount of work. In fact, only special pseudo-operations make the disk file unit necessary at all; a modification which needs neither disks nor tapes, but does little more than Soap 3, would be quite easy to prepare. (Now if you don't have core storage, index registers, or TLE either, you may find this manual interesting reading, but otherwise useless.)

The SuperSoap assembly system is designed especially for the do-it-yourself programmer, the man who creates his own programs for the 650, and who hasn't much use for canned routines. Library routines have not been totally neglected, however, and their use is also very flexible.

The speed of assembly is considerably faster than ever before -- when obtaining one-per-card output the punch runs full speed practically all the while, and when processing without punching any cards assembly speed of 140 to 150 cards per minute is average.

Experience in using the new equipment has brought more convenience into the assembly language. SuperSoap also introduces a sweeping variety of outputs: the user may obtain his results in any of 128 different ways. This includes tape, disk storage, seven-per-card, and on-line printer output. Quite a few useful new pseudo-operations are also to be found in SuperSoap. Some things were left behind, too: PIK, OFF, ONN, UNP, and REP have been deleted from the Soap 3 features for the SuperSoap repertoire because of infrequent use; and the block-reservation processor has been speeded up so much that BLR 0000 1999 (in two seconds) makes DRU unnecessary.

Soap 3 was written attempting to get as many features into 2000 memory locations as possible, but SuperSoap was written under a different philosophy;

speed was the prime consideration, and storage space was conserved only when speed was not appreciably decreased. A factor of roughly 3:1 in running time over Soap 3 has thus been obtained.

Acknowledgements. Some of the pseudo-op rules have become more logical thanks to Carnegie Tech's two-pass TASS (the change in absolute addresses, in regional and equivalence cards, and in type cards 4-9 were suggested by TASS). I also wish to thank Bill Lynch, Fred Way III, Joe Speroni, Mel Conway, George Haynam, Gilbert Steil, and George Petznick for the suggestions they have voiced (even though I didn't take them all to heart). Once again much gratitude must be given to the Case Computing Center for letting me chew up thousands of cards.

Disk Storage Allocation. SuperSoap uses disk 00 tracks 00-99, and disk 88 tracks 00-79, for temporary storage; and the program itself operates out of disk 38, tracks 00-66.

S U P E R S O A P

If you are going to get any good out of this manual, you ought to be reasonably familiar with 650 machine language instructions and operation. But that is all you really need to know; you don't need to be acquainted with any other assembly systems.

An assembly program such as SuperSoap greatly simplifies the task of machine language coding, because it almost completely relieves you of assigning actual storage locations. If you wish to refer to a certain memory location you can give it a symbolic name, which has a significant meaning to you, and you can always refer to it by means of this name. As a classic example, you might store salary in "WAGES" or sin x in "SIN X." The assembly system assigns actual locations to these names and produces the running machine language program. And the locations it chooses are optimized -- that is, they are put in the best place on the revolving drum for high-speed operation.

S.O.A.P. stands for Symbolic Optimum Assembly Program; S.U.P.E.R. has an obscure meaning which the reader may ponder at will; and the combination "SuperSoap" is supposed to indicate a highly efficient assembler.

Here is the overall agenda to follow when using SuperSoap: you code up your machine language program in the symbolic language described below; you punch your symbolic instructions onto cards, with one instruction per card; and then SuperSoap goes to work on the cards, producing output which is immediately reloadable as the desired machine language program. After this you make a trial run, find out why the computer hangs up, correct your symbolic cards, and reassemble until your program is working. It's as easy as that.

INPUT CARD FORMAT.

It will be easier to understand the symbolic language if you first learn the layout of Soap input cards.[1] As you know, every 650 instruction has four

---

1. This format is identical for IBM Soap2, Case Soap3, TASS, and probably several other assemblers unknown to me.

major parts: the Location (which will usually be called the L-address), the Operation (OP-code), the Data Adress (D-address), and the Instruction Address (I-address). These parts are the basis of SuperSoap language.

The L-address is punched in card columns 43 through 47, and it represents the location from which the assembled instruction will be executed.

The OP-code occupies columns 48 to 50. Either a symbolic three-letter operation (such as RAL for Reset-Add-to-Lower) or the standard two-digit number can be used for the OP-code. A complete list of the alphabetic symbol codes to use with SuperSoap is given in Appendix I; they are the same as those given in the most recent IBM 650 manuals. When you wish to use a numeric op, leave column 48 blank and punch the two-digit number in columns 49 and 50.

Example:

48
OP CODE
RAL

or

	65

The D-address is specified in columns 51-55. Column 56 is also associated with the D-address, and it is called the "D-tag." The D-tag is usually blank unless you wish the D-address to be indexed or you are in COR-mode -- these tagging features are expounded later, in Appendix II, if you are interested.

The I-address fits into columns 57-61, and the I-tag is put into column 62. The I-tag is, of course, analogous to the D-tag.

Columns 63 through 72 are yours to play with; they are reproduced in the symbolic output but generally have no effect on assembly. They must contain information acceptable to the alphabetic or special character device when you are punching one-per-card or printing on-line to get SuperSoap output. These columns are often used for comments which describe what is supposed to be going on.

The sign of the instruction is specified in column 42; this column must be blank when a plus sign is intended, and any punch in this column will cause a negative instruction to be assembled.

Column 41 specifies the card "type" and it is usually blank. Types 0-9 are described later, and when such a type is desired, the appropriate number should be placed in column 41.

Columns 41 through 72 are all reproduced on one-per-card or on-line-printer output. Columns 23-26 and 33-40 are used for hand optimization if using the HMO or MMO features described in Appendix IV. The remaining columns (1-22, 27-32, and 73-80) are ignored by the SuperSoap program, except columns 2, 4, 5, and 7 should not contain 12(Y)-punches since this will confuse the SuperSoap control panel.

Here is a summary of the whole format, to use when preparing your input deck:

Column(s)	Function
23-26	LH-field (see App. III)
33-36	DH-field (see App. III)
37-40	IH-field (see App. III)
41	Type
42	Sign
43-47	L-address
48-50	OP-code
51-55	D-address
56	D-tag
57-61	I-address
62	I-tag
63-72	Free Comments

TYPES OF ADDRESSES.

Subtle observation of the input card format will reveal that L-, D-, and I-addresses each take five columns of the card. The left-most column of each address (this would be column 43, 51, or 57 respectively) is traditionally called the "symbolizer part." Addresses in SuperSoap language can be specified in five different ways:

A. Absolute Addresses. An absolute address is used to specify a definite four-digit number, and this number is merely reproduced in the output. Addresses like 8001, or the D-addresses of shift commands, are generally given absolutely.

Rules: 1. The symbolizer part of an absolute address must be blank. 2. The numeric address is punched in consecutive columns starting next to the symbolizer part. Leading zeroes need not be punched unless the address desired is 0000, in which case merely "0" will suffice.[1]

Examples of absolute addresses: ("␣" MEANS THE CARD IS TO BE BLANK IN THIS COLUMN)

1.	␣1951	ASSEMBLES AS 1951
2.	␣0␣␣␣	ASSEMBLES AS 0000
3.	␣1␣␣␣	ASSEMBLES AS 0001
4.	␣123␣	ASSEMBLES AS 0123
5.	␣0123	ASSEMBLES AS 0123
6.	00123	NOT AN ABSOLUTE ADDRESS
7.	␣␣123	NOT AN ABSOLUTE ADDRESS
8.	0␣␣␣␣	NOT AN ABSOLUTE ADDRESS
9.	␣01␣␣	ASSEMBLES AS 0001
10.	␣10␣␣	ASSEMBLES AS 0010
11.	␣0␣1␣	NOT␣AN␣ABSOLUTE␣ADDRESS
12.	␣8003	ASSEMBLES AS 8003

B. Regional Addresses. A regional address is useful when you wish to do some sort of arithmetic on addresses; for example, if you want the fifth word of a punch band, or the k-th position in a table, or the like. You may define up to 47 different regions for this purpose.

The 47 regions (count them) are identified by an alphabetic, numeric, or special character punched in the symbolizer part. The address P0001 refers to the first word in region P, and R0012 means the twelfth word in region R. You will find regional addresses especially convenient for two main reasons: they save you from making unnecessary calculations, and they can be easily moved around the drum if desired.

Rules: 1. The symbolizer part of a regional address must be punched. 2. The other four digits of a regional address must all be numeric. 3. Any use of a regional address must be preceded by a card which defines the location of some word in the region. (This process is explained later in the manual.)

---

1. Deletion of leading zeroes generally saves a few milliseconds assembly time since less decoding is required.

4. When a regional address is calculated, the result is always a positive four-digit number; that is, if the value would be over 10,000, the carry would be dropped; and if the value turns out to be negative, it is complemented.

Examples: Suppose C0012 is defined to be location 0010; then C0002 is location 0000 and C0001 assembles as 9999. Conversely, if C0001 is defined to be 9999, C0012 will be location 0010. C0000 is a legal address; in this case, it would be location 9998.

C. Blank Addresses. Very frequently the D- or I-address of an instruction is supposed to refer to the next location, and you will never need to refer to this location again. In this case, you may leave the D- or I-address blank, and the L-address of the next instruction should be left blank, too. More than half of the addresses in a typical Soap program are blank; this makes insertion of additional instructions easy and is also a great time-saver for you when writing and punching the program. Both D- and I-addresses may be blank if they both refer to the next location.[1]

Rules: 1. When either the D- or I-address is blank, the L-address of the following instruction must be blank. 2. When both D- and I-addresses are non-blank, the L-address of the following instruction must be non-blank. 3. The first location of a program must not be blank. These rules are strictly enforced so that errors are detected before running your program.

D. Program Points. A program point is nothing but a spruced-up version of a blank address -- it is used to make easy references between nearby points in your program. The address "2F" (2 forward) refers to location "2" which appears later, and the address "2B" (2 backward) means location "2" which appeared before. The following simple routine, which for some reason writes 50 tape marks on tape Unit 8013 and then stops, should help to clarify this procedure.

---

1. Only the D-address is optimized in this case.

Example Program 1.

Soap Language

T P	S N	LOCATION	OP CODE	DATA ADDRESS	T	INSTR. ADDRESS	T
		4	RAA	49		2F	
		2	WTM	8013			
			NZA			2F	
			SXA	1		2B	
		2	HLT	8013		4B	

Assembled Output

```
0000  +80 0049 0006
0006  +56 8013 0011
0011  +40 0014 0015
0014  +51 0001 0006
0015  +01 8013 0000
```

The addresses 1, 49, and 8013 are absolute, and the other non-blank addresses are program points. You ought to study this example carefully, because it illustrates many of the things you have already learned: symbolic OP-codes, absolute addresses, and blank addresses, besides program points. Even if you are not familiar with the tape and index register commands used, you can compare the input with the output. Please do not read any farther into this manual until you have mastered this example program.

