PROGRAMMING DEPARTMENT MEMO

TO: Ed Bebb

December 1, 1967
PR-67-4239

FROM: Jim MacIntyre

SUBJECT: HOW TO MAKE A 900 META-SYMBOL SYSTEM

COPY TO: D. Corkum, J. DeLuca, S. Klee, D. Perkins, B. Reid, K. Rector

- -

Introduction

Once upon a time, Programmatics wrote a three-pass Meta-Symbol Assembler to run in 12K under Monarch. It was a relatively clean system with a simple system-make procedure. Somebody then made the decision that since Monarch could operate in 8K, all processors under Monarch should operate, and be maintainable in 8K. Also, a Concordance was to be added. The cutting and squeezing began. The result is the present 8-overlay Meta-Symbol Assembler. Because of space considerations, the overhead of I/O handlers was a luxury that could not be afforded. Consequently, only "common" I/O is resident (MSCONTROL); the I/O for LO to the printer, being used only in PAS2, FINISH, and CON2 is written in-line in these passes. (The ramifications of this may be seen in the present Unbuffered Printer Update Packages). Of the 8 overlays, the first is loaded by the Monarch Loader, and intercommunication between the programs which make up this overlay is by external references and definitions. The last 7 overlays, on the other hand, are absolute decks with no external references or definitions, since the small resident system overlay loader (TAPE LOADER) can load only the restricted absolute, unblocked format. All inter-communication between overlays is through absolute locations which are assembled into the routines of each overlay as absolute EQU's. Essentially the system is set in concrete; even though "relocatable" decks are used in constructing absolute overlays, the whole system is extremely sensitive to relocation of any segment or change in size and arrangement of tables.

This memo will endeavor to describe the things about the system which the user will need to know to generate a working Meta-Symbol system, and in particular, will emphasize the pitfalls the user must avoid if he wishes to modify Meta-Symbol successfully. The reader with more than a curiosity about the system should be equipped with a set of Meta-Symbol listings and a system map of the Monarch tape. Although the 910 and 920 systems do not operate interchangeably, the listings are identical; the difference lies in the use of POPS and method of creating system overlap. (910 = 910/925; 920 = 920/930 throughout the discussion.) The discussion which follows describes the generation of both 910 and 920 systems. (Note: Although the 9300 Meta-Symbol operates like 900 Meta-Symbol, its method of generation is so radically different as to merit only this cursory note.)

The Routines of Meta-Symbol

The Routines of Meta-Symbol are listed below, numbered as individual assemblies and identified by the overlay they are used in (the POPS are indicated only as separate assemblies, although they in essence are included in each overlay. The procedure will be explained later.)

1.	920 POPS	
2.	910 POPS	
3.	ENCODER	
4.	S4B	
5.	MON1	OVERLAY 1
6.	TAPELOADER	
7.	MSCONTROL	
8.	PREASSEMBLER PART1 (P1)	OVERLAY 2
9.	PREASSEMBLER PART2 (P2)	
10.	SHRINK	OVERLAY 3
11.	ASSEMBLER PART1 (M1)	
12.	ASSEMBLER PART2 (M2)	
13.	ASSEMBLER PART3 (M3)	OVERLAY 4
14.	ASSEMBLER PART4 (M4)	
15.	ASSEMBLER PART5 (M5)	
16.	PAS2	OVERLAY 5
17.	FINISH	OVERLAY 6
18.	CONCORDANCE PART1 (CONCRD)	OVERLAY 7
19.	CONCORDANCE PART2 (CON2)	OVERLAY 8

Assembling the Routines of Meta-Symbol

Each routine of M-S may be assembled with META910 or META920, with the exception of the 910 POPS, which must be assembled with META 910 and the 920 PSEUDO-POPS, which must be assembled with META920. Each routine is preceded by PROCedures which define 920 instructions with operation codes between octal 100 - 117. This causes any 920 instruction to POP on either 910 or 920. For example, the OP code for CAB is 100, for SKR, 107. This is true for each routine. These arbitrary POP codes will be generated, no matter whether the routine is assembled with META910, or META920. (Of course, the 910 POPS and 920 POPS should contain no POPS themselves - if you find I flags on any instructions in these routines, they have been incorrectly assembled. This would lead to a most peculiar form of recursion!)

