MASTER IN COMPUTER SCIENCE

An implementation of 3-graphs : first version

Hewitt, M.

Award date:
1976

Awarding institution:
University of Namur

FACULTES UNIVERSITAIRES NOTRE-DAME DE LA PAIX - NAMUR

INSTITUT D'INFORMATIQUE

ANNEE ACADEMIQUE 1975-1976

AN IMPLEMENTATION OF 3-GRAPHS : FIRST VERSION

*****

M. HEWITT

Mémoire présenté en vue de l'obtention du grade de Licencié et Maître en Informatique.

I would like to take this opportunity to thank Mr. Claude Cherton for all his help over the past three years, and especially for having taught me how to program a computer. His help and encouragement have been invaluable.

My thanks also to all my friends at the Institut d'Informatique and to Mme Labidi who typed this report.

C O N T E N T S

1. INTRODUCTION

This report describes a first version of a set of programs designed to implement 3-graphs on a Siemens 4004/151 computer. As this set of programs has developed, so has the theoretical side of the work, with the result that the former lags somewhat behind the latter. However, as we hope to show, the necessary modifications should be relatively simple to carry out and localised in their effect, due to the way the programs are organised.

In writing these programs we have exploited certain features of the computer system on which they run. These will be explained where necessary and the alternative in a system not presenting these features outlined.

We will first give a brief description of 3-graphs. For a more complete description see [1].

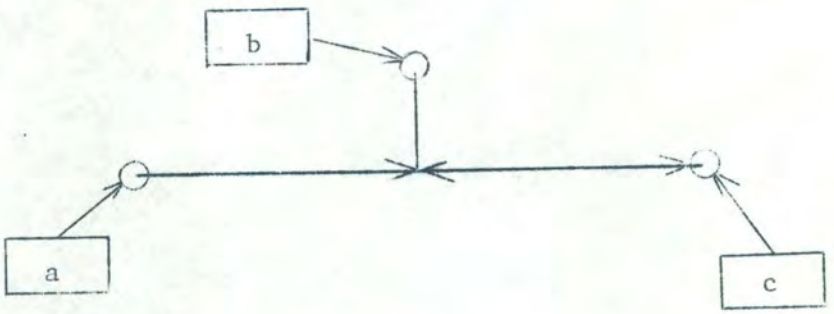
1.1. 3-graphs

3-graphs are an attempt to model the concepts underlying the words and phrases of a language. These concepts are the end results of the processes of filtering and assimilation of all the information reaching an individual. They are modified continually by new information and also by internal transformations.

This evolution of concepts over time implies that an individual will attach different "meanings" to the same word or phrase at different periods, and that two individuals with different backgrounds will also attach different "meanings" to the same word or phrase. Despite this, the underlying structures are sufficiently similar for them to be expressed using the same word or phrase and thus to allow communication to take place between individuals. We believe that it is possible to abstract such structures from the words or phrase used to express them, and thus to model semantics. The 3-graph model is an attempt to do this in order to study the structures and the processes of modification that cause them to evolve.

A 3-graph is a set of "notions" structured by a ternary relation. The structure represents the semantics of the concepts modelled. If we represent the relation between three notions (represented by a, b and c respectively) by the triplet < a, b, c> we say that the notion represented by a is the "root", the notion represented by b is the "nature", and the notion represented by c is the "value", of an "attribute" Any notion can potentially play any one of the three roles in different attributes (or even in the same attribute e-g. < a, b, c>, < a, a, a> etc are not excluded). We also say that an attribute whose root is the notion represented by a is an "attribute of the notion a". The semantics of the notion is then described by the whole structure in which it is embedded, by all the attributes to which it is directly or indirectly connected via other attributes.

The attribute <a, b, c> can also be represented graphically, as follows



The boxes containing the letters designating the notions are used for convenience of identification only and do not belong to the structure.

Any static structure can be represented using one of the above conventions, but a characteristic of such structures is their ability to evolve in time, an ability which is built-in. In the 3-graph model this dynamic aspect is provided by distinguishing a certain number of notions having intrinsic semantics and thus an existence independent of their relations with other notions (that is, their role as root, nature or value in attributes) in the 3-graph. These notions are a set of "basic actions" plus certain notions used to specify any particular instance of one of these.

The basic actions are :

COMP - comparison of two notions

CREN - creation of a notion and an attribute where the new notion is a value

CRER - creation of an attribute

LAT - "load an attribute"

SUP - suppression of an attribute specified by root and nature

SUPR - suppression of an attribute specified by root, nature and value

TMV - a value of an attribute is "transmitted" to another attribute

REP - a value of an attribute is "replaced" by a value of another attribute.

The other "special" notions are :

OCCDE - specifies which action the notion is an occurrence of.

SUC1, SUC2 - designate the possible successors of a notion

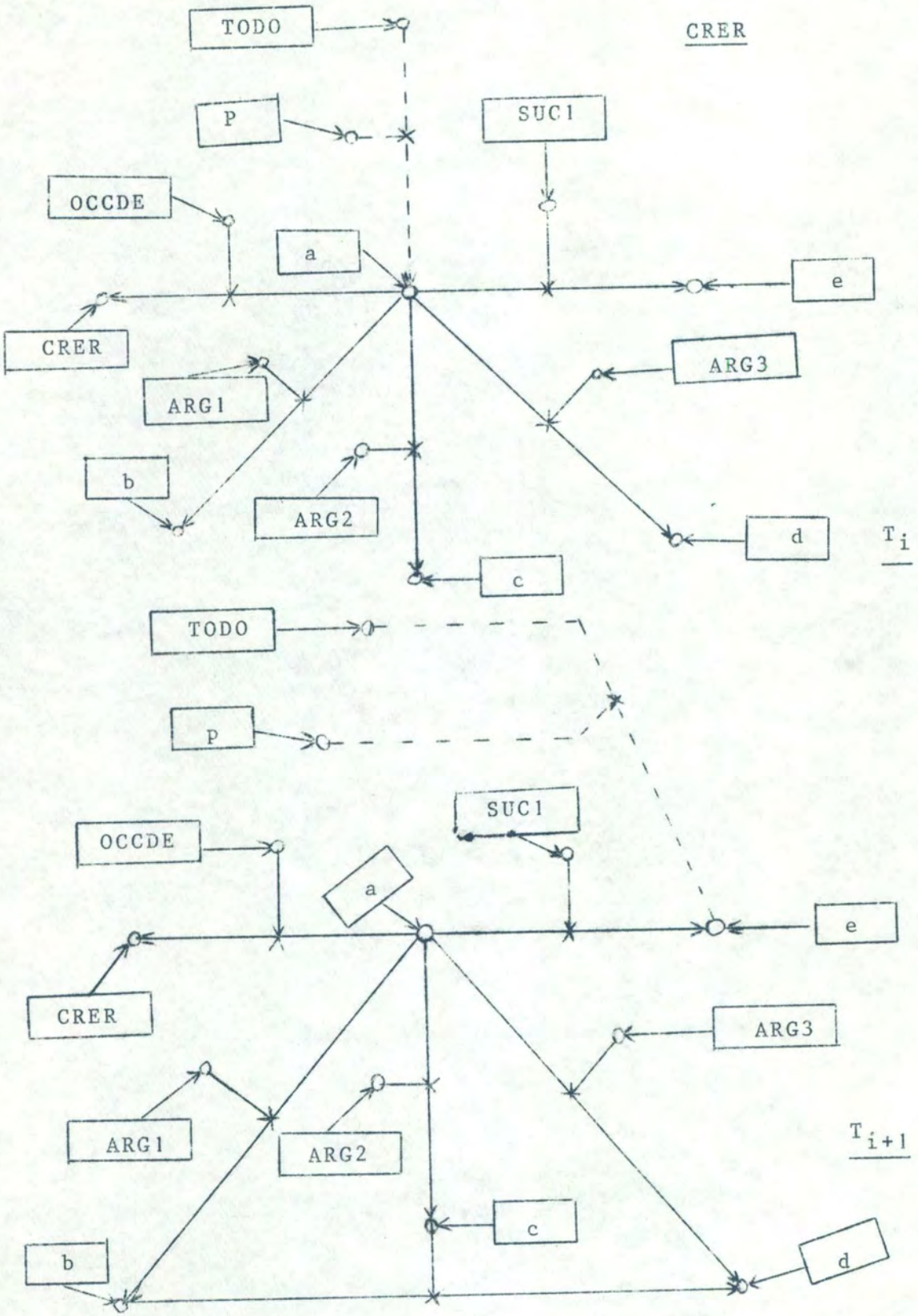
ARG1, ARG2, ARG3, ARG4 - designate the arguments of an occurrence of an action