Ten program points are available, 0 through 9, and each may constantly be reused during the course of the input program. In the above example, program point 2 was used twice. The address "1F" appearing in the D- or I-address always refers to the next location 1 in your program; "3B" always means the previous location "3" in your program. Now notice: if you are in location "4," neither the address "4F" nor "4B" will refer to its own location "4," but to the next or previous location "4" respectively. A D- or I-address of merely "4" will, however, refer to its own L-address. To illustrate this, here again is Example Program 1, with all the blank addresses filled up by program points, and with the D-address of the halt-command changed:

Example Program 1-A

T P	S N	LOCATION	OP CODE	DATA ADDRESS	T	INSTR. ADDRESS	T
		1	RAA	49		2F	
		2	WTM	8013		3F	
		3	NZA	3F		1F	
		3	SXA	1		2B	
		1	HLT	1		1B	

0000 +80 0049 0006
0006 +56 8013 0011
0011 +40 0014 0015
0014 +51 0001 0006
0015 +01 0015 0000

Rules: 1. A D- or I-address is a program point if the symbolizer part is a number, the second column is a "B" or an "F," and the remaining columns are blank. "B" indicates a so-called backward forward program point, and an "F," a forward program point. 2. An L-address is a program point when the symbolizer part is a number and the remaining columns are blank.[1] This type of address is known as a "simple program point." 3. A simple program point in the D- or I-address of an instruction will be assembled as the location of that instruction, regardless of the number used or of the type of L-address.

E. Symbolic Addresses. If an address is not absolute, regional, blank, or a program point, it is called symbolic. Any combination of characters acceptable to the alphabetic or special character device is permissible, although, of course, obscenity is in poor taste.[2] Symbolic addresses are usually used when references to the same location are made from relatively distant points of the program. As an illustration of symbolic addresses, here are Example Programs 1 and 1-A rewritten so that symbolic addresses everywhere replace program points:

---

1. If a forward or backward program point occurs in the L-address it is assembled as if it were in the D- or I-address provided it is already defined; SuperSoap stops if it is undefined.

2. If you're interested, there are 255,775,968 different possible ways of specifying an address. Of these, 11,110 are absolute; 470,000 are regional; 1 is blank; 30 are program points; and 255,295,827 are symbolic. In Soap 3 these figures are 10,000; 360,000; 1; 30; and 255,405,937. In Soap 2 they are 10,000; 260,000; 1; 0; and 250,187,302 plus 5,318,665 which are meaningless.

Example Program 2

TP	SN	LOCATION	OP CODE	DATA ADDRESS	T	INSTR. ADDRESS	T
		START	RAA	49		MARK	
		MARK	WTM	8013			
			NZA			HALT	
			SXA	1		MARK	
		HALT	HLT	8013		START	

Example Program 2-A

		START	RAA	49		MARK	
		MARK	WTM	8013		TEST	
		TEST	NZA	SUB A		HALT	
		SUB A	SXA	1		MARK	
		HALT	HLT	HALT		START	

In either case the output would be the same as when the program points were used. You can see how the use of blank addresses and program points simplifies coding, and does not force you to dream up new symbolic names incessantly. The symbolic coding for the SuperSoap program itself uses a total of 138 symbols, and this number would be increased to over 530 if no program points were used.

Examples of addresses:

8000	Absolute (8000)
8000	Symbolic
000	Absolute (0000)
8	Simple Program Point
8	Absolute (0008)
8F	Forward Program Point
8 F	Symbolic
R0123	Regional
R123	Symbolic
R123	Symbolic
	Blank
.F	Symbolic
009	Symbolic
0009	Absolute (0009)
40009	Regional
/0009	Regional
9B	Backward Program Point

DATA.

Constants and data are easily assembled by SuperSoap. Numerical data are written using numeric OP-codes and absolute D- and I- addresses, as in the examples below.

>ALF: Alphabetic data of up to five characters per word may be assembled (in 650 standard double-digit representation) using the OP-code "ALF." The five characters are written in the D-address, and the I-address has no effect on assembly.[1]

Examples:

TP	SN	LOCATION	OP CODE	DATA ADDRESS	T	INSTR. ADDRESS	T
		ONE	00	1		0	
	-	C0005	31	4159		2751	
		5	ALF	SUPER		SOAP	
	-	9010	ALF	709			

1. If the I-address is blank, however, as on the fourth example, SuperSoap will fill it with information so that no blank-address-validity error will occur.

1. The number +0000010000 is loaded into symbolic location ONE. 2. The number -3141592751 is loaded into the fifth location in region C. 3. The number +8284776579, "SUPER," is placed into the location of program point 5; the I-address, "SOAP," is ignored. 4. The number -0097909900 is loaded into (core) location 9010.

Note: ALF is called a "pseudo-operation"; that is, it does not correspond to any regular 650 operation code, yet it has meaning in SuperSoap language. Pseudo-operations are the way to communicate with Soap to control assembly of your programs. Each pseudo-op will be designated by the symbol > as its function is defined in this manual. Those on which SuperSoap requires the disk storage unit while assembling are preceded by two of the signs.

AVAILABILITY. As SuperSoap begins to assemble a program, all 2000 drum locations are considered "available"; that is, any of them may be chosen as the location assigned to a symbol, program point, or blank address. As soon as a location is chosen, however, it is made unavailable so it will not be used twice.

Absolute and regional addresses are not made unavailable when encountered by SuperSoap, but they may be reserved at the beginning of assembly if desired, as described below. You can easily control the availability status of all locations to your liking with the pseudo-ops listed here.

>BLR: (Block Reservation) This pseudo-operation makes sequential locations between two limits unavailable, regardless of their former status. The address of the first location (called FWA, the first-word-address) is specified by the D-address of the BLR-card, and the last-word-address (LWA) is specified in the I-address.

Rules: 1. Both FWA and LWA equivalents must be in the range 0000-1999.[1] 2. FWA must be less than or equal to LWA. 3. If the L-address is non-blank, BLR has a special effect (see the EQU pseudo-op), so leave the L-address alone when only block-reservation is desired.

---

1. If they are both in the range 9000-9059 a special set of rules goes into effect, as described in Appendix II.

>BLA: (Block Availability) BLA is the same as BLR except that all locations from FWA to LWA are made available regardless of their previous status. The L-address of a BLA-card is automatically blanked out if you happen to fill it.

Example: If the symbol SIXTY has the equivalent 0060, and C0001 is location 0600, the two cards

T P	S N	LOCATION	OP CODE	DATA ADDRESS	T	INSTR. ADDRESS	T
			BLRS	1XTY		900	
			BLAC	0001		C0003	

in that order, make locations 0060-0599 and 0603-0900 unavailable and locations 0600, 0601, and 0602 available.

Availability may also be doctored somewhat using the pseudo-ops SYN, COR, and DRC, which will be described later.

>PAT: (Punch Availability Table) SuperSoap remembers the availability status of all drum locations in a 2000-digit table. Pseudo-op PAT causes the punching of this table in a format which is highly readable, once you get used to it (see FORMATS, Appendix VI). Available locations are listed as "8," and unavailable locations are listed as 0. The availability table may be wholly or partially loaded as input to SuperSoap, merely by placing it with the input program, thus restoring the availability which existed at some point of a prior assembly.

>>PAL: (Punch or Print Available Locations) All drum locations which are currently available are punched one-at-a-time on reloadable BLA cards (if you are in one-per-card output mode) or printed on-line (if you are in that mode; see OUTPUTS), when PAL is called for. Caution: Use PAL only when you suspect very few locations are left -- otherwise you may chew up thousands of cards. The output of PAL, when preceded by the card BLR 0 1999, will restore the availability table to its status when PAL was given.

THE DEFINITION OF DEFINITION.

To get any farther into Soap language you will need to know what it means to say an address is "defined" or "undefined." Briefly, an address is defined if its equivalent has already been determined by SuperSoap before that point in the program. The specific rules for definition are very natural, but it will be instructive to consider in detail exactly when an address is defined, and when it is not.

A symbol becomes defined after its first appearance. Within each instruction, the order of processing is L-address first, then D-address, and then the I-address. Once defined, a symbol remains so until the end of the program (or until the appearance of the pseudo-op SER described later).

A forward program point such as "2F" also becomes defined as soon as it first appears, and it remains defined until location "2" appears in the L-address of an instruction (or in the L-address of the pseudo-op UND described later). At this point "2F" becomes an undefined address immediately until used again.

A backward program point such as "2B" is defined after the first instruction in location "2" has been completely assembled. Every succeeding location "2" will redefine the meaning of "2B."

A simple program point such as "2" is always defined when it appears in the D- or I-address of an instruction. In the L-address of an instruction, a simple program point like "2" is defined only if "2F" has occurred after the beginning of assembly and after the preceding location "2."

Absolute addresses are always defined.

Blank addresses in instructions are always undefined in D- or I-addresses (unless both D- and I-addresses are blank, in which case the I-address is defined), and they are always defined in L-addresses.

Regions are defined by EQU, SYN, or BLR cards as described later; once a region is defined, all regional addresses corresponding to that region remain defined until the end of assembly.

DEFINING ADDRESSES WITH PSEUDO OPERATIONS.

A symbolic, program point, or regional address may be given any desired equivalent by means of the pseudo operations EQU, SYN, and BLR.

>EQU: (Equivalent) The L-address is set equivalent to the D-address.

Rules: 1. The D-address must be defined. If the D-address is blank it is the same as absolute address 0000. 2. The L-address may not be absolute or blank, but any other type of address is legal, as shown in the examples below. 3. The L-address may be either defined or undefined when the EQU card is given. 4. The I-address must be blank; operation of EQU and SYN with non-blank I-address is described in Appendix III.

>SYN: (Synonymous) Operation of SYN is exactly like EQU, except the equivalent is also made unavailable without regard to its former status. This equivalent must be in the range 0000-1999 (or 9000-9059; see Appendix II).[1]

Examples:

	TP	SN	LOCATION	OP CODE	DATA ADDRESS	T	INSTR. ADDRESS	T
1			UPPER	EQU	8003			
2			START	SYN	1999			
3			3F	EQU	1B			
4			8B	EQU	R0001			
5			8	EQU	R0001			
6			X0005	EQU	4			

(1) The symbol UPPER is defined to have the equivalent 8003. Notice that SYN could not be used. (2) The symbol START is defined to mean 1999, and this location is made unavailable. (3) Program point 3F is given the same equivalent as 1B now has. (4) Program point 8B is given the equivalent of R0001.

1. A peculiar by-product of the manner SYN is handled in SuperSoap is that if the L-address is blank, the D-address is reserved and the EQU phase is omitted. In other words, the card SYN FWA has the same effect as BLR FWA FWA; in fact, the former is processed more speedily since only one address need be decoded.