Note that although POP codes are generated for 920 instructions and I flags occur on these instructions, these codes are absolutely unique; nowhere is a POP ref/def item generated or used. For example, for SKR exp it is as though the op code 0107 were merged with the value of exp. The machinery in the PROCS preceding each routine which generates the I flag without producing a POP reference item is clever and is worthy of the reader's perusal. Although this point may seem tedious, it is important to note that POPS for 920 instructions are unique, forced and exist on both systems.

How POPS are used in the M-S Assembler

As we have seen, a 920 instruction not in the 910 subset will POP on both 910 and 920 systems through a unique POP transfer location in 100 - 117 which is identical for each routine and for both 910 and 920 systems. Let us trace the execution of an ADM instruction first in the 920 system and then in the 910 system. If we looked at the ADM instruction in memory at location L, it would be 0112 in both 910 and 920 systems. On the 920, POP code 112 causes a transfer to location 0112, which contains a BRM CHANGE, where CHANGE is located in the relocatable section of the 920 PSEUDO POPS. The PSEUDO POPS then replaces the POP instruction at location L with the actual 920 instruction for ADM, retaining the index, indirect and address characteristics, and executes the instruction. Thus, when a POP instruction is encountered on the 920, it is replaced by the actual instruction. In loops containing a POP instruction, the POP occurs only the first time and the instruction itself is executed all other times in location L of that overlay. On a 910 system, the ADM instruction at location L is a 0112. When the POP occurs, the instruction is simulated by the 910 POPS, and no modification takes place. Now trace some other instructions through the 910 and 920 POPS to ensure that you really understand how this works.

Note that both the 910 POPS and 920 PSEUDO POPS contain both AORGS and RORGS. The AORGS define the absolute section 100 - 117 where the POP transfers are located. The RORGS define the relocatable section of both packages which will be located at different points in memory for different overlays.

DTAB

In ENCODER and 910 POPS there is cell labelled DTAB DATA N. It is AORGed at 01372. It is an extremely important cell, since it contains the address of the top of the longest overlay in the M-S system. It is used for the beginning of certain tables. At present, since PAS2 is the longest, the value in DTAB would be calculated as the last location in PAS2 plus the length of the relocatable section of the POPS being used in that system. For example, if PAS2 ended at 013500, DTAB for 920 would contain 013500 + 048 (length of relocatable section of 920 PSEUDO POPS) = 013548. We would probably set DTAB to 013600 for a bit of breathing space, depending on the tightness of the system. On 910, DTAB = 013500 + 0260 = 013760 or 014000 for safety. DTAB may be set too high, just don't set it too low! It must clear the top of PAS2 + POPS. The DTAB value for 920 is assembled into the DTAB cell in ENCODER, the 910 value is assembled in the DTAB cell in the 910 POPS. (Note: The 920 POPS contains no DTAB). As description of the system continues, the determination of DTAB value will also be more clearly seen. (See also APPENDIX B).

OVERLAY 1

The routines in OVERLAY 1 in the order of loading by the Monarch loader are (Δ1 and Δ2 records are indicated also):

Δ1 METASYM

Δ2 ENCODER

ENCODER (BIN)

910 POPS OR 920 PSEUDO POPS (BIN)

Δ2 MON1

S4B (BIN)

MON1 (BIN)

Δ2 MSCONTRL

TAPELOADER (BIN)

MSCONTRL (BIN)

ENCODER is ORGed at 01372; although it is a relocatable program, it is loaded at 0 and its ORG effectively absolutely positions it at 01372. Notice that its references to MSCONTRL and TAPELOADER are absolute through EQU's. These must be changed in all overlays if change is necessary. The last definition in ENCODER is ZTABLES EQU $+01640. This value of ZTABLE can be changed only with discretion. ENCODER is the routine which reads in Symbolic/Encoded cards, builds a dictionary in core, merges corrections where necessary, and outputs an encoded bit string to tape X1. ENCODER contains the 920 value for DTAB.