For example : < a, OCCDE, SUP>
< a, SUC1, b>
< a, ARG1, c>
< a, ARG2, d>

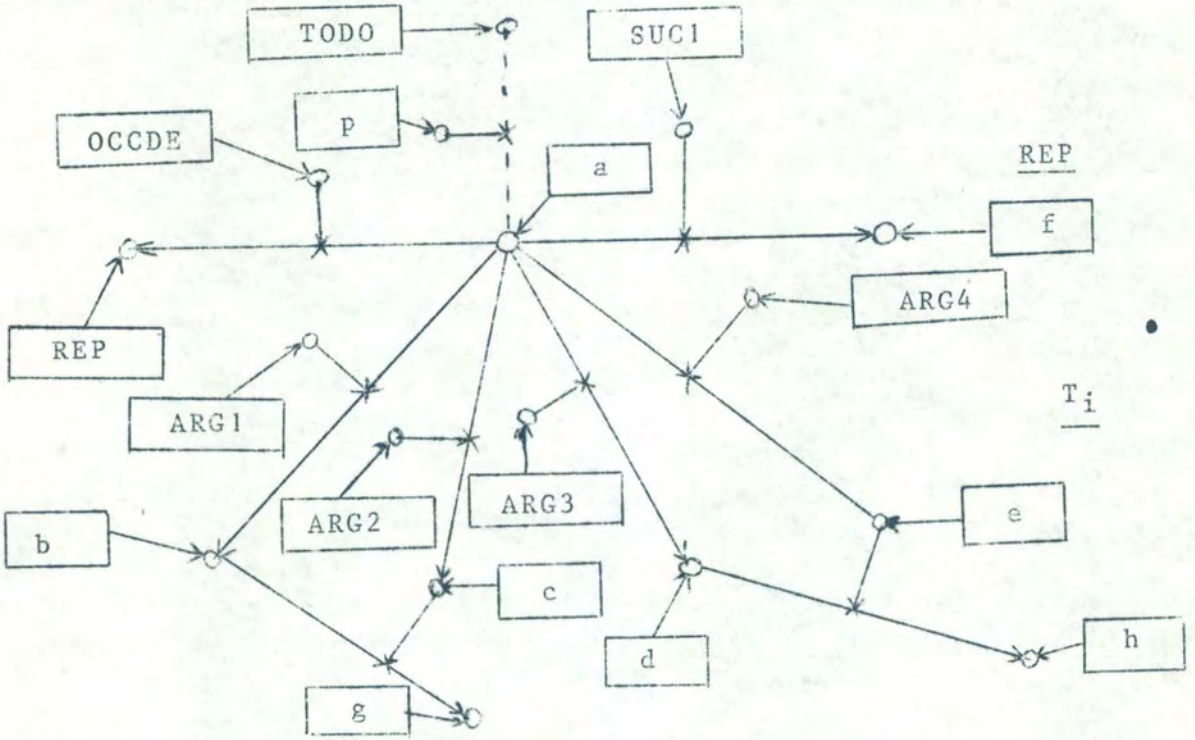
specify the notion a as an occurrence of the basic action SUP.

For detailed descriptions of the basic actions COMP, CREN, LAT, SUP, SUPR and TMV see [1]. The action CRER creates a new attribute with three existing notions as root, nature and value respectively. REP replaces a value of an attribute specified by root and nature by a value of another attribute also specified by root and nature (effectively suppressing an attribute and creating a new one with same root and nature).

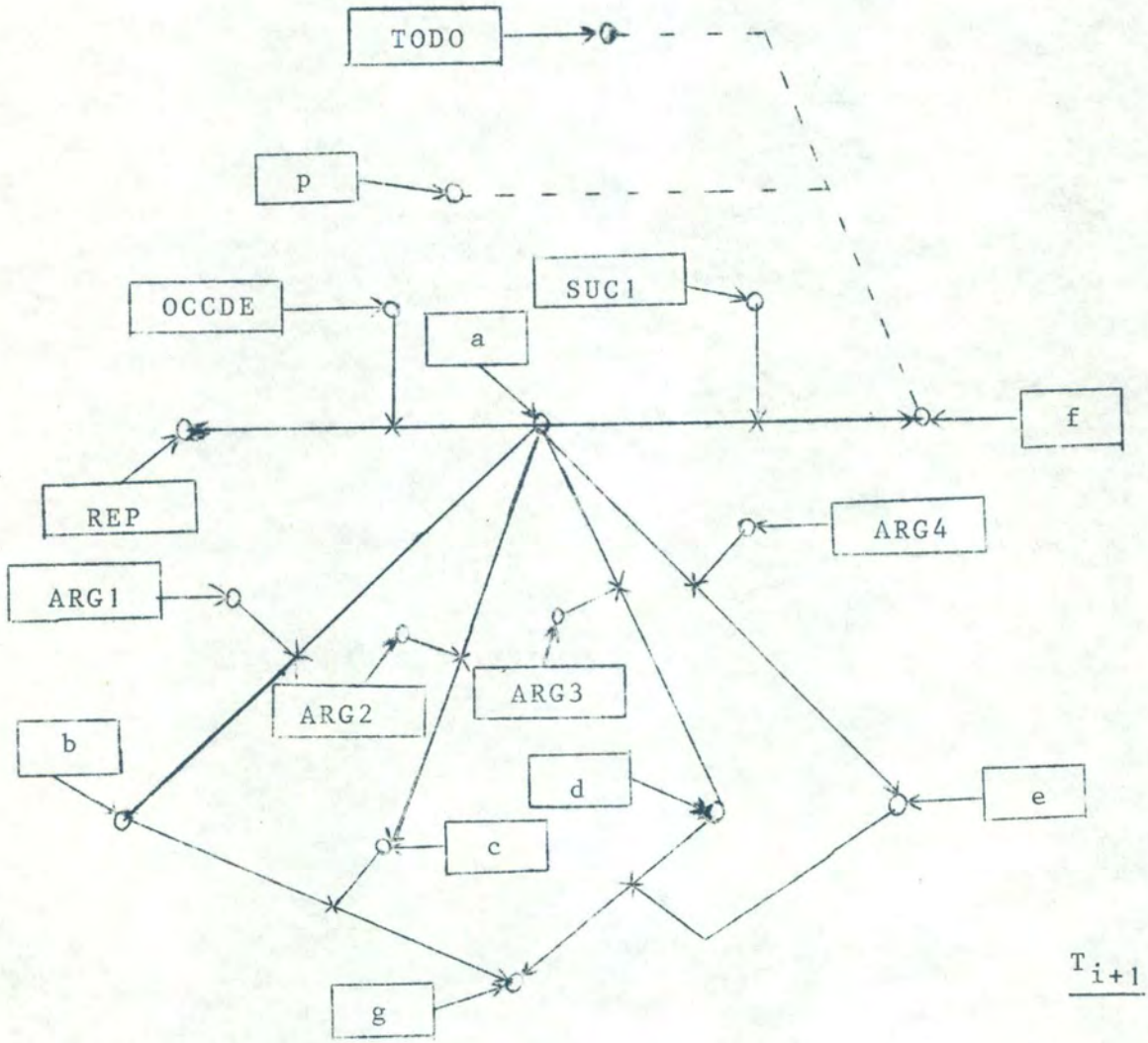
Using the same conventions as in [1], CRER and REP are defined as follows :



At time T_{i+1} the attribute < b, c, d> has been created and the notion e is the next action to be executed.



T_i



T_{i+1}

At time t_{i+1}, the attribute

$$< d, e, h >$$

has been replaced by the attribute

$$< d, e, g >$$

If the notion h is no longer root, nature or value of any attribute it is removed from the 3-graph. The notion f is the next action to be executed.

As in [1] the notion 'NIL' signifies ' nothing' and may be considered to be root, nature or value of any attribute not otherwise defined (non-significant relations). The notion 'TODO' does not exist in the implemented version of 3-graphs described here, therefore the set of actions to be performed at any time t_i is not explicitly shown. (For consistency in the descriptions it has been shown dotted).

The actions REP and CRER were added to the set of basic actions for convenience in describing the processes transforming 3-graphs. They are useful in practice but do not add to the power of the model, that is, to the set of processes that can be described with these actions.

1.2. Why implement the model

Two ways of representing 3-graphs on paper have been used so far in this report. Both methods are, however, far too cumbersome to be of much practical use in studying semantic structures by means of 3-graphs. Even simple structures quickly become very difficult to represent clearly by either notation when considering only the static aspect. It becomes even more difficult to represent the dynamic aspect, not only because of the difficulty in presenting the structures clearly but also because of the conventions required to represent a transition in a static medium.

The storage capacity of modern computers permits the description of extremely complex structures that it would be impossible to contemplate attempting with pen and paper only.

For the study of the dynamic aspect the computer is an ideal tool, providing :

a) both the static and dynamic aspects can be represented in a way comprehensible to the computer, and;
b) the computer has been instructed how to effect the transformations corresponding to the dynamic aspect.

It is evident that we need two new ways of representing 3-graphs :

- the representation in a computer;
- a representation for input to the computer.

Ways must be provided for the computer to translate the latter into the former and for it to interpret those parts relating to the dynamic aspect and execute them. (After this, the computer can be used as a tool for the study of semantic structures) The implementation described here aims to provide those ways.

Two points concerning the implementation should be noted :

- the theory of 3-graphs continued to evolve while the implementation was being developed. In order to keep abreast of changes in the theory, both actual and future, it must be highly adaptable. This necessity has conditioned the implementation from the start, in particular, the programming style and the method of documentation, both intended to simplify maintenance and updating;
- the structure of present day computers does not lend itself to the representation of a ternary relation. This can only be represented indirectly and this fact has also conditioned the representation.

□□□□□□□□□□□□□□□

From now on, we will call the representation in the computer the internal representation. The representation for input to the computer we will call the external or TGL representation. We will also touch on a third system of representation

associated with the computer, that used as output based on the internal representation.

The next section describes the internal representation and the impact of certain features of the computer system on which it is implemented. Subsequent sections will describe the TGL representation, the choice of programming language and the programs written to carry out points a) and b) above.

°
° °

2. THE INTERNAL REPRESENTATION

A 3-graph could be represented in the computer in many ways, some of which would emphasize the roles of the different notions in the attributes. Others would emphasize the attributes, leaving the notions with a totally secondary role in the representation. The choice of a particular representation is influenced by two considerations :

- the amount of space it would occupy;
- the ways in which the structures will be accessed, in particular for the simulation of the dynamic aspect.

When we look at the descriptions of the basic actions we note certains characteristics :

1) the arguments of an occurrence, that is, the attributes to be created or suppressed are specified :
   - in every case by their root notion
   - often by both their root and nature notions
   - only occasionally by their root, nature and value notions.
2) In two of the basic actions, COMP and CREN the arguments of an occurrence are notions rather than attributes.
3) A notion is identified as an occurrence of a basic action because it is the root notion of an attribute with nature OCCDE. What is to be done is specified by a set of attributes whose natures are some of the special notions : ARG1, ARG2, ARG3, ARG4, SUC1, SUC2. These characteristics lead us to emphasize notions rather than attributes.

In a given 3-graph, only a few of the possible attributes will actually be present, so that a representation which enumerates the attributes that are present, seems a good choice. This can be achieved by enumerating the root, nature and value notions of each one.