(5) This card has the same effect as example (4); in general, no distinction is made between simple and backward program points in the L-address of an EQU, SYN, or BLR card. (6) Region X is defined by this card, in such a way that X0005 has the equivalent 0004. Any of the three cards

```
 X 0006 EQU 5
 X 0000 EQU 9999
 X 0001 EQU
```

would have the same effect.

>BLR, L-address non-blank: The single card

```
L    BLR FWA  LWA
```

is equivalent to the two cards

```
L    EQU FWA
     BLR FWA  LWA.
```

This feature is expecially useful for defining a region; for example,

```
P0001 BLR  1977  1986
```

defines region P as the 1977 punch band and reserves the ten locations 1977-1986. The card

```
P0001 BLR  1977 P0010
```

has the same effect as the previous card, due to the order in which region definition and block-reservation take place. Thus you can either specify FWA and LWA when defining a region, or you can specify FWA and the number of words in the region. Notice that if you wished to reserve P0000 also, you could have coded P0000 BLR 1976 P0010.

The only way a region can be defined is by using EQU, SYN, or BLR. Remember that you must define a region before using any regional addresses from that region.

OUTPUT.

The SuperSoap programmer has his choice of 128 outputs. Sounds awesome, doesn't it? Actually, there are seven different kinds of output, and you can

turn these off and on independently of each other. Three of these will be discussed here, and others (which are, by the way, extremely interesting) are given in Appendix V.

>ONE: This pseudo-op calls out normal output, on which the input card is reproduced in columns 41-72,[1] and the assembled instruction is immediately reloadable as a single instruction load card. This one-per-card output may also be used again as SuperSoap input.[2]

>PRT: The same information given when listing the one-per-card output will be printed on-line with output synchronizer 2, once PRT-mode is set in operation.
>FIV: Five-per-card output, which is punched only once for every five assembled instructions, can also be obtained from SuperSoap. The format is given in Appendix VI.

Each of these modes is independent of the others. SuperSoap is initially set for normal output and no more; to start up any one of these outputs, give the pseudo-op with the D-address blank. To stop any of these outputs, give the appropriate pseudo-op with the word "OFF" in the D-address.[3]

Example: To punch only five-per-card output, and list the one-per-card output on-line, give the three cards

```
ONE OFF
PRT
FIV
```

>>PLR: (Punch Loading Routine) A two-card loading routine which will pull in the five-per-card output is punched to fit into any of the forty bands on the drum. This loading routine is listed in Appendix VI; it uses locations 01-24 of the band you specify, and these locations are automatically made unavailable

---

1. With the occasional exception of the L-address of BLA, and I-address of ALF, and certain unpreferred OP-codes.
2. All punched output is done with synchronizer 1, but the ONE or PRT output is easily changed to a different synchronizer since they are done from core.
3. I will let you in on a little secret: any time the D-address is non-blank, SuperSoap is satisfied that you have punched the word OFF. This is for those of you who are poor spellers.

by the PLR card. Of course, it is most convenient to use PLR just before starting to punch five-per-card output.

Rules: 1. The L-address of the PLR card must have an equivalent in the range 0000-1999 or else a ridiculous loading routine will be produced. The loading routine which appears as output will go into the same band as this address. 2. If the L-address is blank, the 1950 band will be used for the loading routine. 3. PLR also has the effect of FIV, so the D-address should either be blank or the word OFF.

To transfer out of the loading routine, your last instruction assembled in five-per-card mode should go into the twentieth word of your loading routine band (e.g., 1970, 1220, etc.) and it will be the first instruction executed after the deck is all in.

Example: The card 1234 PLR will punch a loading routine to load five-per-card SuperSoap output; the loading routine will operate from locations 1201-1224 and these locations are made unavailable; and SuperSoap goes into five-per-card mode.

>CC1, CC5: (Card Count One, Card Count Five) The card count on the output specified (ONE-PRT or FIV output, respectively) will be set to the equivalent of the D-address of the CC1 or CC5 card, starting with the following card.

THE SYMBOL TABLE.

SuperSoap has table space for a maximum of 283 symbols. This magic number was not selected merely because it is prime, but because it gives greatest assembly speed and convenience. More than this number will rarely, if ever, be necessary; but if the symbol table does fill it is possible to punch the symbol table, erase the entire table, and then reload those symbols which will be needed to finish the remainder of the program.

>PST: (Punch or Print Symbol Table) PST causes the entire list of symbols to be displayed, in the order in which they were defined. If you are in ONE mode, it is punched; if you are PRT mode, it is listed on-line; if you are in neither mode, you don't get the table at all (sorry). The symbol table is presented on

EQU cards which are reloadable. PST is convenient for you when getting the bugs out of a new program: keypunching errors become obvious when unfamiliar symbols appear; the order in which symbols were defined is helpful since it sometimes causes unforeseen overlays; and the locations assigned to each symbol are easily located from the listing.

>SER: (Symbol Erase) This pseudo-op causes SuperSoap to forget that any symbols were ever defined in the preceding assembly.

MISCELLANEOUS THINGS YOU SHOULD KNOW.

Type 1 Card: If you wax so eloquent as to need more than ten columns for comments, a type 1 card may be used to gain thirty such columns. SuperSoap will do nothing with card type 1, or a card with blank OP-code, except reproduce it in ONE or PRT mode.

>>FIL: You may often want to put the same instruction or constant into a block of consecutive locations. FIL saves you the trouble of punching a card for each one, and also cuts assembly time considerably.

First you give a card which says FIL FWA LWA, where FWA and LWA are the first and last of the consecutive locations you want to fill. The next card following FIL contains the instruction which you intend to be put into all the consecutive locations. This card must have blank L-address, and the rest of it should be in the same form as an ordinary SuperSoap instruction.

If you are in ONE mode, FIL punches a single card which will fill the locations as you desired. This card works with the ordinary one-per-card loading routine, and it need not be removed if you reassemble the one-per-card output.

If in FIV mode, FIL punches a single card which will temporarily interrupt the loading routine punched by PLR. In this case it is important to specify which drum band you are using for the loading routine, so this is specified in the L-address of the FIL card, exactly as you specified it on the PLR card.

Examples: The two cards

T P	S N	LOCATION	OP CODE	DATA ADDRESS	T	INSTR. ADDRESS	T
			FIL	9000		9059	
	-		ALF				

will produce a core-zero card (it sets all core storage locations to minus zero). If you are in FIV mode, the output card will be for the 1950 band.

The two cards

	1111	FIL	F0000		F0009	
		01	2345		6789	

will load the constant +0123456789 into locations F0000 through F0009; the five-per-card loading routine should use the 1100 band.

Rules: 1. FWA and LWA must both be in the range 0000-1999 or both in the range 9000-9059. 2. The L-address of the card following FIL must be blank. 3. FWA must be less than or equal to LWA.

>>BOP: (Beginning of Program) When assembling several programs at once, a BOP card placed between programs will reinitialize SuperSoap and ready it for another assembly. A BOP card will not appear in the output.

>END: This pseudo op must be the last card of every SuperSoap input program unless a BOP card follows. Furthermore, END must not be used before the very last card of an assembly deck, for it finishes assembly and unconditionally stops the computer. In other words, you must use END exactly once in your assembly deck, and this spot is at the END.

Example: Suppose you are SuperSoaping two programs; your assembly deck should have this order:

1. Input for Program 1
2. BOP card
3. Input for Program 2
4. END card

Optimization: Due to the one-pass nature of assembly, priority for the choice optimum drum locations diminishes as assembly progresses. Therefore frequently

executed portions of the program should be placed toward the beginning of the assembly deck. Soap optimization runs roughly two to three times faster than sequential programming, but it can never compare with the hand optimization you fit together yourself. Hand optimization can give you another factor of three for drum-located routines, and such optimization is readily done using special SuperSoap features described in Appendix IV.

RUNNING THE PROGRAM (A Recipe).

1. Insert SuperSoap control panel on 533 unit, and on printer if required.

2. Place SuperSoap Calling Card in read feed, followed by your assembly deck. Be sure the last card of your deck says "END."

3. Ready punch hopper with blanks if you are getting any card output. Make sure disk file unit is on and the three arms are on. Make sure the arm control switch is set to "P." Ready tape units and printer if you are going to use them while assembling.

4. Set 650 console as follows: programmed STOP, half cycle RUN, control RUN, display DISTRIBUTOR, overflow SENSE, error STOP, storage entry switches 70 1951 8282+.

5. Press PROGRAM RESET. (While running SuperSoap it is a good general rule to refrain from ever using COMPUTER RESET.) Press PROGRAM START. Press START on the 533 unit.

6. Now the machine should be computing merrily and it shouldn't stop until your assembly deck is finished. But SuperSoap believes that the best place to catch errors is during assembly, and so it will stop if it finds something amiss.

Stops. In each of these stops the offending card is the fourth-last card out if you clear the read feed, assuming you are not on END OF FILE. The procedure to follow in case of a stop depends on the type of error, given by the address lights:

0111. Perhaps a) You are in HMO and forgot to give one of the equivalents for an undefined address. The Program Register will contain 0X000d 0111; d is 1 for LH-field, 2 for DH-field, and 3 for IH-field. Cure: Fix card by punching in the proper equiva-

lent, then replace in read feed and hit Program Start; or, put desired drum level on 8000 switches and manually transfer to location 0234 WITHOUT HITTING COMPUTER RESET. (Use Program Reset please.)

or else b) The symbol table is full; you are trying to define the 284th symbol. Chances are your best bet is to change a few symbols to program points and then to start over; you won't be that many symbols over. Or you can restart in case of emergency as follows: Take cards from read hopper (save them!), clear read feed; add the four last cards to the pile from the read hopper; insert cards "PST" and "SER" in read feed, preceded by ONE if you are not in ONE mode; hit Program Start and read them in; reload those symbols you will need for the remainder of assembly; then finish with your assembly deck as before.

0123. Meaningless location (such as 8497 or 9060) for assembled instruction; you are probably in COR or DRC mode and have gone too far. Hit Program Start and zero will be filled in as the location; or if you want to change the card and reprocess it, set storage entry switches to 00 1951 0111, push Program Reset, and push Program Start to read in the correction card.

0222. Perhaps a) You are in HMO or MMO, and there are no locations available on the drum level you specified. Program Register contains 01 1ddd 0222 where ddd is four times the level which is filled. Either 1) change the card and reprocess using Program Start, or 2) put desired drum level on 8000 switches and manually transfer to location 0234 WITHOUT HITTING COMPUTER RESET. (Use Program Reset please.)

or else b) There are no more drum locations available. To restart, insert a BLA card in the deck and hit Program Start.

