3. INITIALISATION

3.1 Introduction

Once the Bootstrap has been written for a particular machine all future routines added to the system can be written as ITEM routines. As stated in the previous chapter, the macro orders are in a machine independent form and so the remainder of the Compiler-Compiler need only be written once. The ITEM routines added fall into well defined sections. Each section will produce a significant difference in the power of the system. The current section replaces the simplified form of the routines defined in the Bootstrap by their final forms and initialises the dictionaries of the system.

3.2 Deletion and Replacement of Items

The Bootstrap defines two Master Phases ITEM and REPLACE ITEM and provides an analysis routine capable of recognising these two phrases and passing control to the relevant routines. These are not the only basic Master Phrases that we would like in our dictionary and our first aim will be to replace the [MP] dictionary by one containing all the Master Phrases we intend to build in. Therefore our first task is to define the REPLACE ITEM routine (164) and its subroutine 155 which deletes an item from the store. Routine 155 has one parameter B61 which is the number of the routine requiring deletion.

The action of the REPLACE ITEM routine is straightforward. It first calls the ITEM routine which will compile the new copy of the relevant routine and store the old entry point in X165. This can then be deleted by calling 155 with B61 set to 165.

In general execution would then continue by the REPLACE ITEM routine returning control to the Master Routine which would look for the next

Master Phrase. However if the Master Routine itself has been replaced we would be returning to the deleted old version of the routine. Therefore this special case has to be tested for, in which case the END OF MESSAGE routine is called to reinitialise the compiler.

The routine 155 will delete the routine at index position B61 and return with the length of the deleted routine in B62. As the code used in the routines of the Compiler-Compiler are relocatable without alteration it is only necessary to move routines down over the space left by the deleted routine and alter the relevant Index positions. The only complication is due to the routine 155 itself being one of the routines that may need moving. The action of the routine is as follows:-

a. Find the length of routine B61. As no record of the routine's length is kept it is necessary to search the index positions to find the starting position of the routine following the routine B61 in the record store.

b. Move all routines higher up the record store down over the position held by this routine. If the routine 155 is below the deleted routine, the move is trivial. If above, then we must move all routines down including the first half of routine 155. Control is then passed to the front of routine 155 which moves the remainder of itself together with any routines above it in the store.

c. The Index Position for the deleted routine has its address field set to zero. The Index Positions for all the routines moved have their address fields decreased by the amount B62.

d. The routines 232 (Transplant) and 239 (Down) for Historic reasons have their entry points also stored in B74 and B76 respectively.

These must therefore be reset if either of these routines have moved.

The new [MP] dictionary will also define a Master Phrase:-

DELETE ITEM

This routine will set B61 to the routine number and enter 155 to delete this routine.

3.3 Dictionary Initialisation

A complete description of the structure of dictionaries in the Compiler-Compiler is given in the paper 'Trees and Routines.' The dictionaries are able to reside either in the Record Store or the Chain Store. The most commonly used dictionaries tend to be left in the Chain Store, while the remainder reside in the Record Store until alterations are required in which case they are moved to the Chain Store. The Analysis Routine is capable of analysing input with respect to dictionaries stored in either the Record or Chain Store. However some routines expect to find dictionaries in one or other of the stores and it is necessary for the Master Routine to ensure that the dictionaries are in their correct positions. Two supervisory lists are therefore provided which point to the dictionaries which are required to be moved:-

1. CONVENTIONAL LIST OF FORMAT CLASSES 129

   This contains a list of the standard format classes MP, AS, BS and SS provided by the system. This can be extended by the user.

2. LIST OF ADDITIONAL DICTIONARIES TO BE PACKED 169

   This is initialised to contain the Class Identifier Dictionary, (CID) 134 and the Double Entry List, 256 which stores the Compile versions of routines.

The Master Routine therefore moves all the dictionaries referenced by 129

and 169 to and from the Chain Store automatically.

Copies of all these dictionaries must be provided therefore before the final form of the Master Routine is added. The BS, AS and SS dictionaries are initialised to empty as is the Double Entry List 256.