From the description of the basic actions, we can see that paths through the 3-graph to be followed during a simulation lead from a root notion to nature and value, or from root and nature to a value. This leads us to sort our enumeration on the root notion of attributes and then on the nature of the attributes. The root notion does not now need to be explicitly present in each attribute's representation.

Since nearly all the basic action occurrences involve the creation or suppression of attributes, easy manipulation of these is important. So our internal representation must take the form of a list structure.

In the 3-graph it is not excluded that a set of different attributes may exist, where the root and nature are always the same and only the values differ.

e.g. < a, b, c >
< a, b, d >
< a, b, e >
< a, b, f > etc ...

Representation of this situation in the strict pair-list representation would complicate the search for an attribute or a place for a new attribute unduly. It is easier to group such attributes "under" the same nature within the root notion grouping and the nature-value pair becomes a nature-pointer pair with the second element pointing to a secondary list containing the grouped notions that are values of the attribute.

One further grouping under the root notion is to be considered. This is also concerned with simplifying the search for particular attributes. Any notion can be or can become an occurrence of an action, and the identification of such a notion would be considerably simplified (and speeded-up) if the attributes defining it as an action occurrence were grouped, say at the beginning of the list of attributes. For an occurrence of a basic action, the attributes concerned are those with nature

one of the notions OCCDE, SUC1, SUC2, ARG1, ARG2, ARG3, ARG4, through these are not all significant for every basic action. However, if we are to be certain that these attributes are always the first when they are present they must be "potentially present" in every root notion's list of attributes. This can be achieved by reserving the first seven places in every list. The loss of space involved can be reduced by identifying the attribute by position in the list, its nature need no longer be explicitly mentioned leaving only the value to be identified.

Every root notion's attribute list begins with seven spaces reserved for the values of the attributes with nature OCCDE, SUC1, SUC2, ARG1, ARG2, ARG3, ARG4 in that order. These are followed by the nature-value pairs of other attributes. The space reserved for a value will contain 'NIL' if the attribute is non-significant, otherwise it will specify a notion or point to a list of notions that are values of the attribute.

2.1. Representation of a notion

We have not yet said anything about how a notion is to be specified when it is nature or value of an attribute or how we locate a notion.

When a notion exists in the representation, space has been reserved for the first seven "special" attributes, even if the notion has no attributes of any sort. The address, within the space occupied by the 3-graph, of the beginning of this list identifies the notion uniquely while it exists. This address is used to identify the notion when it is nature or value of an attribute of another root notion.

2.2. Page organisation

While the internal representation described so far is not in any way limited by size or space considerations they are important and they will affect the representation of the 3-graph in the computer. During the process of modification of a 3-graph (evolution of the 3-graph) the actual part involved at a given

instant in time is likely to be small in relation to the size of the whole. Practical considerations of memory size and its use lead to the idea of cutting the 3-graph into blocks that could be stored in auxiliary storage and brought into memory only when needed. Whenever an attribute is required, we would need to know whether the root notion's attribute list is already in memory or needs to be brought in from auxiliary storage. This would mean copying one or more blocks from auxiliary storage. A block already in memory might have to be ejected to make way for a new one, involving problems of updating the copy in auxiliary storage and the choice of the page to be ejected.

The actual implementation on the Siemens 4004 does not do any of this explicitly. Instead, it makes use of the virtual memory environment of the computer. This provides a program with up to 1M bytes of contiguous virtual memory, divided into pages of 4K bytes. The minimum number of pages required is determined when the program is loaded, and can be extended during execution (up to a maximum of 256). A copy of each page of the program is stored on magnetic drum storage and copied into memory when needed. A paging algorithm chooses the page which may be ejected from main memory, if necessary. If the contents of the page have been altered, they are copied back to drum, before the page is overwritten.

Not only does this save programming time since the equivalent routines are not included in the implementation but at execution time the computer uses hardware when paging and can thus go much faster than could a program that does the same thing.

The internal representation of the 3-graph is divided into 4K-byte pages and loaded as a program, leaving it to the paging system to ensure that the pages are in memory when required.

Since the original 3-graph is unlikely to be exactly an integral number of pages long, and during its evolution the space used will grow and contract, any unused space in each page is held in a free list from which it can be allocated when new notions are created or new attributes added, and to which space can be returned when attributes are removed. If the 3-graph grows beyond the original load-time allocation of pages, new pages can be acquired dynamically.

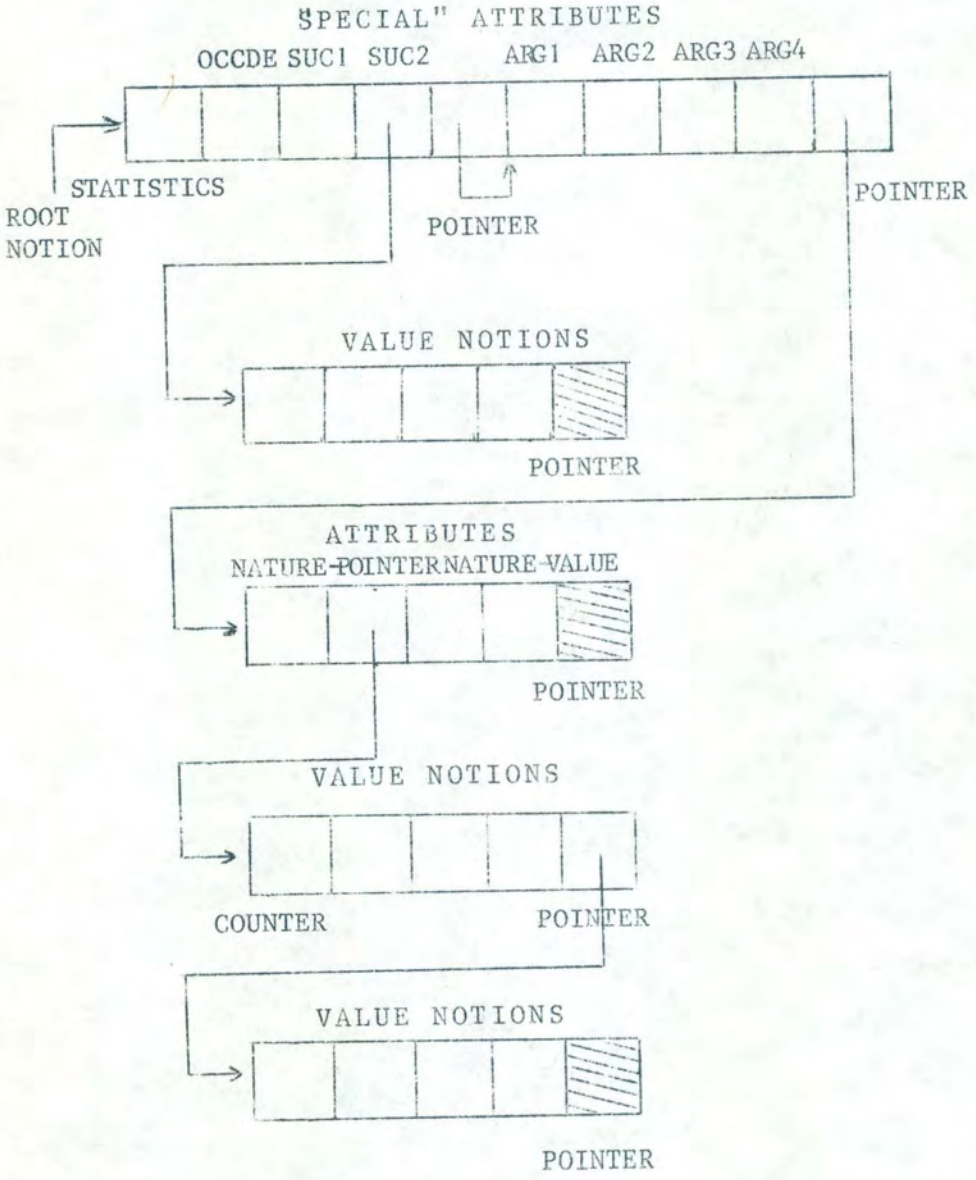
A notion is allocated the minimum space for the seven attributes OCCDE, ARG1, ARG2, ARG3, ARG4, SUC1, SUC2 when it is created. If more space is required, the minimum needed is usually for a nature-value pair specifying a new attribute. But this must be linked to the existing list of attributes. Allocation of this minimum unit is likely to be expensive in space since the proportion of links to information is large. The time needed to scan through the links when searching the list of attributes is also likely to be costly. It is more economic to allocate more than the minimum and so reduce the number of links, even though there is the risk that some of the space may never be used.

2.3. Attribute list organisation

A notion in the 3-graph is represented by the address of its first space allocation. An address occupies 3 bytes in the computer but this is not an easy unit to handle because the instruction set is word or half-word oriented so our basic unit is the computer word. The address occupies the 3 right-hand bytes of a word leaving the left-hand byte free for indicators or flags. Two words or cells are needed to specify a nature-value or nature-pointer pair with a flag to distinguish one from the other. For the special attributes a single cell is sufficient to specify the attribute, so the minimum primary allocation is 8 cells - 7 plus a pointer. There after the minimum allocation is 3 cells - nature-value or nature-pointer + pointer. Not only is this expensive as we have already noted but handling blocks of different lengths poses problems of allocation or realloca-

tion (dividing a block of 8 into blocks of 3 means you "lose" 2, regrouping poses similar problems), and also in scanning lists of attributes. It is easier to have blocks of a standard size that can be re-used anywhere, greater than the minimum allocation and with a pointer to its successor in the same position in every block.

We have chosen to allocate space for 2 nature-value or nature-pointer pairs at once. With the pointer to a succeeding block, we have a block of 5 cells (20 bytes). Thus, when a block is allocated, at most one pair of cells remains unused in any block. Two blocks are sufficient to contain the seven special attributes leaving one empty cell. This size of block is also used in the secondary lists. The organisation of different blocks is shown below :



The exact contents of each cell is indicated by its position in the block and by flags in the cell itself. The value cell of an attribute pair specifies the notion which is value of the attribute except when a flag is set to indicate that it points to a secondary value list. Other flags indicate, for example, that the cell reserved for the attribute with nature OCCDE specifies a basic action occurrence.