910/920 POPS

The 910 POPS or 920 PSEUDO POPS are loaded so that the transfer vector has an AORG 0100 and the relocatable section is located above ENCODER. These will function thus for the first overlay only. They will be repositioned for succeeding overlays. If the 910 POPS are loaded, a new 910 value for DTAB (AORG 1372) overlays the 920 value loaded in ENCODER. If 920 POPS are loaded, the initial 920 value in DTAB is unchanged.

S4B

S4B (RORG 0) is relocated above the POPS. If the C option is called, it will be utilized to translate from old Symbol 4 code to Modern Meta-Symbol code; it translates such items as VFD to FORM, etc. The actual translation is done during encoding and the ENCODED or Source Output (including LO) will contain the translation into Meta-Symbol language.

MON1

MON1 is a relocatable routine with RORG0, loaded just above S4B. It is the I/O initialization section of Meta-Symbol. By querying the Monarch Unit Assignment Table and MSFNC (0273 in MSCONTRL, the cell which Monarch Action routine initialized with parameters on the META control card), it initializes the unit and channel numbers in all resident I/O in MSCONTRL. After initialization has taken place, MON1 will be overlaid by Encoder tables.

TAPELOADER

The TAPE LOADER (AORG 2) is a short loader used to load overlays from the systems tape. It reads only absolute subset of the 900 Standard Binary Format, unblocked records only; it can search the system tape for Δ2 labels.

MSCONTROL

MSCONTROL (AORG 0200) contains the resident I/O information. It is responsible for all input/output except the printing to the Line Printer or Typewriter done by PAS2 or Concordance (CON2) when listing. If PAS2 puts listing out to Mag Tape for instance, the Mag Tape routine in MSCONTROL will be used. MSCONTROL also contains the ABORT logic for typing out the Meta-Symbol ABORT message and returning to Monarch. MS CONTROL, which is the last program of OVERLAY 1 to be loaded, contains an end transfer to ENCODE, a cell containing a BRU to TRACOR, the entry point of ENCODER. Thus ENCODER is the first program to be executed after the loading of the first overlay.

OVERLAY 2

PREASSEMBLER PART 1 is a relocatable program with an origin of Octal 1403. Loading this at 0 effectively positions this overlay absolutely in the correct place. Looking down to approximately line 167 of the listing of PREASSEMBLER PART 1, we find an ORG 01540 followed by some EOM's and SKS's. If you follow the Octal addressing, you will notice that this section effectively overlays the preceding reserve area. In addition, around line 348, just preceding the label PREASSEM, there is another ORG at PIERT + 2. This is the initialization section of PREASSEMBLER, and will be overlaid later by quantities placed into the reserve section defined at the beginning of the program. Further on down about line 416, there is another ORG at CHNG1+2 following the comment "END OF INITIALIZATION CODE." This is the actual operating portion of the PREASSEMBLER. It is worth taking the time to actually map out the sequence of code and the overlays into memory of this particular program. Notice that the lowest portion in memory where meaningful coding exists is Octal 1540 where those EOM's and SKS's are established. The drawing of the map is left an exercise to the user. The relocatable section of the POPS will be loaded between P1 & P2. The second portion of the PREASSEMBLER is RORGed at 0. It is also relocatable and will be loaded after PREASSEMBLER PART 1 and the POPS. Notice that PREASSEMBLER PART 2 has as its last cell the label, $LLITX and the unique literal 01234567. It uses this to find the end of its own string of literals and thus begin its tables.

PROCEDURES

Since the PROCS go on the tape just as they come in ENCODED form, it is not necessary to alter these. However, the reader is warned that there is a machine definition card which must precede every PROC Deck on the System tape. The description of this card is contained on Page 42 of Symbol - Meta-Symbol Manual, under the heading System Procedures. The PRE-ASSEMBLER will search the tape for the Δ2 label of the proper set of PROCS, load it into memory and build all the Symbol Tables accordingly as it makes its first pass through the bit string on X1.

OVERLAY 3

The next program is SHRINK, AORG at Octal 4000. It has external references to many labels in PREASSEMBLER PART 1 and 2, and overlays only a portion of Part 2 (i.e., the portion from Octal 4000 to the end of SHRINK). Notice that the second to the last label in SHRINK is called PSMPLC EQU $+0100. This effectively allows room for the literals and gives SHRINK some working storage. The purpose of SHRINK is to purge unwanted procedures from the procedure sample table so that more table space will be allowed for the rest of the Assembly.