We now have the routines required to allow us to replace the original form of the [MP] dictionary and the new version contains the following Master Phrases:-

PHRASE	218
ITEM	266
END OF MESSAGE	227
FORMAT	220
FORMAT CLASS	272
DELETE ITEM	145
REPLACE ITEM	164
DEFINE COMPILER	275
LIST	284
DO NOT LIST	285

The CID Dictionary is initialized to contain the following class identifiers:-

AS	132
ABN	154
AB	153
A	166
BS	131
B	167
PI	181
EOL	4
SS	133
SP	65
SEP	179
GENERATED-P	186
RESOLVED-P	185
LABEL	184

N	149	
COMMA	229	
0-3	178	(Probably redundant)
,	10	
ERAZE	127	
FD	251	
[	81	
OW	173	
MP	130	

3.4 Dictionary Packing and Unpacking

The routine 243 is added which will pack all the routines defined in the two lists described above into the Record Store. It uses a subroutine 228 which transfers an individual dictionary to the Record Store. The routine 245 and its subroutine 231 do the reverse operation.

The standard form of a dictionary in the Record Store is:-

M & ..... I K

where the dotted interior is described in Trees and Routines with the ampersands pointing at the category numbers. The keywords described in Trees and Routines were not implemented. Instead a single K word giving the maximum number of ampersands on the top level of any alternative is added at the end. We have a similar form in the Chain Store with the Index entry pointing at the M word and the link from the K word pointing back to the M word.

The subroutine 231 is entered with B69 pointing at the initial ampersand and B68 at the M word. It will convert all information between and including the outer ampersand and I word to chain form with B67 pointing at the ampersand on return. In addition B66 is left pointing at the K word. The routine 245 works through all the dictionaries in both lists; filling in the M and K words itself. The action for a Double entry list is different. This conversion is done entirely by

245. In the record store the first entry of the dictionary contains length of list followed by a word containing o and then the words of the dictionary. In the Chain Store only the words of the dictionary appear with the Index position pointing at the last word.

The action of 243 and 228 is similar but provides the reverse operation. The flowdiagrams in the appendix give the details of these routines.

3.5 Analysis Routine

The Analysis Routine 215 is the central routine of the Compiler and provides all the mechanism for the syntactic scan. In the appendix are two flowdiagrams. One gives a complete description of the routine while the second gives a simplified form of the routine showing only the main flow. Its purpose is to analyse a string of characters with respect to some class of phrase or dictionary and produce on recognition an analysis record showing how the scan proceeded.

The input parameters to the routine are:-

B61 = address of string to be analysed. This must be in the form of a circular chain with B61 pointing to the last word.

B62 = class number with respect to which the analysis is to take place. The dictionary or phrase may be either in the Record or Chain Store.

B63 = 0 for usual entry

1 for a special entry

2 for re-entry

In its use during the bootstrapping of the compiler, the special and re-entry modes are not used. Consequently discussion of these will be left until later.

In the case of a Phrase being defined as 'Contract Record,' the analysis record is delete . This will be used when the analysis record contains no useful information. In the dictionary this is signified by the M word having the J4 bit set. All tests on J4 deal with contracting out records in the routine.

Another section of the Analysis Routine which can be discussed later in more detail is that involving parameters. It is possible that when we are reading in genuine Compiler-Compiler routines that the input stream may contain phrase identifiers. The section dealing with this (bottom left of flowdiagram (ii)) can be left for the time being.

Looking at the simplified flowdiagram (i), the main paths are as follows:-

1. For basic symbols a test is made between the character in the dictionary and on the input stream. If a match is found we move down both checking until we get to the end of an alternative. This is known, as an ampersand always points to the category number at the end of an alternative and so we check until we reach this point. If a failure occurs then to get to the next alternative we must reset the original position of the input pointer, find where the last ampersand pointed and reset the pointer to the analysis record being produced. This information is made available by storing in the stack on arrival at any ampersand the three quantities B62, B63 and where ampersand is pointing. Getting to the next alternative is then simply a matter of unloading the top three stack positions.
2. If the dictionary or phrase contains phrases in its definition as well as basic symbols then when this appears in the dictionary we must suspend our present search on the current level and

virtually reenter the Analysis Routine looking for the recognition of the sub-phrase. This could have been implemented in this way. However for efficiency reasons the stacking of levels is done internal to the routine. On descending to attempt recognition of a sub-phrase we store on the stack the origin of the stack for this level (B69), the pointer in the analysis record (B63), the pointer to the dictionary (B67), the M word (B66) and the origin of the dictionary (B65).