The unused blocks are chained together by their pointers and the address of the first block in the chain is kept in one of the four words left at the end of each page. Blocks are added to or removed from the chain last-in first-out and a record is kept for the whole 3-graph of the pages where space is available.

2.4. The statistics cell

In the two blocks allocated for the special attributes of a notion only 9 cells are used leaving a spare cell. The special attributes are so positioned that this is the first cell in the first block, which means that its access is immediate when the root notion is accessed. It is used for the storage of certain information about the notion's role in the 3-graph. It contains a count = n + v

where n is the number of times the notion is nature of an attribute, and

v is the number of times the notion is value of an attribute.

In the implementation this information has an important practical application : when this count becomes zero the space allocated to the notion is returned to the free list, in effect the notion is suppressed.

2.5. The counter cell

The counter cell is the first cell in the first block of a secondary list. It contains a count of the total number of cells, used or unused, in the list and is used in the choice of a value from the list, as we shall see later.

2.6. Empty block indicators

The ability to free unused blocks whenever possible is important for the most efficient use of the space available. Blocks may become available when an attribute is suppressed and as a consequence, one or more notions that are nolonger attached to the 3-graph, because they are no longer nature or value of any attribute. Note that, a notion may be suppressed even though it is still root of an attribute. In [1], only notions that are "isolated" are suppressed, substructures that are "inaccessible" (see [1]) from the rest of the 3-graph would remain. In the implementation such substructures will disappear.

As a general rule, when an attribute is suppressed, all blocks freed are returned to the free list. However, during the transformation process, it is possible that certain notions will acquire and lose attributes rapidly and frequently. This could lead to repeated allocation and removal of blocks from the same notion, a situation to be avoided if possible. The counter and statistics cells contain flags that indicate the presence of an empty block in the list. When a block in the list becomes empty, it is returned to the free list only if the flag indicates that there is another empty block in the attribute or secondary list.

2.7. The special notions

The eight basic actions and seven special notions plus the notion NIL are always "known", whether or not they are attached to the rest of the 3-graph. They are never created or suppressed and normally, they are not root of any attribute. For this reason only the seven special notions - OCCDE, ARG1, ARG2, ARG3, ARG4, SUC1, SUC2 - are physically present in the 3-graph like other notions. They are the first notions specified in the 3-graph's address space.

In the next section we will describe how the basic actions are specified. The notion represented by 'NIL' is represented in the 3-graph by a high value that cannot be confused with the address of any notion.

2.8. Simulation of the dynamic aspect

We have seen how a static 3-graph structure is represented in the computer and how this has been influenced by the basic modifications - location, addition and removal of attributes - that are necessary for simulating the dynamic aspect. From this internal representation those elements defining the process of transformation must be extracted. The attributes of a root notion that identify it as an occurrence of an action modifying the 3-graph must be interpreted and the corresponding modification carried out.

2.9. Identification of actions in the 3-graph

We distinguish two kinds of actions transforming the 3-graph :

- the eight basic actions : COMP, LAT, CREN, CRER, SUP, SUPR, TMV, REP
- non-basic actions.

If such a thing as a '3-graph computer' existed, the basic actions would form the machine's basic instruction set. The non-basic actions would have to be interpreted as a combination of these basic actions to be directly executable by the machine.

The basic instruction set of this mythical computer is simulated in the current implementation by a set of routines. This set of routines, which we will call the "simulator" is linked with the original 3-graph in internal representation, and the result loaded as one pageable program in the virtual memory environment. The links between the simulator and the 3-graph are established via the OCCDE attributes (☆) of the notions in

---

(☆) When we wish to refer to an attribute by specifying the notion that is nature of the attribute, we will say nature name attribute, for example, OCCDE attribute here.

the 3-graph. These links identify the actions to be simulated in any 3-graph evolution process.

A flag, used only in the cell reserved for the OCCDE attribute indicates that a notion is an occurrence of a basic action. This flag is not set for a non-basic action and the cell contents specify a notion. At present, the interpretation of these is not part of the simulator. The simulator merely passes on to the successor of a non-basic action which thus corresponds to a no-operation. In this way, with the use of the "basic action" flag, it should be possible to add to the list of basic actions or to include interpretation of non-basic actions in the simulator at a later stage. However, we will usually ignore non-basic actions in this discussion. From now on, action is to be taken to mean basic action. When the flag is set, the cell contents identify the routine that simulates the action.

Although, it is theoretically possible, (see [1]) for a notion to be an occurrence of several actions at the same time, the current version of the simulator excludes this possibility.

Once the action has been identified, we distinguish three stages in its simulation :

- the arguments involved must be chosen;
- the action itself can then be simulated;
- the successors to this action must be determined.

This distinction between the second and third stages is not always clearcut as we shall see.

2.10. Choosing the arguments

The possible arguments are the values of the attributes ARG1, ARG2, ARG3, ARG4 of the occurrence. Not all of these attributes are required in every action, for example, CREN needs only ARG1, ARG2. The attributes not required may have values but they are ignored in the simulation. If one or more of the attributes required are non-significant, that is, the value is the notion 'NIL' then the action is null and the only result is to determine its successors. There is one exception to this rule : a notion can be compared to 'NIL' by COMP. If one of the attributes has several values (the cell contains a pointer to a secondary list of notions), a value is chosen "at random" to be the argument used in simulation of the action's current occurrence.

2.11. Simulation of an action

The last two sections described how an action occurrence can be detected and identified so that the correct simulator routine is invoked, and how this routine first determines its arguments. Table 1 shows the main functions needed to simulate each action. Certain of these functions call on other functions such as allocation of a block or return of a block to the free list. We will discuss each in turn and the conditions in which it is used.

Neither Table 1 nor the following discussion indicate all the separate functions used in the simulation of action occurrences. In general, each function isolated logically is also isolated physically in the simulator routines. The programming is to a great extent modular with the different functions communicating only by the passing of parameters. Each separate routine has, with few exceptions, one entry and one exit point and performs one function. This function may be a complex one, in which case the routine will issue a series of calls to others that perform simpler sub-functions, and so on to the simplest sub-functions that are performed directly.

These last sub-functions are the only ones that deal with the physical representation. The aim is to facilitate modification in two ways :

- the logic of the simulation of the basic actions can be altered or added to since it depends on the sequence of function calls;
- the physical representation can be changed and only one level of subfunction will be affected.

This method of programming is designed to simplify the maintenance of the simulator and to allow modifications and extensions to be made fairly easily. As we indicated in section 1 the implementation must be highly adaptable in order to keep abreast of modifications in the theory of 3-graphs.

2.12. Comparison of two notions

A notion is uniquely identified by its address. COMP compares the two notion addresses that are its arguments and sets the choice of successor depending on whether they are equal or not.

2.13. Scan of a root notion

With the exception of COMP and LAT, the arguments chosen specify the root and nature of an attribute and in the cases of CRER and SUPR the value of an attribute. This attribute must be located in the root notion's attribute list. The absence of this attribute often means that the effect of the action is nullified, see table 2. When the nature of the attribute is one of the special natures, (OCCDE, etc ...) the attribute is located by position. For other attributes the chain of blocks is scanned and the contents of the first cell of each nature-value pair is checked against the nature specified until there is a match or the end of the chain is reached.

This function is also used to find an unused nature-value pair for the creation of a new attribute. Unused cells always contain a value conventionally used to specify the notion 'NIL'.

When blocks are allocated, all cells except the pointer cell are initialised with this value and when the contents of a cell are removed the cell is reinitialised to 'NIL'. The search for an unused nature-value pair is thus a search for an attribute whose nature is the notion 'NIL'.

2.14. Scan of a secondary list

When the attribute has been located by its nature in the root notion's list, the second cell may point to a list of associated values.

For the CRER and SUPR actions, this list must be scanned block by block and each cell checked against the value of the attribute specified by the third argument. The effect of the action depends on the absence or presence of this value, see Table 2.

This function is also used to locate an unused cell in a secondary list when adding a new value in creating a new attribute.

2.15. Choice of an attribute at random

This function is specific to simulation of the action LAT. The number of nature-value pairs allocated to the root notion specified by the first argument is determined by a rapid scan of the list. A number, a timer obtained from the system, is divided by this count to generate a "random number" (the remainder) which is used as the ordinal number of a nature-value pair. The attribute is then located.

Should the attribute be non-significant a second and then a third number can be generated to locate a significant attribute. If even these fail, the first non-significant attribute to be found in a systematic scan will be chosen. The preliminary counting scan also ensures that the root notion possesses at least one attribute.

TABLE 1.

	COMP	CREN	CRER	SUP	SUPR	TMV	REP	LAT
Comparison of notions	X							
Scan of a root notion to find an attribute		X	X	X	X	X	X	
Scan of a secondary list to find a value			X	X	X	X	X	X
Choice of an attribute at random								X
Choice of a value at random				X		X	X	X
Creation of a notion		X						
Addition of an attribute		X	X			X	X	X
Addition of a value		X	X			X	X	X
Suppression of a notion				X	X		X	
Suppression of an attribute				X	X		X	
Suppression of a value				X	X		X	

TABLE 2

Effect of action

	CREN	CRER	SUP	SUPR	TMV	REP
1st attribute : root, nature absent	ATTR. CREATED		NULL		NULL	NULL
1st attribute : root, nature present	NULL		ATTR. SUPPR.		VALUE CAN BE TRANSM.	VALUE CAN BE TRANSM
1st attribute : root, nature, val. spec. absent		ATTR. CREA-TED		NULL		
1st attribute : root, nature, val. spec. present		NULL		ATTR. SUPPR.		
2nd attribute : root, nature spec. absent					VALUE TRANSM.	VALUE TRANSM. AS TMV
2nd attribute : root, nature spec. present					NULL	VALUE CAN BE REPLA-CES

2.16. Choice of a value at random

This function has already been used in the choice of the arguments for the action simulation. It is used whenever it is necessary to completely specify an attribute that has been specified by root and nature only and there is a list of values associated with it. A 'random number' is generated by dividing the timer value obtained from the system by the contents of the list's counter cell and taking the remainder. This remainder is used as the ordinal number of a cell containing the value notion chosen. Up to 3 numbers will be generated in order to find a significant value before a systematic search is used.