OVERLAY 4

M1 through M5 are the portions of the first pass of the Assembler. It was split into portions only because it could not be assembled in 8K as a single overlay. Notice that M1 is RORGed at Octal 1407; although it is a relocatable program, loading it at 0 effectively places it correctly in memory. M2 through M5 have RORGS of 0 and are located consecutively following M1. The loading of these 5 programs plus the POPS constitutes the whole of ASSEMBLER PART 1.

OVERLAY 5

PAS2 is an absolutely origined program (AORG 01407). It is put on the tape just as it comes from the Assembly with definition cards removed. The definition cards from PAS2 will be used to satisfy the external references in FINISH, the next overlay.

OVERLAY 6

FINISH is an absolutely origined overlay with an AORG of Octal 4700, It overlays a portion of PAS2 and makes references to routines in PAS2.

OVERLAY 7 & 8

CONCORDANCE PART 1 and CONCORDANCE PART 2 are both assembled absolute and they go on the system tape exactly as they come from the Assembly.

At this point, we have made one rough run over the Meta-Symbol Decks. The user can familiarize himself with the system, first of all by going through and marking all cells which are absolute, by noting all the inter-communication which is done by absolute cells and by mapping the origins of each overlay. While this is rather tedious, it is somewhat necessary to an understanding of how the system works.

Construction of the Overlays

This section describes how the Overlays are to be formed in memory and the absolute overlay created for the M-S tape. A following section will describe the actual System Make procedure in more detail.

Overlay 1

Overlay 1 consists of the Binary Decks as they come from Assembly in the following order:

Δ1 META SYM

Δ2 ENCODER

(Binary deck of ENCODER)

(Binary deck of 910 or 920 POPS)

Δ2 MON1

(Binary deck of S4B)

(Binary Deck of MON1)

Δ2 MSCONTROL

(Binary Deck of TAPE LOADER)

(Binary Deck of MSCONTROL)

These routines make up the first overlay. When a Meta-Symbol Card is encountered by the Monarch System, it will go to the Meta-Symbol action routine which searches the system tape for the Δ1 METASYM Label. Ignoring Δ2's, it will load decks up to the end transfer which is on MSCONTROL. Prior to this the cell MSFNC has been initialized by the action routine according to the parameters on the Meta-Symbol Card. ENCODER is loaded by the Monarch Loader at 0 and its ORG of Octal 1372 positions it in memory. The POPS, which have a relocatabl origin of 0, and an absolute origin of 0100, S4B which has a relocatable origin of 0, followed by MON1 which has a relocatable origin of 0 are then loaded following ENCODER. This completes the relocatable section of the first overlay. TAPE LOADER is then loaded starting at absolute origin of Octal 2. Finally MSCONTROL is loaded with an absolute origin of Octal 200. All refs and defs are satisfied by Monarch loader and control is transferred to the end transfer location of MSCONTROL, initiating the Meta-Symbol System. From here on, the Monarch System is not used; the Meta Symbol TAPE LOADER takes care of loading all the overlays necessary for the execution of the Meta-Symbol Assembly. Control is only returned to Monarch in case of completion of the Assembly and/or CONCORDANCE or an ABORT situation. The first Overlay is the only Overlay on the system tape which contains external references and definitions. It is the only Overlay which is loaded by the Monarch Loader, which can satisfy all these refs and defs.

Overlay 2

Overlay 2 consists of PREASSEMBLER PART 1 (P1), the POPS (910 or 920) and P2. It will be formed by loading PREASSEMBLER PART 1, the suitable POPS, and PREASSEMBLER PART 2 into memory with the Monarch loader and dumping out in absolute version 100 to 117, which contain the POP transfer locations, and Octal 1540 through the top of PART 2 of PREASSEMBLER. In general, in making the absolute decks reserve locations are not to be dumped. Output only meaningful data. The reserves are often used as inter-communication between two different Overlays; by dumping them in making the absolute decks we may overlay some meaningful data which we meant to leave in memory between overlays. So although PREASSEMBLER PART 1 is ORGed at 01403, we dump only from 01540, the first meaningful data.