0300. You are in the midst of some pseudo-op which is processed from a special disk-storage routine (such as PLR, FIL, etc.) and one of the addresses on your card is undefined. (The D-address of the

Program Register indicates which one: 1951 or 1696, L-address; 1953, D-address; and 1952, I-address.) Another card, the card following FIL or TAP for example, may actually have been processed before this particular error was detected, so the offending card may be the fifth-last card out instead of fourth-last. The only way to restart is to put the desired equivalent in the I-address of the storage entry switches and hit Program Start; the storage entry switches will be used as a replacement for the incorrect address on your card.

0333. Undefined Symbolic OP-code. Correct the card and hit Program Start.

0444. Undefined regional address. The D-address of the Program Register displays which region you used is undefined, in 650 double digit representation; for example, if region S is undefined, the Program Register would contain 01 0082 0444. To restart, insert an EQU or BLR card defining the region in question, and push Program Start. Or you may go into a special console restart procedure: Put 0300 in the Address switches, set Control to Manual, hit Program Reset (not Computer Reset), hit Transfer, set the I-address of the storage entry switches to the address equivalent you wish to replace the undefined address, turn Control back to Run, and hit Program Start. This does not define the address which was undefined, but does keep the assembly going as if you had punched an absolute address instead of the one which was undefined.

0555. Undefined symbol or program point used on pseudo-op card where a defined equivalent should have been specified. The D-address of the Program Register displays which address was askew: 1 for L-address, 2 for D-address, and 3 for I-address. To restart, either change the offending card and hit Program Start, or use the special console restart described under stop 0444.

0666. L-address of instruction is blank when previous card had no blank D- or I-address. To restart, correct the bad card by punching in

a meaningful L-address, reinsert it in the deck, and hit Program Start. Or, you may restart at the console by setting the storage entry switches to 00 1951 0069+ and depressing Program Reset, Program Start.

0777. L-address of instruction is non-blank when previous card had either a blank D- or I-address. To restart, you have three alternatives, depending on the problem you have:

a) You meant for the L-address to be blank. In this case, simply depress Program Start.

b) You like the L-address as it is. In this case, set the storage entry switches to 00 1951 0079+, hit Program Reset (not Computer Reset) and Program Start.

c) You want to change the card now being processed. Set the storage entry swtiches to 00 1951 0111+ and hit Program Reset, Program Start.

0888. Either a) Undefined backward program point; or

b) EQU, SYN with L-address absolute; or

c) EQU with L-address blank; or

d) Undefined forward program point in L-address of instruction.

In case of a program point error, (a) or (d), the D-address of the Program Register contains 1000 plus the program-point number. To restart, correct the mistaken card and depress Program Start. Or you may fix errors (a) and (d) by the special console restart procedure described under stop 0444.

0999. Improper BLR, BLA, or SYN card. (Either FWA and LWA are in wrong range, or FWA > LWA.) To restart you must fix the card, replace it in the read feed, and depress Program Start.

1996. Normal stop caused by OPT. (See writeup of extension you are using.) Number is displayed in lower accumulator and distributor. Set storage entry minus to skip cards, plus to keep cards.

8000. For some reason you have the Op-code RPQ. Hope you are enjoying yourself.

8282. Wrong 533 plugboard

9911. Four unsuccessful attempts to write on tape. Start over with a good tape or a good machine.

9898. Successful end of assembly; caused by END card.

Important. The stops above may be divided into three categories: The first class consists of the 0111, 0222, 0333, 0444, 0555, 0666, 0888, and 0999 stops; on these stops, depressing Program Start resets SuperSoap to the state it was in before the offending card was sensed, then makes ready to read a new card as though nothing wrong had happened. (Of these, stops 0111, 0222, 0444, and 0888 all lead to an elaborate error correction procedure which is brought in from disk storage.) Stops 0123, 0300, 0777, and 1996 call for making a setting of the console switches and depressing Program Start; they do not lead to a read command that calls for a correction card. Finally, stops 8000, 8282, 9911, and 9898 are not meant to be restarted.

Machine Errors. In case of a machine error, you will be able to restart assembly unless you are in the midst of one of the pseudo-ops which is pulled in from disks, or unless the error has occurred in a most unlikely place. To restart after a machine error, run the cards out of the read feed and replace them again beginning with the card which was being processed when the error occurred (the fourth-last one out). Then set the storage entry switches to 00 1951 0111+, hit Computer Reset to clear out the machine error (the only time Computer Reset is appropriate!) and hit Program Start.

Manual Effects. You may duplicate PST manually by transferring to location 0800; PAT, by going to 0900; and BOP, by sending control to 1000.

EXAMPLE PROGRAM.

To summarize some of what has gone before, here is a simple program which will prepare a table of $F(x) = A x^2 + B x + C$ for $x = 1, 2, \ldots, 100$ assuming A, B, and C are integers and $|F(x)| < 10^{10}$. The output is to contain x in word 1, F(x) in word 2. Input as written on a SOAP coding form is shown below followed by the assembled output.

41	42	43	48	51	56	57	62	63
T P	S N	LOCATION	OP CODE	DATA ADDRESS	T	INSTR. ADDRESS	T	REMARKS
1		EXAMPLE	LE	CALCULATE	C	ULATE		F OF X
			BLR	1951		1960		READ AREA
		P0001	BLR	27		P0002		PUNCH AREA
		START	RAU	ONE		1F		SET X
		1	STU	P0001				TO 1
			MPY	A				CALCULATE
			ALO	B				F
			RAU	$0002				
			MPY	P0001				
			ALO	C				
			STL	P0002				
			WR1	P0001				PUNCH
			RAU	P0001				
			SUP	XMAX				IS X MAX
			NZU			9876		
			AUP	101		1B		INCREASE X
		ONE	00	0		1		CONSTANTS
		XMAX	00	0		100		
		101	00	0		101		
			END					

```
0001  1       EXAMPLE   CALCULATE F OF X
0002
0003              BLR    1951     1960    READ AREA
0004        P0001 BLR    27       P0002   PUNCH AREA
0005        START RAU    ONE      1F      SET X          0004  +60 0007 0011
0006        1     STU    P0001            TO 1           0011  +21 0027 0030
0007              MPY    A                CALCULATE      0030  +19 0033 0003
0008              ALO    B                F              0003  +15 0006 0061
0009              RAU    8002                            0061  +60 8002 0019
0010              MPY    P0001                           0019  +19 0027 0047
0011              ALO    C                               0047  +15 0000 0005
0012              STL    P0002                           0005  +20 0028 0031
0013              WR1    P0001            PUNCH          0031  +71 0027 0037
0014              RAU    P0001                           0037  +60 0027 0081
0015              SUP    XMAX             IS X MAX       0081  +11 0034 0039
0016              NZU                9876                0039  +44 0043 9876
0017              AUP    101      1B      INCREASE X     0043  +10 0046 0011
0018
0019        ONE     00   0        1       CONSTANTS      0007  +00 0000 0001
0020        XMAX    00   0        100                     0034  +00 0000 0100
0021        101     00   0        101                     0046  +00 0000 0101
0022              END
```

APPENDIX I

SYMBOLIC OPERATION CODES.

650-SOAP--English	
00	NOP* No operation
01	HLT Halt
02	UFA Unnormalized Floating Add
03	RTC Read Tape Check
04	RTN Read Tape Numeric
05	RTA Read Tape Alphabetic
06	WTN Write Tape Numeric
07	WTA Write Tape Alphabetic
08	LIB Load IAS Block (Core)
09	LDI Load IAS
10	AUP Add Upper
11	SUP Subtract Upper
12	Not Used
13	Not Used
14	DIV Divide
15	ALO* Add Lower
16	SLO* Subtract Lower
17	AML Add Magnitude to Lower
18	SML Subtract Magnitude Lower
19	MPY Multiply
20	STL Store Lower
21	STU Store Upper
22	SDA Store Data Address
23	SIA Store Instruction Address
24	STD Store Distributor
25	NTS No Tape Signal
26	BIN Branch Inquiry
27	SET Set Buffer Ring
28	SIB Store IAS Block
29	STI Store IAS
30	SRT Shift Right
31	SRD Shift and Round
32	FAD Floating Add
33	FSB Floating Subtract
34	FDV Floating Divide
35	SLT Shift Left
36	SCT Shift and Count
37	FAM Floating Add Magnitude
38	FSM Floating Subtract Mag.
39	FMP Floating Multiply

650-SOAP--English	
40	NZA Non-zero Index Register A
41	BMA Branch Minus IR A
42	NZB Non-zero Index Register B
43	BMB Branch Minus IR B
44	NZU Non-zero Upper
45	NZE Non-zero Accumulator
46	BMI Branch Minus
47	BOV* Branch on Overflow
48	NZC Non-zero Index Register C
49	BMC Branch Minus IR C
50	AXA Add to IR A
51	SXA Subtract from IR A
52	AXB Add to IR B
53	SXB Subtract from IR B
54	NEF No End of File
55	RWT, RWD Rewind
56	WTM Write Tape Mark
57	BST Backspace Tape
58	AXC Add to IR C
59	SXC Subtract from IR C
60	RAU Reset Add Upper
61	RSU Reset Subtract Upper
62	Not Used
63	TLE Table Lookup on Equal
64	DVR Divide and Reset
65	RAL Reset Add Lower
66	RSL Reset Subtract Lower
67	RAM Reset Add Magnitude
68	RSM Reset Subtract Magnitude
69	LOD*, LDD Load Distributor
70	RD1,RCD Read Card (Input 1)
71	WR1,PCH Punch (Output 1)
72	RC1 Read Conditional 1
73	RD2 Read Input Area 2
74	WR2 Write Output Area 2
75	RC2 Read Conditional 2
76	RD3 Read Input Area 3
77	WR3 Write Output Area 3
78	RC3 Read Conditional 3
79	RPY Reply

A P P E N D I X I (Continued)

650-SOAP--English		650-SOAP--English	
80 RAA	Reset Add to IR A	90 BD0+	Branch Distributor 10
81 RSA	Reset Subtract from IR A	91 BD1	Branch Distributor 1
82 RAB	Reset Add to IR B	92 BD2	Branch Distributor 2
83 RSB	Reset Subtract from IR B	93 BD3	Branch Distributor 3
84 TLU	Table LookUp	94 BD4	Branch Distributor 4
85 SDS	Seek Disk Storage	95 BD5	Branch Distributor 5
86 RDS	Read Disk Storage	96 BD6	Branch Distributor 6
87 WDS	Write Disk Storage	97 BD7	Branch Distributor 7
88 RAC	Reset Add to IR C	98 BD8	Branch Distributor 8
89 RSC	Reset Subtract from IR C	99 BD9	Branch Distributor 9

* O's must be alphabetic O's, not numeric zeroes.
+ Either O or zero is acceptable.

When two Op-codes are listed, the first is preferred for faster assembly.