2.17. Creation of a notion

Specific to simulation of the CREN action, this function calls for the allocation of two blocks for the first seven special attributes and returns the address of the first block as the address of the new notion.

2.18. Addition of an attribute

If an attribute with the same nature already exists, this function calls for the addition of the new attribute's value to the existing values (see below). Otherwise it requires a scan for an unused nature-value pair with automatic allocation of a new block to the attribute list, if necessary. The nature-value pair is completed and the reference counts of the notions specified, updated to reflect the new attribute's creation.

2.19. Addition of a value

This function is a special case of the previous one and follows the same plan. It requires a scan of the secondary list to locate an empty cell in which to store the new value. The reference count of the notion specified is updated. Location of an empty cell may imply the addition of a new block to the list. Allocation of a block is also required when a list does not exist and is to be created (a nature-value pair is transformed into a nature-pointer pair plus a list).

2.20. Suppression of a notion

A notion is never explicitly suppressed in the 3-graph. It is removed as a result of the suppression of the only remaining attribute linking it to the rest of the 3-graph. This function is thus a side-effect of the following one.

2.21. Suppression of an attribute

This function updates the reference counts of the notions that are nature and value of the attribute to reflect its suppression then reinitialises (to 'NIL') the nature-value pair of cells that contained it. If, instead of a nature-value pair there is a nature-pointer pair, only the value is suppressed (see below). As a result of the suppression one or more blocks may be returned to the free list :

- the block containing the attribute is now completely unused, the block can be freed if the empty block indicator is set.
- one or both notions updated are no longer attached to the 3-graph (their reference count = 0) and the space they occupy can be freed. The seven special notions - OCCDE, SUC1, SUC2, ARG1, ARG2, ARG3, ARG4 - are an exception, they are never physically removed even if their reference counts are zero.

The suppression of a notion may have a snowball effect if it is the base of a substructure that was attached to the rest of the 3-graph only by the attribute that has been suppressed. The space used by each of the notions is returned to the free list.

2.22. Suppression of a value

This is a particular case of the suppression of an attribute. The value notion's reference count is updated and the value cell contents reinitialised. If the whole block is then unused it may be returned to the free list under control of the empty block indicator.

2.23. The successors of an action

The last thing in the simulation of an action occurrence is the determination of the successors of the occurrence. The attributes whose natures are the notions SUC1 and SUC2 specify the possible successors. Except in the case of COMP only the SUC1 attribute is considered. In the case of COMP the choice of the SUC1 or SUC2 attribute is determined by the arguments of the occurrence as we have seen.

An action occurrence may have zero, one or any number of successors.

This is specified by the contents of the cell reserved for the SUC1 (or SUC2) attribute. If the action has no successors, we have reached the end of one process of transformation though there may be others still in progress. If there is one successor then it is the next occurrence to be simulated. The existence of a list of notions associated with the attribute does not cause a random choice as it did for the arguments of the occurrence. Instead, all the actions are taken together and theoretically a set of parallel processes is initiated. Since the simulator works in a sequential environment this parallelism must be simulated also.

2.24. Simulation of parallelism

In [1], the set of actions to be executed at time t_i is the set of all values of all attributes of the notion TODO. After execution of these actions, the values of the attributes of TODO are their successors and the actions to be executed at time t_{i+1}. We stated in section 1 that the notion TODO does not exist in the version of 3-graphs implemented, so the set of actions to be executed is not explicit in the 3-graph itself. This information is kept externally in the simulator, in the form of a pointer to the current action occurrence to be simulated and a stack of pointers to the successors of previous actions.

The successors of an action occurrence are transferred to the stack from the list associated with the SUC1 (or SUC2) attribute. In this way, the stack contains a reference to each parallel process to be initiated. Then one occurrence is transferred from the stack to the current action pointer, and the process is initiated.

The current process will be executed until it terminates or itself gives birth to another set of parallel processes In the first case, one of the remaining processes on the stack will be initiated. In the second case the new set will be added to the stack and one of these processes will be initiated.

Simulation will halt when all the processes have been removed from the stack and the last one terminated.

It is clear that this first approach to the simulation of parallel processes is unsatisfactory in several ways and priority must be given to improving this in future versions of the implementation.

2.25. Initiation of the simulation

The first action or actions to be executed are given by the original attributes of TODO, in [1]. In the implemented version, the simulator must be given this information by the user, since it cannot be deduced from the 3-graph itself. The simulator is designed for an interactive environment and it prompts the user to give it the first action. At present, only one action can be specified, that is, the user cannot initiate parallel processes.

The way in which the user can specify the first action will be discussed in the section on dialogue with the simulator.

3. THE TGL REPRESENTATION

In section 2 we described how the 3-graph is represented in the computer and how the dynamic aspect is simulated. We now turn to the representation used as input to the computer and how it is transformed into the internal representation.

We need a way of describing a 3-graph that is simple and also practical to use. The graphical representation and the triplet representation described in section 1 are basically simple but far from practical. It is extremely difficult in the graphical representation to represent anything but the simplest structures clearly. In a list of triplets it is difficult to discern the structure described, so that, although it would be quite possible to use a triplet notation for our input representation it would not be very practical.

As computer scientists we tend, by training and habit, to divide input to a computer into two categories :

- programs or sequences of actions to be executed by the computer;
- data on which these sequences act.

If we project this distinction onto descriptions of 3-graphs we distinguish sequences of occurrences of basic actions from more general structures. This is an arbitrary distinction that is not reflected in the 3-graph itself, but it seems at the moment, to be useful in practice.

It leads us to express occurrences of basic actions as instructions of a simple language. Each of the basic actions has a corresponding "operation code" with arguments that define a specific occurrence of the action. More general structures are described using "pseudo-operation codes" that specify one or more attributes by giving their root, nature and value notions in a form of triplet notation.

Now we want the format of our language to be simple and easy to use. A format resembling an Assembler-type language is both simple and easy to handle.

Since we have to program the transformation of a 3-graph described in the language into internal representation such a format allows us to take advantage of the Siemens Assembler language. The description of a 3-graph is written in a language - TGL - that conforms to the format of the Assembler. A 3-graph description in this language, presented as input to the computer, we will call a TGL text.

The process of transforming a description into internal representation falls into two parts :
- analysis of the text;
- generation of 3-graph structures in internal representation based on this analysis.

We have used the system's macro-assembler and macro library facilities to eliminate the programming of the text analysis phase and concentrated on the generation part of the transformation process.

The existence of the macro-assembler allows a programmer working in assembly language to extend the set of "operation codes" or "pseudo-operation codes'available to him. He can create, in this way, an assembly language tailor-made for the application he is programming, and understandable by the machine. The system also provides a utility program for creating and updating libraries of macros. When a program containing the macro codes is assembled, the programmer can direct the Assembler to use a macro library containing their definitions.

In our case, each of the TGL operation and pseudo-operation codes is defined as a macro. The Assembler performs the analysis of the input description, identifying the operation or pseudo-operation code and isolating the different notions.

These last correspond to the arguments of the macros and the Assembler establishes the correspondence between the arguments in the definition and in the text. It also deals with the recognition of notions when they first appear and when they are used subsequently.

For someone who is unaware of this link between TGL and Assembler, a TGL text is written in lines divided into 3 columns. The central column contains the operation code or pseudo-operation code. Each line defines one or more attributes of a root notion which may be defined at the same time or may previously have been defined.

So in TGL we must be able to denote a notion and in particular the eight basic actions and eight special notions.

3.1. The name of a notion

We denote a notion by giving it a name, a string of up to six alphanumeric characters beginning with an alphabetic character.

The strings of characters we have used to denote the basic actions - COMP, CREN, CRER, SUP, SUPR, LAT, TMV, REP and the special notions - OCCDE, ARG1, ARG2, ARG3, ARG4, SUC1, SUC2, NIL are those used in TGL. The names of the basic actions become the "operation codes" when they appear in the central column of a TGL line. The notion defined by the line is an occurrence of the basic action and the list of notion names in the third column specifies the attributes associated with it. For example,

JOE REP BILL, APPLE, PEAR, ORANGE, SOAP

defines an occurrence of the basic action REP, the notion named BILL is the value of the attribute ARG1, APPLE, PEAR and ORANGE are the names of the notions that are the values of the ARG2, ARG3, ARG4 attributes respectively and SOAP is the name of the notion that is the successor to this action occurrence - value of the SUC1 attribute. The string JOE is the name of

the notion (occurrence of the action). If there is nothing in the left-hand column then the notion has no name.

3.2. Name blocks and the dictionary

In connection with the notion names, we return briefly to the internal representation and the simulator routines.

We have seen that every notion existing in the 3-graph is allocated at least two blocks for the special attributes. A notion with a name has a third block physically following these two but not chained to them. This block contains the name and the address, in the 3-graph space, of the notion.

Further, the statistics cell in the first block contains a flag indicating the presence of a name block.

The blocks remain in the 3-graph but their contents are also copied, before the simulator routines take over, to a dictionary of notion names. With this dictionary it is possible during the simulation, to dialogue with the 3-graph about attributes and notions using the notion names given in the TGL text. The use of the name blocks is necessary since there is no other way of preserving the notion names after the Assembler has completed its work.

3.3. Sections of a 3-graph.

TGL is a simple language, and though more concise than the triplet notation of section 1, it shares the disadvantage of this simplicity : descriptions of complex structures are long and tedious to do. We would like to be able to build descriptions of parts of a 3-graph that are fairly independent, preserve these original descriptions separately and then join them together to form a complete 3-graph for simulation of the transformation processes they contain. Different 3-graphs can be formed by joining different parts together, and extra parts can be added on.