Overlay 3

The third Overlay, SHRINK, will be formed by loading P1, POPS, and P2 along with the SHRINK deck to satisfy all references and definitions and then dumping from the beginning of SHRINK (Octal 4015) to the end of SHRINK.

Overlay 4

The fourth OVERLAY, ASSEMBLER, will be formed by loading the POPS into memory at a position where they will not be overlaid by PAS2 and yet will lie under the value of DTAB. After the Monarch loader has been used to load the POPS, and M1 through M5, then the portion 0100 to 0117 will be dumped absolutely and the portion from 01705, which is the first meaningful data cell of ASSEMBLER PART 1, through the top of POPS will be dumped. This will form the ASSEMBLER Overlay.

Overlay 5

PAS2 will be formed by stripping the definition cards from the front of the Binary Deck as it came from the Assembly and using this absolute deck as the Overlay. Notice that when it is read into CORE, it uses the POPS left there by PAS1. Remember that DTAB was calculated such that if the POPS were loaded directly beneath DTAB, PAS2 could load in without overlaying the POPS. Therefore, we just use the Binary Deck for PAS2 as it comes from the Assembly sans def cards.

Overlay 6

FINISH will overlay a portion of PAS2; it makes references to labels and subroutines in PAS2. Notice that it is an absolutely origined deck. To form the FINISH Overlay, we attach the defs from PAS2 to the FINISH Binary Deck, load it into memory, and punch out the portion from the beginning of FINISH (04700) to the end of FINISH.

Overlay 7 & 8

The Overlays for CONCORDANCE PART 1 and PART 2 are put on the system tape exactly as they come from the Assembly.

Actually Making the SYSTEM

We will assume that the user is now sitting at a machine with a Card Punch, Set of Binary Decks and a Monarch System. The user will also need a copy of Program Catalog Number 000018C, Binary Dump to Paper Tape or Cards. Load 18C into Memory with the Monarch Loader at DTAB or above, but where it will not conflict with the Monarch Loader Tables. It will remain resident in memory during the making of all the Overlays. Take note of the entry location for 18C. Also notice that for the punching of cards, Break Points 3 and 4 must be set; otherwise a tape with bootstrap will be punched.

Overlay 2 PREASSEMBLER

Boot the Monarch System. Using the ΔLOAD, STOP, Commands, load PREASSEMBLER PAS1 at 0, load the 910 or 920 POPS, and load PREASSEMBLER PART 2. Note that the loader will stop after the loading of each of these decks. After PART 2 has been loaded, the C register will contain the transfer address and the B register will contain the last location plus 1. Now transfer to the Dump program. Dump location 100 through 117 with no transfer address (i.e., set X = 0). Now dump location 1540 through the top of PART 2 with the transfer address in the X register. The Deck punched out is now the absolute deck for PREASSEMBLER. This is preceded by a Δ2 PREASSEM Card in the System Deck. Now the PROC Decks follow with their Δ2 Cards and the machine identification card discussed earlier.

Overlay 3 SHRINK

To form the SHRINK Overlay, set up a Binary Deck as follows: P1, POPS, P2 with its end card removed and the SHRINK Deck. This effectively loads SHRINK as though it were part of P2. If we left the end transfer on PART 2, we naturally could not load the SHRINK deck. Boot Monarch in. Using the ΔLOAD, STOP function and a bias of 0, load P1, POPS, and then the third deck consisting of P2 plus SHRINK. Note the ending location and transfer address of SHRINK. Now using the punch program, punch from the beginning of meaningful data in SHRINK, 04015, through the end of SHRINK with transfer address in the X register. Note that it is not necessary to punch out the POPS at this time as they will be left there from the PREASSEMBLER Overlay. SHRINK does not overlay the POPS. This constitutes the SHRINK overlay Deck for the System and is now put in the System Deck with a Δ2 SHRINK Card preceding it.