A P P E N D I X I (Continued)

Summary of Pseudo Operations.

Sign	L addr.	OP	D addr.	I addr.	Page of Manual
Sign	i	ADN	D	I	37
Sign	L	ALF	XXXXX	i	11
i	b	BLA	FWA	LWA	13
i	L	BLR	FWA	LWA	12,16
i	i	BOP	i	i	20
i	i	CC1	D	i	18
i	i	CC5	D	i	18
i	FDA	CDD	D	I	47
i	L	COD	D	i	39
i	i	COF	i	i	44
i	i	CON	i	i	44
i	i	COR	FWA	i	32
i	ρ	DEK	i	i	45
i	i	DMO	i	i	42
i	i	DOF	i	i	44
i	i	DON	i	i	44
i	L	DRC	FWA	i	32
i	FDA	DSK	D	I	47
i	i	DUP	SYMBL	i	40
i	i	END	i	i	20
Sign	L	EQU	D	I	15,35
i	ρ	FIL	FWA	LWA	19
i	i	FIV	β	i	17
i	i	HED	ωiiii	i	38
i	i	HMO	i	i	41
i	i	LAT	FDA	i	36
i	i	LST	FDA	i	37
i	i	MMO	i	i	41
i	80ln	MRK	i	i	46
i	i	NMO	i	i	41
Sign	i	NXT	D	I	42
i	i	ONE	β	i	17
i	i	OPT	D	i	40
i	i	PAL	i	i	13
i	i	PAT	i	i	13
i	i	PDL	i	i	45
i	ρ	PLR	β	i	17
i	i	PRT	β	i	17
i	i	PST	i	i	18
i	i	PUD	SYMBL	i	40
i	i	SAT	FDA	i	36
i	i	SCR	ωiiii	i	38
i	i	SER	i	i	19
i	80ln	SKP	D	I	46
i	i	SST	FDA	i	36
Sign	L	SYN	D	I	15,35
Sign	80ln	TAP	D	I	46
i	miiii	UND	i	i	36
i	i	OOA	i	i	FIL card bypassed

A P P E N D I X I (Continued)

Key: i: ignored by SuperSoap but reproduced on output; b: ignored by SuperSoap and blank on output; L, D, I: proper address as determined by rules of the pseudo op; FWA, LWA: first or last word address; n: tape unit desired; m: program point desired; ω : heading or scrambling character; β : either blank or the word OFF; ρ : specification of drum band for loading routine; FDA: First Disk Address (disk and track number).

APPENDIX II

USING THE INDEX REGISTERS AND CORE.

Indexing. When the D- or I- address is to be modified by an indexing register, the corresponding TAG column (56 or 62) is punched A, B, C to indicate the appropriate register.[1] SuperSoap will automatically increment tagged addresses by the proper multiple of 200 if the address equivalent is in the range 9000-9099,[2] or by the proper multiple of 2000 if the equivalent is in the range 0000-1999. Other addresses, such as 2000, are not tagged even though A, B, or C is punched. The tags are effective also on pseudo-op cards; e.g., the card "SYMBL EQU 1951B" gives SYMBL the equivalent 5951.

Core Mode.

>COR: This pseudo-op causes SOAP to stop optimum programming and to begin programming sequentially. FWA is punched in the D-address of the COR card; from then on, every undefined address is given a sequential location, starting with FWA, and defined addresses are assembled as usual.[3] COR mode is terminated by a COR or DRC card with D-address blank.

>DRC: (Drum-Core Mode) DRC is used to program routines which are stored on the drum but will be block-transferred with LDI and executed up in core storage. It has two functions:

1) DRC sets up sequential assembly starting at the D-address just as with COR. Sequential assembly is stopped by COR or DRC with Blank D-address.

2) The L-address of the DRC card, which we will call DRUM, specifies the drum location which will be moved to FWA when executed.[4] While DRC is in operation, all L-addresses which are in the range 9000-9059 are translated

---

1. Tags of J, K, L, S, T, 1,2, 3, and / also cause indexing, so they should never appear in TAG columns.
2. Notice that the address 9070 can be tagged since the 650 will correctly process an address such as 9470 when running.
3. COR affects only instructions, not pseudo operations except ALF. FWA need not be a core location, but if it is a drum location the sequential equivalents assigned are not reserved. A 0123 stop will occur if a meaningless L-address equivalent, such as 9060, is encountered.
4. Actually DRUM could be a core location . . . if you want it to be and you know what you're doing.

A P P E N D I X II (continued)

by the amount (FWA - DRUM), while L-addresses on the drum are not affected. This translation of core addresses is terminated only by any COR card, unless superceded by another DRC card with non-blank D-address. On a DRC card which has D-address blank, the L-address is ignored, and the translation of core addresses specified by the preceding DRC card will remain in effect. Caution: Remember that equivalents assigned to symbols and program points in the L-address during DRC mode are the core equivalents, not the drum equivalents, so references to these locations should be made only when they have been moved up to core. DRC mode does not change the locations filled by FIL.

Drum tags. When in sequential COR or DRC mode, the sequential assembly can be suspended momentarily by tagging an undefined address with the letter D.[1] This tag will cause that address to be defined on the drum as if sequential COR mode were not in operation. This is extremely useful for linkages out of COR or DRC assembly, or when a routine is just a bit too long to fit in core and several instructions in the middle are to be left on the drum.

When you are not in sequential assembly mode, and if you tag an undefined address with a "D," SuperSoap will enter COR mode momentarily and will assign a sequential location to the undefined address. This address will be the next sequential location after the last one assigned during the preceding COR mode; if no COR cards preceded, it will begin assigning locations at 9000.

Block Reservation In the Core. A limited type of availability table is kept for the core, according to the following scheme: When in sequential assembly and an undefined address comes along, the next sequential location is calculated; if it happens to be in the range 9000-9059, the core availability table is interrogated. Then if the locations is listed as unavailable, it is bypassed and the next sequential location is tried until an available or a non-core location is found.

---

1. M, U, or 4 will have the same effect.

A P P E N D I X II (continued)

Core locations are made available or unavailable only on BLR, BLA, or SYN cards in the usual format. Important: Every COR or DRC card with non-blank D-address has the additional effect of BLA 9000 9059, so block reservation in the core is a one-shot deal, and it is nullified by every COR or DRC card that comes along.

EXAMPLE PROGRAM 4.

Careful study of this program, which is designed to illustrate everything discussed in this Appendix, should demonstrate the use of all these features. Experience in coding with them will reveal their flexibility. Many other examples of their use may be found in the listing of the SuperSoap program. The following example does nothing but zero out locations 0300-1999; routine begins at location 0000. It is not the simplest way to do the job, but it was designed to be an example of the SuperSoap features, not an example of shrewd coding.

```
        BLR   300      1999    RESERVE
START   SYN                    START AT 0
X0001   BLR   260      X0009   REGION X

START   SET   9000             BLOCK TO          0000  +27 9000 0005
        LIB   X0001    9004    CORE              0005  +08 0260 9004
X0001   DRC   9000
        BLR   9004     9004
 9004   RSA   49               INITIALIZE  9004  0264  +81 0049 9000
        RAU   8001     1F      CLEAR 8002  9000  0260  +60 8001 9001
1       STL   9059A            ZERO FIFTY  9001  0261  +20 9259 9002
        NZA           1F       WORDS OF    9002  0262  +40 9003 9005
        AXA   1        1B      CORE        9003  0263  +50 0001 9001
1       RAC   1650     1F      INITIALIZE  9005  0265  +86 1650 9006
1       SET   9010           D  BLOCK ZERO  9006  0266  +27 9010 0015
        STI   300 C            ONTO DRUM           0015  +29 6300 9007
        NZC           1F   D                9007  0267  +48 9008 0006
        SXC   50       1B      LOOP        9008  0268  +59 0050 9006
        DRC             COR    MODE OFFF
1       RAL   1F   D                               0006  +65 9009 0013
        HLT   START    START        STOP           0013  +01 0000 0000
1        01    2345     6789                9009  0269  +01 2345 6789
        END
```

APPENDIX III

FOR "ADVANCED" PROGRAMMERS.

Here are a few added attractions of SuperSoap for the Hardened Man of Experience Who Thinks For Himself and wants a Smoother, Firm, and Fully-Packed Assembly System.

800X Instructions and Type 2 Cards. Instructions which take place in locations 8000-8007 may be optimized by using the address as an absolute location address. Instructions which take place in erasable locations but do not need to be loaded into these locations may be optimized by placing them on Type 2 cards. Type 2 cards are treated just as ordinary cards except that they, along with 800X instructions, are not loaded into the location specified when the output deck is loaded. (Word 1 on ONE output cards is 69 1954 1953; on 800X and Type 2 Cards it is 69 1954 8000. In other forms of output the instructions assembled from 800X or Type 2 Cards do not appear at all.)

Other Card Types: Card Types 5 through 9 and Type 0 are processed exactly as if the type were blank. This enables certain cards to be identified easily in the output deck, or may be used to signal the printer to skip to another page, etc. In the SuperSoap program all disk storage constants are placed on type 7 cards so that the disk storage allocation could be changed if desired. Type 4 cards are treated the same as Type 2 cards.

>EQU, SYN extended: The form of an EQU or SYN card can be generalized as follows:

± L EQU D I

The equivalent of L is defined to be the equivalent of D plus or minus (acc. to the sign) the equivalent of I. The result is to be a positive four digit number, obtained uniquely by adding or subtracting 10000 until the result is in the range 0000-9999. When the D- or I-address is blank, it is treated as if it were zero.

APPENDIX III (continued)

Examples: (Absolute addresses are given in the D- and I- addresses of these examples, but other defined addresses may be used.)

	TP	SN	LOCATION	OP CODE	DATA ADDRESS	T	INSTR. ADDRESS	T
1			X0007	EQU	500		7	
2		-	BINGO	SYN	1500		730	
3		-	1F	EQU			852	
4				SYN	1001	C	20	A

Explanation: (1) same as X0000 EQU 500. (2) The equivalent of symbol BINGO is set to 0770, and location 0770 is made unavailable. (3) Program point 1F is defined to be 9148 (the complement of 852). (4) This is certainly the trickiest example and also the least useful: Location 9021 is made unavailable in the core availability table (since 7001 plus 2020 is 9021); no address is defined because the L-address is blank.

>UND: This pseudo-op undefines forward program points so they will be available for reuse later. The program point to be undefined is punched in the L-address; only column 43 is interrogated, and it must be numeric. Example : 1 UND causes program point 1F to be undefined.