This brings us to the concept of a section of a 3-graph. A section is a description in TGL of part of a 3-graph that is transformed as a single unit into internal representation. It will usually be joined to other sections to form a complete 3-graph before simulation of the transformation processes contained in one or more of the component sections.

Two pseudo-operation codes are used to delimit a 3-graph section - BEGIN and TGEND, BEGIN initialises the process of transformation of a TGL text into internal representation. The right-hand column contains a character string and a number. The character string allows the user to give the section a mnemonic title but is otherwise ignored. The number is associated with the section and is used to identify it, particularly when a number of sections are joined together.

TGEND indicates the end of a TGL text.

3.4. Communication between sections

The different sections of a 3-graph are linked by attributes. A root notion's attribute list in one section can contain as nature or value of an attribute, notions that are defined in another section. In this way, for example, a section containing a transformation process can have attributes whose values are the root notions, in another section, which provide access to the structure on which the process will act. It is not possible at assembly-time, because of limitations imposed by the use of the Assembler, to define other types of links between 3-graph sections.

It would be possible to define other types of links using the 3-graph initialisation phase but we have not yet envisaged these possibilities.

In a TGL text we must be able to indicate in one section those notions that are defined in another and form the links between sections. The pseudo-operation EXT provides this capability. In the right-hand column the user identifies the

other section by its number and the notions concerned by their names. Notions in several different sections can be specified by the use of as many EXT lines and in this way several sections can be linked together.

In each section, the notion name and the number of the section are concatenated when the notion is defined and this composite name is the notion's name in the total 3-graph although it is referred to, within the section, by its name in the TGL text. For exemple, in 3-graph section number 3, the notion named SNOOPY will be defined as SNOOPY3, and every reference to SNOOPY in the section is a reference to SNOOPY3. (A reference, in this context, is the appearance of the notion as nature or value of an attribute). In section 5 references to the same notion must be given as SNOOPY3.

3.5. The INIT pseudo-opeation

In connection with the concept of 3-graph sections we have another pseudo-operation used in TGL. When we discussed the special notions - OCCDE, ARG1, ARG2, ARG3, ARG4, SUC1, SUC2 in the section on the internal representation we stated that these notions are always the first ones to be found in the 3-graph and they are always "known". This means that, when several sections of a 3-graph are joined together, one of them must contain these notions in the correct positions.

To be certain that this is always the case we impose two conditions on any set of sections linked together. At least one of these sections must bear the number 1 and in the TGL text of this section the 1st line following the BEGIN pseudo-operation must be an INIT pseudo-operation. This ensures three things :

- INIT causes the creation of the seven notions at the beginning of the section numbered 1.
- This section is always the first in the complete 3-graph.
- In sections numbered other than 1, the notions are automatically identified as EXTernal when the transformation process is initialised by BEGIN.

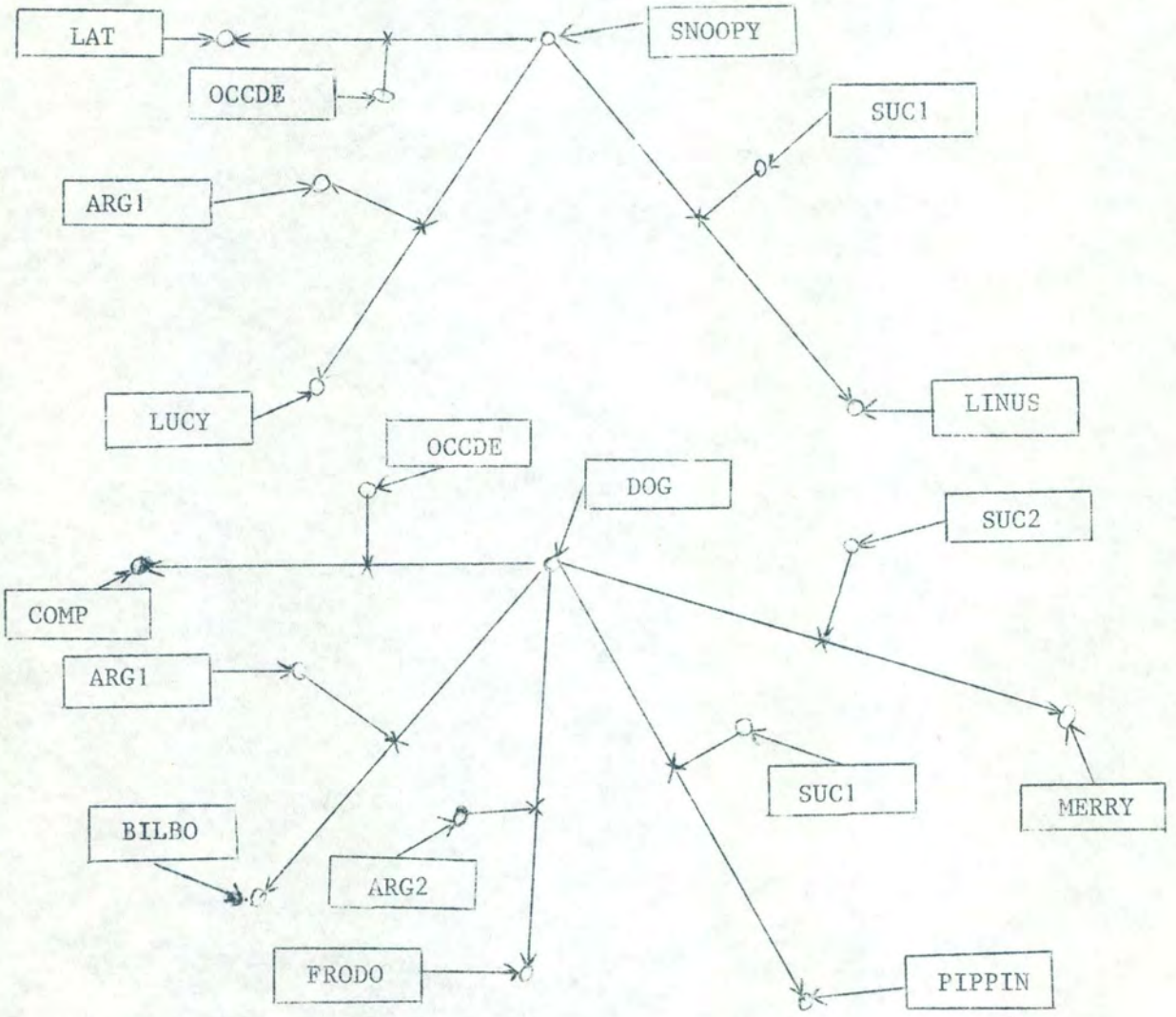
3.6. Specification of basic action occurrences

A basic action occurrence is defined by the operation code used. The notion is created that will be the occurrence and its arguments are specified by the list of names in the third column separated by commas. They are specified in the order ARG1, ARG2, ARG3, ARG4, SUC1, SUC2 though only those needed for the action are given.

For example, the lines

```
SNOOPY      LAT      LUCY, LINUS
DOG         COMP     BILBO, FRODO, PIPPIN, MERRY
```

represent the following structures :



If the successor of an occurrence is not explicitly named, it is the notion defined on the following line in a TGL text.

For example, if the successor of SNOOPY were DOG instead of LINUS, the above two lines could be written :

```
      SNOOPY        LAT         LUCY
                    COMP        BILBO, FRODO, PIPPIN, MERRY
OR    SNOOPY        LAT         LUCY, DOG
      DOG           COMP        BILBO, FRODO, PIPPIN, MERRY
```

When more than one attribute with the same root and nature notions are to be defined for a basic action occurrence, for example :

< SNOOPY, ARG1, LUCY >
< SNOOPY, ARG1, LIKES >
< SNOOPY, ARG1, LUDWIG >

they can be specified by writing the names of the value notions as a list separated by commas and enclosed in parentheses, thus

```
SNOOPY        LAT        (LUCY, LIKES, LUDWIG), DOG
```

When the value of an attribute will be determined during simulation or does not exist in the case of the last action in a process, an asterisk (*) is used to indicate the absence of a value notion. The attribute is created with NIL as value notion, as it would be if the name NIL had been used but the * notation indicates that the attribute will change during simulation whereas NIL defines the attribute's value notion once and for all.

3.7. Specification of other attributes

In TGL we have two "pseudo-operation codes" for the specification of attributes in a general way. These are denoted by DEF and DEFS and each line defines one or more attributes whose nature notion is the same. The nature of the attribute is given by the first notion name in the right-hand column of the line, and the value notion or notions by the other names in this column.

If there is a name in the left-hand column it specifies the root of the attribute, otherwise this attribute belongs to the last root notion given previously. For example, the lines :

```
PATTY        DEF        SNOOPY , SALLY
             DEF        BROWN, DOG, LINUS
             DEF        LUCY, MIKE
PETE         DEF        SNOOPY, LINUS
```

define the following attributes :

< PATTY, SNOOPY, SALLY>
< PATTY, BROWN, DOG>
< PATTY, BROWN, LINUS>
< PATTY, LUCY, MIKE>
< PETE, SNOOPY, LINUS>

The special attributes of a notion can also be specified using DEF or DEFS. The asterisk is used to indicate a nature or value that may be defined during the process of transformation of the 3-graph but when used with DEF it may have a specific significance. This stems from a restriction imposed on the implementation that any notion referred to by name must have its name occur at least once in the left-hand column. Usually, when a name appears in this column it means the notion is the root of the attributes being defined. But certain notions in a 3-graph are not roots of any attribute, so they must be declared without attributes. This is done by specifying an asterisk in the nature and value position in the right-hand column of the defining TGL line.