If recognition at the sublevel is achieved we reset the above values and carry on where we left off. If no recognition then we must apart from recovering these values also go on to next alternative.

3. On recognition at the top level we must split the recognised and unrecognised parts of the input (Routine 252); change the Analysis Record to its DUAL form (Routine 222) and exit.

The output parameters are

B61 = pointer to unrecognised part of input.

B62 = pointer to recognised part.

B63 = pointer to analysis record.

B64 = 0 for no recognition

= 1 for recognition

> 1 for maybe

A flowdiagram for the Dual routine is given in the appendix. A complete description of analysis records is given in the Trees and Routines.

3.5 Fault Routines

The Analysis Routine and most other routines of the system on recognition of fault conditions enter one of the two fault routines. If some kind of recovery is possible from the error then Routine 258 is entered otherwise 259.

Both routines require a fault number to be set in B65. The action of these routines is mainly up to the implementation. The fault number together with the routine number (B75) give a good indication of the fault. Additional information that could be printed would be the contents of the B variables, the stack and other parts of the Record and Chain Store as desired.

3.6 Master Routine

The Master Routine provides the implicit control structure of the Compiler-Compiler. There are instructions available in the Compiler-Compiler which allows the user to override the implicit control but, until these have been implemented, all control on the global level is provided by the Master Routine. This can be expressed briefly as follows:-

1. Form line of input and attempt to recognise Master Phrase.
2. If successful enter the relevant routine and on return repeat the process from 1.
3. If unsuccessful attempt to recognise a Source Statement.
4. If successful enter relevant routine and on return repeat from 3.
5. If unsuccessful try and recognise a Master Phrase again.

Safeguards are of course added to make sure that we do not get into an infinite loop. The Master Routine (i) flowdiagram shows the flow for steps 1 and 2 above. Note that the line of input has a subscan by Routine 261 before entering the Analysis Routine. This is mainly for the removal of spaces and the setting of unique basic symbols for the meta-syntactic symbols '[' and ','. On Atlas different input forms used different representations for the open bracket and it was helpful to have a unique form being passed around. On the G21 these have still been given unique

values although I am still not clear how necessary this is now. The second main point is that some routines called on recognition of Master Phrases assume that the dictionaries reside in the Chain Store and so these dictionaries must be put there (if not already there) before entering routine.

The Master Routine (ii) flowdiagram gives the flow for steps 3 and 4 above. This is very similar to the flow for recognition of Master Phrases. A separate subscan routine 142 is provided and the input is passed through this before calling the analysis routine 135. Routine 135 is basically identical to Routine 215. On Atlas it is compiled in a non-relocatable form for efficiency purposes and also produces the Dual form of the analysis record directly. (This is the reason for the K word on dictionaries.)

The other main difference between the two flows is that there is an assumption that the Master Phrase can be recognised with the available input. In the flow for Source Statements it is possible to get a 'maybe' recognition which will reenter the Analysis Routine after processing further input. The Routine 135 on Atlas will only analyse with respect to dictionaries in the Record Store (again for efficiency). Consequently dictionaries must be packed into Record Store the first time we look for a Source Statement.

A little care must be taken when replacing the Master Routine. As stated before the REPLACE ITEM routine will call the END OF MESSAGE routine in this case. Just before replacing the Master Routine an interim version of the END OF MESSAGE routine is added. This reenters the Master Routine as though it had been called from Routine 150. Once the Master Routine has been replaced the final version of END OF MESSAGE is added. This packs up dictionaries into the Record Store. This was not required originally as the unpacking had not been done by the original Master Routine due to Routine 245 being replaced by a dummy.