Overlay 4 ASSEMBLER

Assuming that the calculation for DTAB has been done, we now have to calculate a bias for the POPS approximately 42 or 260 Octal locations below DTAB, depending on which POP system we are using. Using the Monarch Loader with the stop function, we load the POPS at this bias. When the Loader stops, we reset the bias in the B register to 0 and load Overlays M1 through M5. Because of the ORG on M1 they will be located correctly in memory. Having loaded the POPS in at its bias below DTAB and loading M1 through M5, we now punch out locations 100 through 117 for the POPS transfer locations, and 1705 through the top of the POPS with an end transfer as determined from the values in the C and B registers. We now have an absolute deck consisting of 100 to 117 and 1705 through the top of the POPS with an end transfer. This constitutes the absolute Overlay of the ASSEMBLER.

Overlay 5 PAS2

To form Overlay 5, PAS2, we strip the DEF Cards (type 1) from the front of the Binary Deck, we got from the Assembly, and use the remaining deck as the absolute overlay. Looking at the listings, we must be careful to determine that the last location on PAS2 lies below the current bias for the POPS. When that check is made the Deck is ready to go on the system.

Overlay 6 FINISH

To make the FINISH Overlay put the DEF Cards from PAS2 onto the front of the FINISH Binary Deck. Load it with the Monarch Loader into 0, STOP. Its absolute origin biases it correctly. After loading we determine the final location in B and the transfer location from C. Punch from the beginning of FINISH, (04705) through the end of FINISH with the transfer address. Note we do not punch the POP locations or any of PAS2, since these will still be left in CORE from the previous Overlay at execution time. This absolute deck is now the Overlay for FINISH.

Overlay 7 & 8 CONCORD and CON2

To make the CONCORDANCE PART 1, CONCORDANCE PART 2 Overlays we merely use the Binary Decks from the Assembly.

These constitute the Overlays for Meta Symbol for Monarch Tape. Make sure that every Overlay is preceded with the proper Δ2 Label Card. Notice that it is possible to remake a single Overlay and to replace this using the UPDATE procedure on the Monarch Tape. However, care must be taken that the values of DTAB and linkages with the POPS, etc., for that Overlay are properly taken care of. Appendix A describes the final overlay deck structure for system update.

Review

Let us do a final review of the making of overlays so that we know the exact nature of the decks we are using to update the Meta-Symbol processor on the Monarch tape.

Following the Δ1 METASYM I.D. Card, we have Δ2 ENCODER I.D., the binary decks for Encoder and 910 or 920 POPS, a Δ2 MON1 I.D., the S4B and MON1 binary decks, a Δ2 MSCONTROL I.D. and binary decks of TAPELOADER and MSCONTROL. This constitutes OVERLAY 1 and will be loaded by the Monarch Loader upon encountering a ΔMETA Control Card. Overlay 2 is preceded by a Δ2 PREASSEM I.D. The absolute overlay was formed by loading P1, the POPS and P2 and dumping 0100 - 117, and 01540 to top of P2 with an end transfer into P2. It is a single absolute deck.

The six procedure decks follow, each one an encoded deck preceded by a Δ2 PROCXXXX I. D. and machine identification card. Overlay 3 is preceded by a Δ2 SHRINK I.D. The absolute deck was formed by loading P1, POPS, P2 (without end card) and SHRINK binary deck, and dumping 04015 (beginning of SHRINK) to the end of SHRINK with transfer address. It is a single absolute deck. Overlay 4 is preceded by an Δ2 ASSEMBLER I.D. The absolute deck was formed by calculating DTAB, loading the POPS at a bias below DTAB, resetting the bias to zero, loading M1 - M5, and dumping 0100 - 117 and 01705 to the top of the POPS, with end transfer into M5. It is a single absolute deck. Overlay 5 is preceded by a Δ2 PAS2 I.D. It is a single absolute deck from the assembly with the def cards removed (type 1). Overlay 6 is preceded by a Δ2 FINISH I.D. The absolute overlay was formed by putting the def cards from PAS2 on the front of the FINISH deck, loading it at 0 and dumping from the beginning of FINISH (04700) to the top of FINISH with end transfer. It is a single absolute deck. Overlay 7 is preceded by a Δ2 CONCRD I.D. It was formed using the binary deck direct from assembly. Overlay 8 is preceded by a Δ2 CON2 I.D. It was formed using the binary deck direct from assembly.