The necessity for two pseudo-operations DEF and DEFS also stems from a restriction on the implementation. DEFS allows the definition of attributes of a notion anywhere in a TGL text after the line originally defining the root notion. For example, the root notion EX2 given earlier with the attributes :

< SNOOPY, OCCDE, LAT>
< SNOOPY, ARG1, LUCY>
< SNOOPY, SUC1, LINUS>

If we wish to add the attributes :

< SNOOPY, BILBO, PIPPIN>

< SNOOPY, LINUS, MERRY>

we could write, anywhere after the original line,

```
SNOOPY      DEFS      BILBO, PIPPIN
            DEFS      LINUS, MERRY
```

An attribute specified by an asterisk in a previous line can also be redefined using DEFS but it cannot be used to add attributes with the same root and nature notions, that is :

```
LUCY        COMP      ☆, NIL
            ....
LUCY        DEFS      ARG1, LINUS
```

is permitted :

```
LUCY        CREN      LINUS, SNOOPY
            ...
LUCY        DEFS      ARG1, BILBO
```

is not.

3.8. Restrictions in TGL imposed by the use of the Macro-assembler

Before going on to discuss the transformation of a TGL text into internal representation we collect together the restrictions imposed by the use of the Macro-assembler in this transformation. Most of these have been previously mentioned.

1) A notion cannot be referred to by name, unless it is explicitly defined which means that its name must appear in the left-hand column of a TGL line, hence the

```
KNOT        DEF       ☆,☆
```

notation to define notions that are not root of any attribute.

2) It is impossible to know whether a notion has been previously defined so that, when defining a general set of attributes the pseudo-operation code must indicate whether the root notion is to be defined or whether it has already been defined , hence DEF and DEFS;

3) We cannot have access to the information previously generated so it is impossible to add attributes where the root and nature notions are the same as those of previously defined attributes.

4) We cannot specify any information other than the address of a notion defined in another section hence the limited way in which sections can be linked. Also the attribute lists of root notions in other sections cannot be extended in the current one.
5) It is impossible to generate a dynamic number of assembly-time variables so we cannot associate external notion names with their section numbers to create their complete names automatically. Hence, the complete names must be used.

3.9. Transformation of a TGL text into internal representation

Now that we have described TGL, we turn to the process of transformation of a TGL text into internal representation.

The raw input to this process, the TGL text, is a string of characters. The macro-assembler analyses this string and produces the information required to generate the attribute lists and secondary lists that form the internal representation of the 3-graph based on the TGL description.

These lists will be generated within the page framework imposed, as described earlier, in order to take advantage of the computer system's virtual memory environment. In addition to the chains of blocks forming the internal representation, the blocks containing the name of each named notion will be generated. The free list headers and chains of free blocks are also to be generated.

We have already stated that each of the TGL operation and pseudo-operation codes is defined as a macro to the Macro-assembler. A macro definition can contain calls to other macros whose definitions can, in turn, contain macro calls. In this way the definitions of the TGL operations and pseudo-operations contain calls to other subordinate macros. These usually correspond to functions needed for the generation of attribute lists and secondary lists. They may contain calls

to more elementary functions involved in the generation process. The depth of nesting varies with the complexity of the original and intermediate functions. At the bottom level we find the simplest functions such as the entry into a cell of an address of a notion, secondary list or succeeding block, or the setting of a flag in a cell.

The macros, including the TGL operation and pseudo-operation code definitions, will be referred to collectively as the TGL Reader. (They form a macro library that was created and can be updated using the MLU utility program). In the following discussion we will divide the set of macros into 3 groups :

- the TGL operation code definitions and their subordinate macros
- the DEF and DEFS definitions and their subordinate macros
- the BEGIN, TGEND, EXT and INIT definitions, of which only INIT has any subordinates.

Certain subordinate macros belong to all three groups, while others belonging to different groups have almost identical functions and only the conditions in which they are applied force the existence of more than one macro for a function.

However, the groups are fairly independent and can be discussed separately.

3.10. Generation of a basic action occurrence

The macro definitions of the TGL operation codes are almost identical. The only differences are in the generation of the contents of the OCCDE attribute cell, since the cell contents indicate the routine in the simulator that simulates the corresponding basic action, and in the number of arguments and, therefore, the number of calls to the appropriate functions. Those attributes that are not significant as arguments of a basic action occurrence are initialised to NIL. Table 3 lists the main functions required to generate a notion as an occurrence of a basic action, they are further discussed below.

TABLE 3

Creation of the root notion (TNAME)	Allocation of two blocks for the special attributes. Testing of the successor flags, see below(1). If the notion is named creation of the block containing its name and address, definition of the name as a symbol for the assembler (with concatenation of the section no).
Completion of the OCCDE attribute	Indicate the correct routine in the simulator. Set the flag for a basic action occurrence
Completion of the SUC1, SUC2 attributes (TESDF)	Setting of a special flag if the successor is not explicit. This flag(1) is tested when the successor is created and if set, the attribute will then be completed. If one successor explicit, completion of the attribute cell. If a list of successors, creation of the secondary list and its pointer. If ☆ specified, cell completed with NIL.
Chaining of the 2nd block to the 1st.	
Completion of the ARG1, ARG2, ARG3, ARG4 attributes (TESDF)	If one notion is given, completion of the attribute cell. If a list of notions, creation of the secondary list and its pointer. If ☆ or NIL specified, cell completed with NIL. (A null string causes generation of an error message).

(1) The successor flag is an assembly-time indicator, not a flag that appears in the 3-graph.

Allocation of a block or blocks : A record is kept of the number of bytes allocated in the current page. When a block is allocated this count is updated and if it exceeds the page size the current page is completed and a new page started. In addition, if required, the new block will be chained to its predecessor or to a value cell. (ALLOC and TROOM macros).

Definition of a notion's name and creation of the name block : When the 1st two blocks are allocated to the notion the name is declared attached to the beginning so that name and notion address are associated. These can then be inserted in the name block and the flag that indicates the notion is named is set (TNAME, DEFN macros).

Setting and testing of the successor flag (assembly-time indicators): a successor is implicit when the corresponding argument is a null string. There are two flags, one to indicate that SUC1 is implicit, and the other to indicate SUC2 is implicit. The address of the SUC1 or SUC2 cell is saved when the flag is set so that it can be located later(w the flag is tested)(SFLAG and TFLAG macros).

Setting of a flag in a cell : This function sets the flags that indicate that a cell contains a basic action occurrence, or a pointer to a secondary value list or a notion. (FLAG macro). Not included are the notion name flag (set by TNAME) and the empty block flags (set only by the simulator).

Creation of a secondary list : This function involves the iteration over the list of notion names to complete a cell in the secondary list for each one. Before this, a number of blocks sufficient to cover the list is allocated and chained to the value cell of the attribute (ALLOC macro) and the counter cell value is set (SETCNT macro).

Completion of a value cell of an attribute : The TSTAR macro deals with an * or NIL argument for a value of an attribute. The TSVAL macro detects a reference to one of the special

notions (OCCDE, ARG1, ARG2, ARG3, ARG4, SUC1, SUC2) and completes the value cell. The TESDF macro completes the value cell in all other cases. TSVAL is not strictly necessary. It would be sufficient to distinguish between notions created in the current 3-graph section and those external to it, including the special notions as a particular case (except in section 1). This distinction must be made, in any case, since before reference to a notion created in the current section can be made, the section number must be added to its name and must obviously not be added to a notion that is external. However, the detection of the seven notions is kept as a separate function because it possesses two advantages :

- section 1 is not treated as a special case as it would have to be if one wished to avoid concatenating the section no with these notion names;
- to remove it would be to presuppose that the handling of the special notions will remain the same and in the event of a change, make it much more difficult to modify this.

3.11. Generation of attributes from DEF and DEFS lines

We have already discussed the basic differences in the use of DEF and DEFS to define attributes. Because of these, there are one or two differences in the macro definitions. We will note these differences first before discussing the main functions that are common to both.

1) A name in the left-hand column of a DEF line means that a new notion is to be created with this name. The name in the the left-hand column of a DEFS macro is the name of a notion that must previously have been created in the same 3-graph section. Thus the DEF macro definition contains a creation function while the DEFS macro definition contains a function to locate the notion concerned.
2) A special case of a notion created by the DEF function is the notion with no attributes. The right-hand column contains asterisks for the nature and value of the attribute. This special case does not, of course, apply in the DEFS macro definition.

Although a DEF or DEFS line defines one or more attributes, there will usually be a group of such lines defining a set of attributes, all with the same root notion that is specified in the left-hand column of the first line. The DEF and DEFS function definitions were designed with this situation in mind and the first line of a group is treated as a special case.

The location counter of the assembler is used as a pointer to the current position in the 3-graph space. In the general case, it points to the next available nature-value pair in the root notion's attribute list. The first line of a DEF group causes the creation of a new notion, that is, the allocation of the 1st two attribute blocks and the name block, after which the location counter points to the beginning of the 1st block. The first line of a DEFS group also forces the location counter to the beginning of the notion's 1st block. For the simple case where a nature-value pair is added to an attribute list the DEF or DEFS generation is straightforward. It is the special cases that make the DEF and DEFS definitions complex since a lot of indicators are required to keep track of the current situation.

The simple case follows the basic pattern :

1) The location counter is set at the next free nature-value pair.
2) The notion that is nature of the attribute is not one of the special notions. Its address is inserted into the nature cell.
3) The notion that is value of the attribute is not a special notion. Its address is inserted into the value cell.
4) No special flags are needed and the location counter is set to the cell following the newly created attribute.

This basic pattern is disrupted when one or more of the following occurs :

- the first line of a DEF or DEFS group : involves creation of a new notion or location of an existing notion. The location counter is not set to a nature-value pair but to the beginning of the notion's attribute list.

- the location counter is set to the pointer cell of the current block and not to a nature-value pair; involves allocating a new block, chaining it to the list and setting the location counter before creating the attribute;
- the nature of the attribute is one of the special notions; involves saving the current position of the location counter, setting it to the attribute position and resetting it after the attribute has been created. If the attribute is OCCDE this must be noted during creation of the attribute.
- there is a list of values of the attribute : involves saving the current location counter position so it can be reset afterwards, allocating a sufficiant number of blocks for the secondary list, setting the counter cell and pointer to the list and filling in the values of the attribute.
- the value of the attribute is a special notion or a basic action : the cell must be completed with the appropriate contents including flags (for an occurrence of a basic action).