>>LAT, SAT: (Load Availability Table, Store Availability Table) These pseudo operations may be used to store and restore different availability tables for use at different points of your assembly. Each availability table is kept on four disk tracks; if you are using SAT you must use four consecutive tracks which are not being used for another purpose. You may not use tracks 0000-0099 since SuperSoap uses them as temporary storage. If you don't know of any others, tracks 8838-8879 are always available to use for this purpose, and they are located so that seek time is kept short. The D-address of LAT and SAT is used to specify the first track of the four consecutive tracks to be used. For example, SAT 8838 will store the availability table on tracks 8838, 8839, 8840, and 8841.

APPENDIX III (continued)

>>LST, SST: (Load Symbol Table, Store Symbol Table) These pseudo-ops are completely analogous to LAT and SAT except the symbol table requires nine consecutive disk tracks. For example, the card LST 8841 will bring in the symbol table which was previously stored on disk tracks 8841 through 8849. Never use LST unless SST has preceded it.

Addends.

>ADN: (Add to Next instruction) This feature enables you to do arithmetic directly on instructions. The equivalent of the I-address of the ADN card is placed in the low order four digits and shifted left the amount specified by the D-address;[1] the result is added to or subtracted from the next instruction or ALF card that comes along, according to the sign of the ADN card. ADN may be compounded; that is, two or more ADN cards may all affect the same instruction.

Examples of ADN: 1) Symbolic addressing for disk storage addresses. The two cards

			ADN	1	SYMBL
			00	0000	0002

will prepare a disk constant for the four-digit disk track address specified by SYMBL, for use with arm 2. For instance, if SYMBL had equivalent 2503, you would obtain the constant 0000025032.

Or you may wish to change the arm number of all disk addresses in a program. At the beginning you could define "ARM" to be 0, 1, or 2, and then precede every disk storage constant with the card "ADN ARM" (a blank D-address indicates no shift). Reassembly with the single card defining ARM changed will change all the disk arm addresses.

---

1. The rule for the amount of shift is the same as the 650's rule: thus a D-address of 10 is the same as 0; 9016 is the same as 6; and 8222 would hang up.

A P P E N D I X I I I (continued)

2) Compound adding and subtracting. The cards

```
          ADN 9    9485
-         ADN 4    1227C
          ADN 6    10
-         ADN      5
-X0004    RAL 0    0
```

will cause the number

```
+ 50 0000 0000   (other digits are lost)
- 00 1227 0000
+ 00 1000 0000
- 00 0000 0005
- 65 0000 0000
- - 15 6227 0005   to be loaded into location X0004.
```

Heading.

>HED: Heading is used to avoid doubling up on symbols when several programs or several sections of a single program are to be assembled together. The need for heading is paramount if several persons have contributed to a program. Heading is accomplished in SuperSoap by the automatic insertion of a heading character into the right-most position of symbolic addresses which have this position blank. (When the right-most column is non-blank, the symbol is not headed.) The heading character is punched in the symbolizer part of the D-address of a HED card (it may be any character acceptable to the alphabetic device or special character attachment) and all other columns are ignored. This heading is applied to every symbol thereafter until another HED or SCR card appears or the program ends. To "turn off" the heading, a HED or SCR card with no punch in column 51 may be used. If within the section headed by "A" it is desired to refer to the symbol "TAXbb" (b indicating a blank column) which appears in a section headed by "F", the symbol "TAXbF" should be used.

>SCR: (Scramble) SCR is a stronger form of heading. It operates like HED except that it is completely impossible to refer to the symbols which were headed under a SCR card unless you are under a similar SCR card. The Scramble Character, punched in column 51, may be any punch acceptable to the alphabetic

A P P E N D I X I I I (continued)

or special character device except a period, dollar sign, asterisk, diagonal, comma, per cent, left parenthesis, A, B, I, J, K, and S. Numbers are recommended as Scramble Characters.[1] Thus, in a section of program not scrambled by "F", it is not possible to refer to the symbol "TAXbb" which was scrambled by "F."

Transfer Out Card. You easily transfer out of SuperSoap into direct machine instructions executed at assembly time if you are reasonably familiar with the SuperSoap program. (I repeat, if you are reasonably familiar with the SuperSoap program.) A card with a 12-punch in column 41, and with any negative 10-digit number punched in columns 1-10, will be read as a load card into the 1950 read band and will transfer control to location 1954. Such a card may be used to alter some phase of assembly slightly. You will generally exit to the read command, that old standby RCD 1999 1998, which is kept in location 0999.

Core-Drum Mode.

>COD: Occasionally you will load routines into core and execute them on the drum; e.g., routines brought in from disk storage or tape. Core-Drum Mode translates drum L-addresses in a way similar to DRC, which relocates core L-addresses. The two modes work independently of each other: one applies only to core locations and the other only to drum locations. COD does not set up sequential assembly. Core-Drum Mode is terminated by a COD card with blank D-address. Example: 9000 COD 1024 sets up Core-Drum Mode so that an instruction assembled for the location 1024 would be loaded into location 9000. 9001 COD 1025 would be equivalent. COD mode does not affect the locations filled by FIL.

Library Routines. Several pseudo-ops have been included for writers of library routines which are to be Soaped with the programs of others.

---

1. No doubt you wonder where this list comes from. SCR subtracts the character while HED adds it, and thus the permissible SCR characters are those whose complement is not defined in double-digit representation.

A P P E N D I X I I I (continued)

>DUP: Groups of library subroutines often have common instructions which need not be repeated if two similar subroutines are both used simultaneously. DUP allows these common portions to be coded in all symbolic decks, but used only the first time they occur in the Soaping.

Rules: If the D-address of DUP appears in the symbol table, all succeeding cards are completely ignored by SuperSoap except types 1 and 3. The type 3 card stops the effect of DUP (see below). DUP does not enter the symbol into the table. If the D-address is not in the symbol table, nothing happens to assembly.

>PUD: PUD is the same as DUP except it rejects cards when DUP wouldn't, and does not reject them when DUP would.

>OPT: For portions of library routines which are optional and can be omitted, OPT avoids the preparation of several decks for distribution. When OPT appears, the 650 halts (with 1996 in the address lights) and displays the equivalent of the D-address in the lower accumulator and distributor. The operator at the console now chooses whether he wishes to include the next portion of the program or not, by adjusting the sign of the console switches, and then pushes Program Start. If he turns the sign minus, he starts rejecting cards (as in DUP or PUD), and if he turns the sign plus, no effect on assembly occurs.

Type 3 Card: Card type 3 is exactly like card type 1 except it stops the rejection mode which might exist because of DUP, PUD, or OPT.

APPENDIX IV

HAND OPTIMIZING.

SuperSoap has four standard ways of optimizing: normal optimization according to 650 rules, hand optimization specified by the programmer, a mixture of the two which allows forced optimization on selected addresses, and a special optimization for use in debugging (which might be called poor optimization).

When hand optimization is specified, it is given in those portions of the input card that would receive the assembled instruction on output. These fields are known as the LH-field (columns 23-26), the DH-field (columns 33-36), and the IH-field (columns 37-40). When an address is to be defined on the drum, the exact drum level where it is to be put is punched, plus arbitrary multiples of 50, in the proper H-field as a four-digit number. (It must not be punched without leading zeroes, as an absolute address; all four digits must be filled.) Since the assembled instruction will have the same drum level, hand-optimized output may be reloaded as input.

>HMO: (Hand Soap Mode) When in HMO, you must supply a drum level for all undefined drum addresses. Procedure to set up your program is similar to that you would follow when assembling the entire program by hand, but you get all the advantages of symbolic assembly: readability, easy changeability, and automatic bookkeeping. The SuperSoap program itself was hand-optimized after it was put into working order, except for certain of the pseudo operations.

>MMO: (Mixed Mode) When an H-field equivalent is given for an undefined address and you are in MMO, Hand Soap Mode is entered momentarily. If the H-field is blank, however, normal optimization is given momentarily. When reSoaping MMO output it is best to use your old input deck rather than the assembled output.

>NMO: (Normal Mode) NMO puts SuperSoap back into its normal optimizing mode regardless of what mode it was in previously. Initially SuperSoap is in normal mode, so NMO is redundant unless preceded by HMO, MMO, or DMO. When in normal mode, SuperSoap calculates the optimum place to put a location; if there are no locations available, it tries the next level, until it finds an unused

A P P E N D I X I V (continued)

location. Hand Soap Mode looks only for locations on the drum level specified, and stops if none are left.

>NXT: When in normal mode, NXT will change the orginary rule for optimizing the next instruction. Let us call the D-address equivalent of NXT "αaββ" and the I-address equivalent "γγδδ" (If either of these addresses are blank they are given equivalent 0000.) Then the optimizing rule will depend upon the sign of the NXT card:

NXT card positive:	L even, D = L + αa	D even, I = D + γγ
	L odd, D = L + ββ	D odd, I = D + δδ
NXT card negative:	L even, D = L + αa	L even, I = L + γγ
	L odd, D = L + ββ	L odd, I = L + δδ

If the next instruction has tagged addresses, L is <u>not</u> changed to an effective L as it would ordinarily have been.

Example: Suppose you want to include the instruction

BOV

which shuts off the overflow circuit if it was on, and does nothing if it wasn't; both D- and I-addresses are blank. Now ordinarily the D-address would be optimized as L+3, and if the I-address branch is taken, two extra word times are needed, wasting a whole drum revolution. But if you precede the BOV instruction with the card NXT 0505, the D-address will be optimized as L + 5, giving the I-address branch enough time.

>DMO: (Debug Mode) When catching errors in a program you will often come upon an address where trouble is lurking, and you have to locate the spot in your program where this address occurs. At such time it is convenient to know just how Soap goes about picking locations, so that you can immediately pinpoint the place where any location was chosen. If Soap picks locations by some system of simple ordering, so that given any two different locations you can tell which was defined first, you get the desired effect. DMO sets up such a system: addresses are picked in the order 0000,0050,0100, ...,1900,1950,0001,0051, ...,1951,0002, ..., 1952,0003, ..., ...,1998,0049, ...,1949,1999. Get the pattern? Caution: DMO is best used for the entire program. MMO and HMO will <u>not</u> work properly after DMO has been employed, unless BOP intervenes.

APPENDIX IV (continued)

Table of SuperSoap Normal Optimization Rules. These are generally, but not always, as tight as possible, so they are not intended to serve as a hand-optimization guide. A question mark indicates that SuperSoap doesn't really care.