This completes the description of ABS overlays for the creation of the Meta Symbol Tape. The only difference then between the 910 and the 920 Tape is the POPS which are used. You should see now, however, that the size of the POPS makes the size of the Overlays differ and makes the value for DTAB differ for the two systems. Only a 910 system with a 910 POPS will run on a 910/925. Only a system containing 920 PSEUDO POPS will run on the 920/930. Although the Monarch System Tapes run interchangeably on both systems, processors do not.

It seems advisable that during the creation of the system a careful map of loading and dumping should be kept in case the system does not function correctly. Once again, remember that reserve locations at the beginning of the Overlay are not punched out. However, the listings must be studied carefully to determine that useful information is not neglected. For instance, remember the situation of the PREASSEMBLER where the Origins are reset in the body of PART 1 of the PREASSEMBLER. Assemblies to create the binary decks may be done with either Meta 910 or 920, since any instructions which are not in the 910 Subset are automatically forced to POP by the procedure definition at the beginning of the Deck (except 910/920 POPS). If a reassembly is done, check that the POP operation codes on the listing correspond with the actual transfers in the POPS. None of the Overlays use the POP machinery of the Monarch loader. All POP operation codes must be generated absolutely at assembly time, or the system will not function.

The complete discussion here has been oriented to creating a system using Card Input and Card Output. It seems that this could be done equivalently on Paper Tape, with two exceptions. In the making of the SHRINK Overlay, we removed an end card from P2 to properly orient the loading of the SHRINK Overlay with P1, POPS, and P2, so that defs and refs could be satisfied. Also, we used PAS2 defs on the FINISH deck. On Paper Tape this may be rather difficult. A little bit of ingenuity on your part may overcome this. Good Luck.

CONCLUSION

It is hoped that utilizing this discussion, the information in the SYMBOL - META-SYMBOL Manual and the information in the Meta Symbol Tech Manual (Section 5, the Operational Information, in the back of the Manual), the user may successfully create his own Meta-Symbol System. It is this writer's suggestion that the user try to recreate an existing system before trying any modifications.

APOLOGY

The writer apologizes for the length, complexity and redudancy of this memo. However, it is somewhat like trying to explain the nature of an elephant to the three blind Indian philosophers, who held the tail, trunk and leg of the animal respectively. I hope the memo has explained the task from your point of view.

[illegible]