OP-codes	D-address	I-address
AUP,SUP,ALO,SLO,AML, SML,RAU,RSU,RAL,RSL, RAM,RSM	L + 3	D + 5*
LOD,STD	L + 3	D + 3
SIA,SDA	L + 4**	D + 3
STL	L + 5*	D + 3
STU	L + 5**	D + 3
NOP, HLT	?	L + 4
RTC,RTN,RTA,WTN,WTA, BIN,SET,RWT,WTM,BST, SDS,RDS,WDS	?	L + 5
UFA	L + 3	D + 23*
FAD,FSB,FAM,FSM	L + 3	D + 27*
FMP	L + 3	D + 47*
FDV	L + 3	D + 41*
MPY	L + 3	D + 21*
DIV,DVR	L + 3	D + 11*
LIB,SIB	L + 3	D + 12
LDI,STI	L + 3	D + 2
NTS,NEF,BDO,BD9	L + 4	L + 5
BD1-BD8,BOV	L + 3	L + 5
NZA-B-C,BMA-B-C,BMI ...	L + 3	L + 4
NZU	L + 4**	L + 5**
NZE	L + 4*	L + 5*
AXA-B-C,SXA-B-C, RAA-B-C,RSA-B-C	?	D = 0000-7999,8001: L + 6 D = 8000,8004-9059: L + 8 D = 8002: L + 9* D = 8003: L + 9** D = 9060-9999: L + 11* D + 16**
TLE, TLU	L + 3	?
RD1-2-3,WR1-2-3,RPY, RC1-2-3, 12, 13, 62 ..	?	D=xxxn; n=0: L + 23* n=1: L + 7* n=2-9: L +(2n+3)*
SCT,SLT,SRT	?	D=xxxn; n=0: L + 25* n=1-9: L +(2n+5)*
SRD	?	

* One less if base address is odd.
** One less if base address is even.

A P P E N D I X V

SPECIAL OUTPUTS.

For the following special outputs, SuperSoap keeps a "drum image" or "core image" of your assembled program in coded form on disk storage. All three ram arms must be working: arm 0 is used to store instructions for locations 0000-0999, and arm 1 is used for those in the range 1000-1999; arm 2 is used for all other work in the assembly. When instructions are recorded on the drum image, assembly time is increased approximate one-seventh of a second per card -- a 200-card program would take 20 to 30 seconds longer to record on disks.

>>DON: (Drum On) This pseudo-op sets the entire drum image to stop codes (that is, every location L is filled with +01 L 8000) and all succeeding instructions assembled for the drum will be placed onto the drum image as they occur. Important: Once an instruction has been assembled into a location on the drum image, no other instructions will be put into that location until another DON card comes along. You might think of this as loading the instructions onto the drum image backwards, from the last one assembled to the first. Moral -- don't put two different instructions into the same location.

>>CON: (Core On) The core image is set to stop codes by CON, and all instructions assembled for the core will be put onto the core image as they occur. (On the core image, the second of two instructions into the same location will survive, opposite from the drum image.)

>DOF,COF: (Drum Off, Core Off) Stops putting instructions on drum or core imagine, respectively, and forgets everything which was recorded there.

The pseudo-ops which follow, with the exception of MRK and SKP, should follow DON and/or CON, and should precede the use of COF and DOF, or else they will have no information to use. DOF and COF are meant for use after the output pseudo-ops listed here, in case you have more instructions to assemble and you don't want them to go on the drum or core image.

A P P E N D I X V (continued)

>>PDL: (Punch Doubled Drum Locations) All instructions assembled for the drum since the last DON card are checked to see if two went into the same location. If any such locations are found, they are punched onto cards (in columns 1-4 of a special card designed to make an unpleasant noise when punched).

Seven-Per-Card Output.

>>DEK: (Deck) This pseudo-op punches a complete, condensed self-loading and self-checking deck to load your entire assembled program. The DEK card is given after the program has been assembled, and it will load all instructions on the drum and core images.

There must be some band on the drum into which locations 01-24 are not loaded -- this is a prerequisite for the loading routine. You should specify this band in the L-address of the DEK card by giving some location in the band; if the L-address is blank, the 1950 band will be used.

If you have assembled any positive stop codes, they might be forgotten by the DEK routine, so steer clear of positive stop codes when preparing your program.

The card following DEK must be a transfer instruction which will be the first instruction executed when the whole deck has been loaded. It must have blank L-address (just as the FIL constant).

Example: Here is Example Program 1 again, restricted to drum locations 0000-0006, and prepared for DEK. The use of other outputs (tape, disk, etc.) would be similar.

```
          DON
          BLR   7        1999
4         RAA   49       2F
2         WTM   8013
          NZA            2F
          SXA   1        28
2         HLT   8013     4B
  700     DEK
          NOP   0        4B
```

```
0002   +80 0049 0000
0000   +56 8013 0005
0005   +40 0001 0003
0001   +51 0001 0000
0003   +01 8013 0002

8004   +00 0000 0002
```

A P P E N D I X V (continued)

Tape Output.

>>TAP: The L-address 801n specifies the tape unit to be written on and it must be in the range 8010-8015. There are three types of tape output available from the TAP pseudo-op, determined by the D- and I-addresses and the sign.

A. I-address blank, sign positive. If the D-address is blank, it is treated as if it were the absolute address 9000, otherwise it must be a core address. SuperSoap then loads the core with the core image and executes the commands SET (D), WTN 801n. Thus, one- to sixty-word numeric records may be written.

B. I-address blank, sign negative. This is the same as A except an alphanumeric tape record is written.

C. I-address non-blank. In this case both D- and I-address equivalents, which we will call FWA and LWA, must be drum addresses, and FWA must be less than or equal to LWA. All instructions which have been assembled for locations in the range FWA-LWA since the last DON card are put into self-loading form on 60-word numeric tape records; the loading routine does not touch the drum except for the locations FWA to LWA. The number of records is 34 if FWA and LWA are 0 and 1999 respectively; otherwise the number of tape blocks is the number of drum bands touched in the range FWA-LWA.

Examples:			
	8010 TAP 9040		(1 20-word record)
	8010 TAP 100	149	(1 block)
	8010 TAP 99	150	(3 blocks)
	8010 TAP 0	1998	(40 blocks)
	8010 TAP 0	1999	(34 blocks)

To start the loading routine afterwards, SET 9000 and RTN 801n 9001. It must be loaded from the same tape unit on which it was written.

The card following TAP in this case is a transfer instruction which will be the first instruction executed after the drum has been loaded. It must have blank L-address.

The first five columns of the comments field on this transfer card are used as an identifier, and they are placed in the third word of the first block (location 9002), in double-digit representation.

A P P E N D I X V(continued)

>MRK: (Mark) SuperSoap writes a tape mark on the tape unit specified by the L-address.

Example: 8011 MRK.

>>SKP: (Skip) SuperSoap moves tape, on the tape unit specified by the L-address, forward a specified amount. The D-address specifies the number of tape marks to be passed, and the I-address the number of tape records after that point. If either address is blank it is considered zero. Example: 8013 SKP 4 3 spaces tape 8013 forward until 4 tape marks are read, then forward past 3 more tape records.

Disk Storage Output.

>>DSK: The L-address equivalent will be the four-digit location mmnn, where mm is the disk number and nn is the track number of the first track to be written. The number of tracks starting with this location is determined by the D- and I-addresses. There are two types of DSK output available, depending on whether I-address is blank or non-blank.

A. I-address blank. The core image is simply placed onto disk mm track nn. The D-address is ignored.

B. I-address non-blank. In this case both D- and I-address equivalents, which we will call FWA and LWA, must be drum addresses, and FWA must be less than or equal to LWA. All instructions which have been assembled for locations in the range FWA-LWA since the last DON card are put into self-loading form on consecutive disk tracks; the loading routine does not touch the drum except for the locations FWA to LWA. The number of tracks is 34 if FWA and LWA are 0000 and 1999 respectively; otherwise the number of tracks is the number of drum bands touched in the range FWA-LWA, just as in TAP.

To start the loading routine afterwards, get the first portion into core, and exit to 9000; the distributor must contain XXXX0mmnna when you exit. Arm a will be used to load the entire section. For example, the sequence

```
      LOD 1F
      SDS 9000
      RDS 9000  9000
1      00 000m  mnna
```

A P P E N D I X V (continued)

will do nicely, and the instructions fit comfortably on a single card. (The SuperSoap Calling Card is such a card, except it also checks to make sure the Overflow Switch is set to Sense.)

The card following DSK in this case is a transfer instruction which will be the first to be executed after the drum has been loaded. It must have blank L-address.

>>CDD: (Card-to-Disk) This is exactly the same as DSK in every detail, except instead of writing on disks it punches the identical information onto cards (one plus eight times the number of tracks loaded). These cards will later write the information onto disks when they are read. This is so you can easily restore information which has been erased by mistake, or you can easily correct a few lines of coding without reassembling.

APPENDIX VI

FORMATS, LOADING ROUTINES, AND THE SUPERSOAP CALLING CARD.

I. One-per-card output. This is of three forms: Type A, which consists of the pseudo-ops except ALF, types 1 and 3, blank OP-code, PST output, or PAL output; Type B, which consists of types 2 and 4 and instructions with 800X locations; and Type C, which is all others.

8000: RCD 1951 XXXX +

533 panel: Load in column 1.

Each card:
- col. 1 ..... 12 overpunch
- cols. 1-10 ... 00 0000 8000+ on Type A
  69 1954 8000+ on Type B
  69 1954 1953+ on Type C
- cols. 13-16 .. location before DRC or COD (Types B,C)
- cols. 17-20 .. consecutive card count
- cols. 21-30 .. 24 L 8000+ (Types B,C only)
- cols. 31-40 .. instruction, sign over units (Types B,C)
- cols. 41-72 .. input reproduced

II. Five-per-card output.

533 panel: SuperSoap board, or load in column 7.

Each card:
- cols. 1-6 888888
- cols. 7-10 ... consecutive card count
- cols. 11-20 .. instruction 5, sign over units
- cols. 21-30 .. instruction 4, sign over units
- cols. 31-40 .. instruction 3, sign over units
- cols. 41-50 .. instruction 2, sign over units
- cols. 51-60 .. instruction 1, sign over units
- cols. 61-64 .. location 5
- cols. 65-68 .. location 4
- cols. 69-72 .. location 3
- cols. 73-76 .. location 2
- cols. 77-80 .. location 1

PLR loading routine. Index register C keeps count of how many cards were loaded. The "*" tag indicates the proper multiple of 50 specifying the band is added to the address.

8000: RCD 1951 XXXX +

First Card	Second Card
RD1 9042 9898	RAC 0000 9049
RAA 0004 0023*	STI 0012*0020*
SXA 0001 0024*	STD 4000 0016*
RD1 0017*9898	BMI 0002*0018*
RAU 0007*0014*	RAB 8002 0022*
LOD 2002*0013*	NZA 0019*0020*
ALO 0008*0015*	AXC 0001 0021*
SRT 0004 0015*	LDI 1952 9043

A P P E N D I X V I (Continued)

III. FIL output. "*" indicates the proper multiple of 50 specifying the drum band is added.

One-per-card form			Five-per-card form
LOD	1954	1953	88 8888 (COUNT)
(CARD COUNT)			LOD 0005*0004*
RAA(LWA-FWA)		1957	SXA 0001 0008*
(FIL CONSTANT)			RAA(LWA-FWA)0008*
LOD	1955	8001	(FIL CONSTANT)
SXA	0001	1957	NZA 0003*0020*
STD	(FWA)A	1958	- 88 8888 8888
NZA	1956	8000	STD (FWA)A 0006*

IV. PAT output. "λ" is the drum level 00-49 in each band. "8" on the card means the location is available, "0" means it is unavailable.

Each card: cols. 1-10 ... 00 (λ)(λ+450)
cols. 11-20 .. availability for λ,λ+50,λ+100,...,λ+450
cols. 21-30 .. 00 (λ+500)(λ+950)
cols. 31-40 .. availability for λ+500,λ+550,...,λ+950
cols. 41-50 .. 00 (λ+1000)(λ+1450)
cols. 51-60 .. availability for λ+1000,...,λ+1450
cols. 61-70 .. 00 (λ+1500)(λ+1950)
cols. 71-80 .. availability for λ+1500,...,λ+1950
cols. 10,20,30,40,41,50,60,70,80 ... 12 overpunches

V. Seven-per-card (DEK) output.

8000: RCD 1951 XXXX +

Control Panel: SuperSoap, or load in column 7 or 41

First Card	Second Card	Last Card
RD1 9042 9898	RAC 0000 9049	00 0017*0001
SXA 0001 0023*	STI 0012*0017*	SXC (m-2)0003*
SLT 0002 0014*	RAA 8002 0020*	NZC 0005*0004*
ALO 0016*0022*	SXC 8003 0021*	Transfer Instruction
AXC 0025 0017*	NZA 0018*0019*	HLT 9898 9899
SDA 0012*0015*	STD 2000 0015*	-- ---- ----
LOD 2002*0012*	RD1 0024*9898	-- ---- ----
RAM 0001*0013*	LDI 1952 9043	-- ---- ----

Middle Cards: cols. 1-2 00
cols. 3-6 .... location 1 (see below)
cols. 7-10 ... number of instructions on card
cols. 11-20 .. instruction 1, sign over units
cols. 21-30 .. instruction 2, sign over units
cols. 31-40 .. instruction 3, sign over units
cols. 41-50 .. instruction 4, sign over units
cols. 51-60 .. instruction 5, sign over units
cols. 61-70 .. instruction 6, sign over units
cols. 71-80 .. instruction 7, sign over units
cols. 7,10,41 .. 12 overpunches

A P P E N D I X V I (Continued)

Notes: The location specified in columns 3-6 is changed to 72XX if it is actually core location 90XX. "m" is the total number of cards punched. "*" indicates the proper multiple of 50 specifying the band is added to this address. The loading routine checks to see that the exact number of cards is loaded, but a provision has been made for adding extra cards. The loading routine checks to see that the exact number of cards is loaded, but a provision has been made for adding extra cards. The numbers in columns 1 and 2 of the middle cards are subtracted from the cumulative count, so if you are adding eleven cards, put 11 instead of 00 in columns 1 and 2 of one of the added cards. In that way, if any of the cards is missing, the checking feature will detect an error. The stop 9898 occurs when an improper number of cards is loaded. You may also adjust the last card if you wish to alter the finished deck.

VI. TAP output from TAP 0000 1999.

The loading routine is located in the first forty locations of the first block, and the remaining blocks contain sixty words to be loaded onto the drum. The loading routine in Soap language is as follows:

```
 9001  NTS 1F
       RAL 2F
       RAC  9     3F
3      BST  801n
       SET  9000  8002
2      RTN  801n
       NTS 1F
       SXC  1
       NZC 3B      9900
1      RAC  10     9009
 9009  STI  1969
       RSA  1700
       RAL 1F    2F
2      STI  1700A
       RAB  0      8002
1      RTN  801n
       AXB  1
       LOD J0001B
       NTS         1F
       RAC  10     8001
1      BST  801n
       SXB  1
       SXC  1
       NZC         9901
       SET  9000  8002
J0002  STI  1720A
       STI  1750A 8002
```

APPENDIX VI (Continued)

```
J0003 STI  1780A
      STI  1800A 8002
J0004 STI  1840A
      BMA 8F    9F
8     STI  1850A 8002
J0005 STI  1900A
      STI  1950A 8002
J0006 STI  1960A
      AXA  300  2B
9     RAL       8B
      (transfer instruction)
```

Programmed stops: 9900; 10 unsuccessful attempts to read first block
9901; 10 unsuccessful attempts to read tape block

VII. TAP output from TAP FWA LWA. The loading routine is placed in the first ten locations of each block, and the remaining locations are filled with one to fifty words of information to be loaded into a drum band, ending with location 9059. The loading routine is

```
9000:   RTN  80ln  9001
9001:   NTS  9006  9003
9002:     Identifier
9003:   HLT    d   9004
9004:   BST  80ln  9000
9005:
9006:   LOD  9009  9008
9007:   Transfer Instruction
9008:   SET  90XX  8001
9009:   STI    d   9000 , on last block this becomes STI  d  9007
```

VIII. DSK output from DSK 0000 1999. The loading routine is located in the first forty locations of the first track, and the remaining tracks contain sixty words to be loaded onto the drum. The loading routine in Soap language is as follows:

```
 9000 RAU  8001
      RSA  1700
      ALO 1F
      RAB  0      9005
 9005 STI  1965   2F
2     AUP 10I
      LOD  8003
      SDS  9000   J0001B
J0001 STI  1700A  8002
1     RDS  9000
      AXB  1      2B
J0002 STI  1720A
      STI  1750A  8002
J0003 STI  1780A
      STI  1800A  8002
```

A P P E N D I X V I (Continued)

```
J0004 STI 1840A
      BMA       9F
      STI 1850A  8002
J0005 STI 1900A
      STI 1950A  8002
J0006 STI 1960A
      AXA 300
      RAB 0      J0001
9     LOD1F
      STI 1950   8001
1     Transfer Instruction
10I    00 0      10
```

IX. DSK output from DSK FWA LWA. The loading routine is placed in the first ten locations of each track, and the remaining locations are filled with one to fifty words of information to be loaded into a drum band, ending with location 9059. The loading routine is

```
9000:    RAU  8001  9001, on last track, LOD 9009 9008
9001:    AUP  9002  9003
9002:     00  0000  0010
9003:    LOD  8003  9004
9004:    SDS  9000  9005
9005:    LOD  9009  9008
9006:    RDS  9000  9000
9007:    LOD  8003  9006, on last track, transfer instruction
9008:    SET  90XX  8001
9009:    STI   d    9007
```

X. CDD output. A single card begins the loading routine, then there are eight more cards for each track to be loaded.

8000: RCD 1951 XXXX +

Control Panel: SuperSoap or load in column 1 or 41

Initial Card	1st Card of Track	. . . 7th Card of Track	Last Card of Track
LOD 1952 1953	loc. 9000	loc. 9048	loc. 9056
SET 9056A1956	loc. 9001	loc. 9049	loc. 9057
STD 1900 1954	loc. 9002	loc. 9050	loc. 9058
RSA 0056 1955	loc. 9003	loc. 9051	loc. 9059
RD1 1900 9898	loc. 9004	loc. 9052	LOD 1908 1906
LDI 1901 1957	loc. 9005	loc. 9053	SDS 9000 1907
NZA 1958 1905	loc. 9006	loc. 9054	WDS 9000 1954
AXA 0008 1955	loc. 9007	loc. 9055	00 000m mnn2

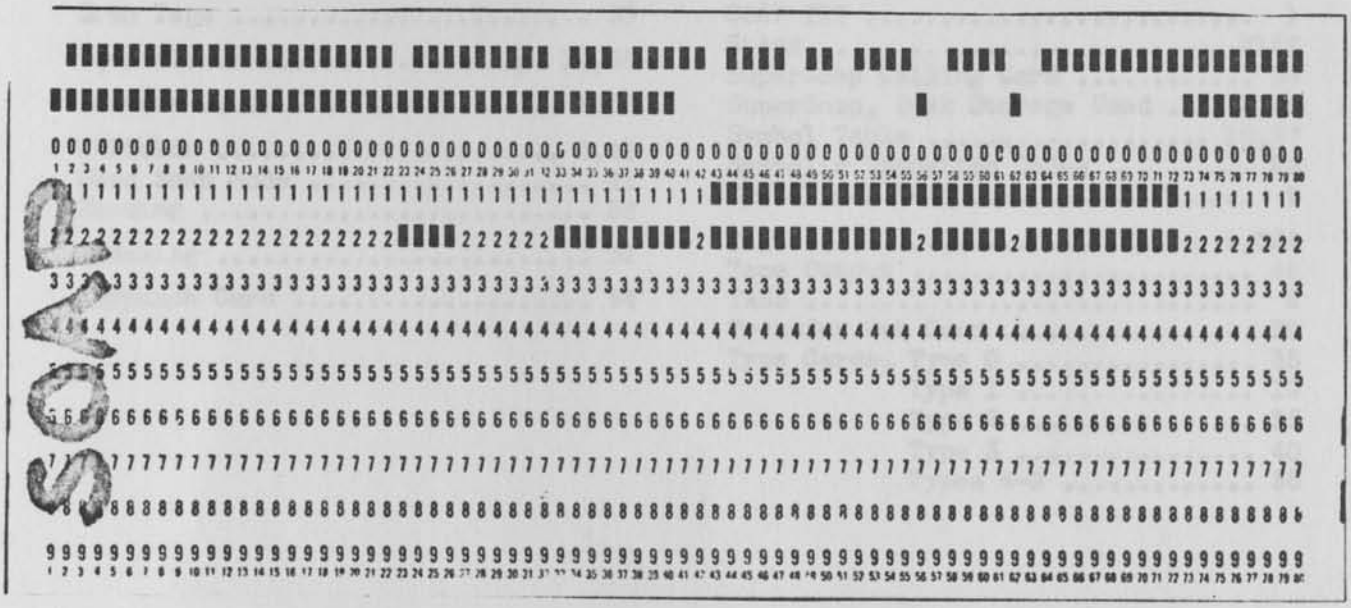
Word 7 of the very last card out is changed to WDS 9000 8000.

A P P E N D I X V I (Continued)

XI. SUPERSOAP CALLING CARD.

```
8000: RCD 1951 8282 +
Control Panel: SuperSoap
Card:  12 overpunch in column 2
                65 1954 1957+
                 S UPER SOAP
                36 0001 1955+
                00 0003 8002+
                47 1958 9876+
                  (date)
                85 9000 1953+
                86 9000 9000+
```

XII. Keypunch Card. This card does not have anything built in for Alternate Program, but variations are easily obtainable with it by turning off the Auto Skip control to punch tags or to punch hand-optimization factors. Backspace to punch type numbers or signs.



INDEX