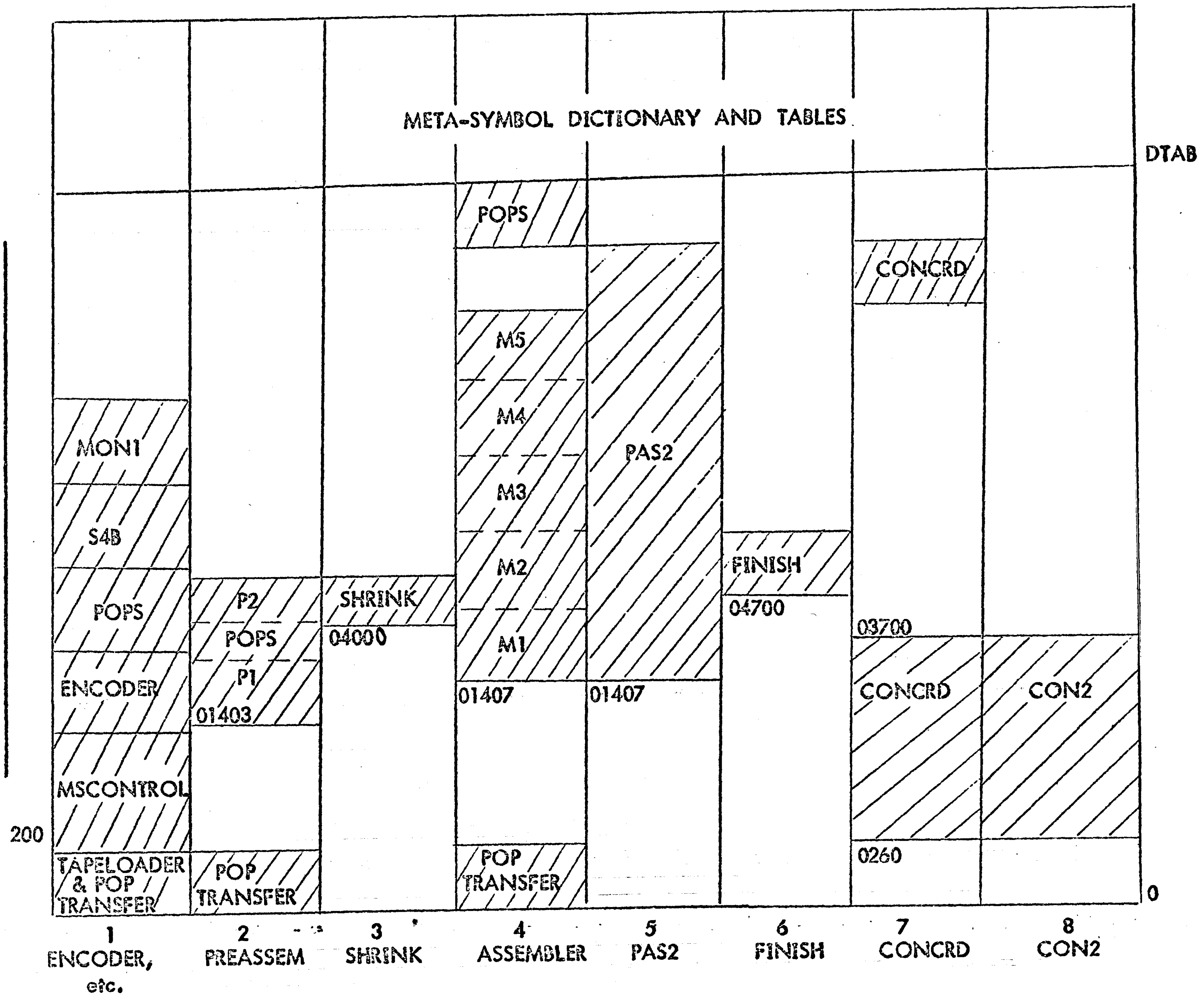
Jim MacIntyre

JM:cf

APPENDIX A - The Meta-Symbol Update Package

Δ1	METASYM
Δ2	ENCODER
	ENCODER BINARY (as assembled)
	910 or 920 POPS BINARY (as assembled)
Δ2	MON1
	S4B BINARY (as assembled)
	MON1 BINARY (as assembled)
Δ2	MSCONTROL
	TAPELOADER BINARY (as assembled)
	MSCONTROL BINARY (as assembled)
Δ2	PREASSEM
	PREASSEM ABSOLUTE (P1 + POPS + P2) loaded & dumped
Δ2	PROC910
	910 PROC (as assembled + machine ident. card)
	etc.
	•
	•
	•
Δ2	PROCB93H
	9300 BUSINESS PROCS
Δ2	SHRINK
	SHRINK ABSOLUTE (P1 + POPS + P2 + SHRINK loaded - SHRINK dumped)
Δ2	ASSEMBLER
	ASSEMBLER ABSOLUTE (M1 - M5 + POPS, POPS biased below DTAB) loaded and dumped
Δ2	PAS2
	PAS2 BINARY (as assembled less def cards (type 1))
Δ2	FINISH
	FINISH ABSOLUTE (FINISH + PAS2 DEFS) loaded & FINISH dumped
Δ2	CONCRD
	CONCORDANCE PT1 BINARY (as assembled)
Δ2	CON2
	CONCORDANCE PT2 BINARY (as assembled)

APPENDIX B META-SYMBOL OVERLAY STRUCTURES


META-SYMBOL DICTIONARY AND TABLES
DTAB
MON1
S4B
POPS
ENCODER
MSCONTROL
200
TAPELOADER & POP TRANSFER
P2
POPS
P1
01403
POP TRANSFER
SHRINK
04000
POPS
M5
M4
M3
M2
M1
01407
POP TRANSFER
PAS2
01407
FINISH
04700
CONCRD
03700
CONCRD
0260
CON2
0
1 ENCODER, etc.
2 PREASSEM
3 SHRINK
4 ASSEMBLER
5 PAS2
6 FINISH
7 CONCRD
8 CON2