Table IV shows the most important macros that are subordinate to DEF or DEFS, and their functions.

TABLE IV

ALLOC	allocates a block, fills in the pointer to the new block.
DEFNOT	creates a new notion, requires allocation of two blocks for special attributes and a third for name block which it completes.
LISTVAL	completes a cell in a secondary list of the OCCDE attribute, calls TSOCC to detect basic action occurrences.
VALST	completes a cell in any other secondary li..., calls TSVAL to detect special notions.
VALUE	completes the value cell of a nature-value pair, calls TSVAL to detect special notions
NATRE	completes nature cell of a nature-value pair unless call to TSNAT detects a special notion and resets location counter.
SETCNT	sets counter cell in a secondary list.

3.12. BEGIN, TGEND, EXT and INIT

The generated output of the Assembler with the TGL reader is an object module containing the internal representation of a 3-graph section. The BEGIN and TGEND macros generate the initial and terminal coding that the Assembler requires to produce an object module, and BEGIN initialises certain global information required during generation.

The EXT macro causes the Assembler to generate the information needed when the object module is linked with others, thus forming a complete 3-graph from the different sections.

We have already discussed the function of INIT.

□□□□□□□□□□□□□□□□□

The result of the transformation of the TGL text is a 3-graph section in internal representation that is still incomplete in two ways :

- it contains references to notions in other sections which will only be completed when all the sections are linked together and the simulator routines loaded.
- the statistics cells in the notions have not been initialised to their correct values. This will be done as part of an initialisation phase before simulation takes place. At the same time the dictionary of notion names will be created. This initialisation phase will be discussed in the next section.

o
o o

4. THE SIMULATOR -LANGUAGE, INITIALISATION AND INPUT-OUTPUT

Now that we have described TGL and the TGL reader which produces a 3-graph in internal representation, we return to the simulation part of the implementation where several aspects remain to be discussed :

- the programming language used to program the routines;
- the organisation of the 3-graph sections, the simulator routines, initialisation and input-output routines into a pageable program in virtual memory;
- the initialisation of the 3-graph, mentioned in the previous section, before the transformation processes can be simulated;
- the possibilities provided by the input-output routines for the user to query the 3-graph.

4.1. Choice of programming language

The choice of programming language was influenced by the following requirements or aims :

- since the 3-graph sections and simulator routines (including input-output and initialisation routines) form one program from the system's point-of-view we wished to minimise the space taken by the latter in order to maximise the possibilities of expansion of the former (every page or part of a page taken up by the routines is "dead" space for the expansion of the 3-graph).
- we need to be able to manipulate addresses directly and indirectly and indirectly and to check bits in a word to test the flags in a cell, that is, we need to work at a very basic level;
- facility of maintenance and modification of the routines means that we wish to be able to isolate the different functions i.e to create a highly modular structure that is also easy to follow and as obvious as possible without requiring a lot of explanatory documentation.

The first two objectives may seem to be contradictory to the third in that the power required for the first two implies use of a low-level language in which it is notoriously difficult to achieve the third.

It is easier to control the size and internal organisation of the machine code of a program written in an assembly language than in a high-level language. An assembly language also gives direct access to memory addresses and individual bits of a machine word without recourse to the sort of "clever" programming so often criticized as a source of errors and obscurities in programs. On the other hand, an assembly language program too often, presents itself to a reader as a long string of instructions where any structure is extremely difficult to discern and where the functions of the different parts are not clear even when they are not so inextricably linked as to disappear altogether in a homogeneous mass. However, we believe that by using to the full the possibilities of a macro-assembler and a modular style of programming, a program written in assembly language need not be a long ribbon of instructions but can reflect the structure of the problem.

The analysis of the problem leads to the definition of the different functions needed to solve it.

These functions usually form a hierarchy with several levels. The macro-assembler provides us with a tool for creating instructions corresponding to the functions at any level from the simplest to the most complex. The coding finally produced, whether it be a sequence of basic instructions or a call to a routine that performs the function is not important, in either case the structure is reflected by the use of the macro-instruction.

Macro-instructions can be used not only to invoke functions but also to define complex data structures. The TGL reader is a good example of this use of macros.

It produces no executable code, the internal representation of the 3-graph is entirely composed of constant and storage definition operations.

Nevertheless, at each level the programmer can still retain a close contact with the machine since he decides what to include in the macro definition. The possibilities offered by the macro-assembler, of testing assembly-time variables and varying the coding generated can be a great advantage. Specific, and often shorter, coding can be produced for a particular case in the function. A routine to perform the same function would have to consider all cases and thus do the testing at execution time. Also the possibilities for varying the coding are greater at assembly time since the arguments of a macro can be other things than just the values that may be passed to a routine at execution time.

Macros can also be used to document a program or a set of programs. For example, the documentation of common variables that are passed between several independent routines. This documentation, if done once in a macro, can be made to appear easily in each routine and when the use or meaning of the variable changes, the documentation is changed in the macro and the modifications appear in the routines when they are re-assembled.

Since, in this way, we can take advantage of the power of an assembly language and still have a program that is structured, making maintenance and adaptability simpler, the choice of the macro-assembler as programming language is obvious.

This is not to imply that there are no more problems. Indeed, the limitations of the macro-assembler available to us, has caused problems though much less so in the simulator rou-

tines than in the TGL reader. Also, any program in whatever language it is written, but especially an assembly language program must be carefully documented if all the programming effort is not to be a sheer waste of time.

4.2. Programming style and documentation of the simulator

The main feature of the programming is the combined use of macros and subroutines to perform the different functions required in the simulation of the 3-graph basic actions. A subroutine in this context is a part of a program that performs one function in the simulation. While not necessarily a separately assembled module, it usually has only one entry point and one exit. It cannot be reached sequentially in the program and the information it needs to perform its function is given to it explicitly when it is invoked. This isolation is intended to limit the impact of changes in the program.

For certain functions we have a 'twinning' of a macro and a subroutine. The macro call provides the in-line indication of a function to be performed. The macro expansion calls the subroutine that carries out the function. It may also perform some "housekeeping" for the subroutine.

Each macro or subroutine contains in its coding the totality of its documentation. This avoids the necessity of separate documents that may get lost or quickly become obsolete because changes in the program are not reflected therein. A short description of the function of each routine and the way in which this is carried out heads the function, together with a specification of its input and output parameters, the routines it calls and the routines by which it is called. In addition, the coding contains abundant comments.

The macros are similarly documented and in the macro library there is also a macro (MXREF) that contains a cross-reference table of macro and routine calls. Since the documentation and source code are kept together, changes in the macros or routines and the updating of the documentation can be

done at the same time without the need to keep several documents in parallel. Consequently, less effort is needed and the programmer is less likely to forget to update the documentation.

4.3. Loading and initialisation of a 3-graph

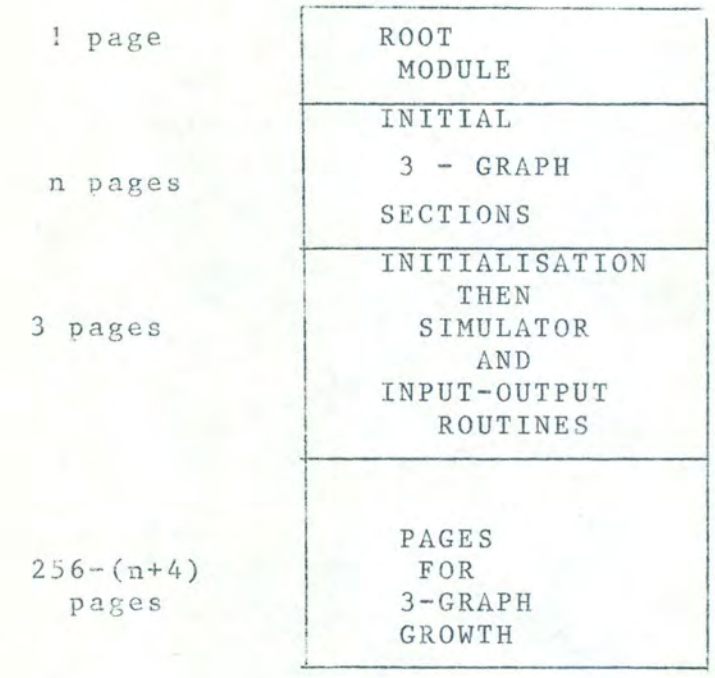
The output of the transformation of a TGL text by the TGL reader is an object module : the format for input to the BS-2000 linkage editor program. Such an object module is stored in a library which will be the principal source for the linkage editor when it resolves unsatisfied references in a specified module. A 3-graph section in its object module form will contain unsatisfied references when the EXT pseudo-operation has been used to specify notions in other sections. Unless it is section #1, it will contain, in any case, unsatisfied references to the notions OCCDE, ARG1, ARG2 ARG3, ARG4, SUC1, SUC2.

When a 3-graph is to be simulated its different sections must be linked together and the simulator routines added to make a single program. The linkage editor performs this task, producing a loadable module. It must be given the names of the object modules it is to link together. For a 3-graph simulation these are :

- TRGR1, the name of 3-graph section 1;
- the name(s) of the other main 3-graph section(s) required (the name is generated by the TGL reader as TRGRn where n is the section number);
- the module containing the simulator routines;
- the module containing the initialisation routines and root module (see below);
- the module containing the input-output routines.

All the 3-graph sections required do not need to be specified, since they will be included by the linkage editor resolving references. Section 1 must be given explicitly because the special notions must be the first notions in the program's address space.

To save space, the final program has an overlay structure. Since the initialisation routines are no longer needed after the 3-graph has been initialised, they are overlayed by the simulator routines. This technique requires the existence of a root module whose sole purpose is to call the initialisation routines and then the simulator routines. Since this root module is not overlayed it also contains the definition of the dictionary of notion names used by the input-output routines and created by the initialisation routines. The layout of the 3-graph and routines in the virtual memory space is shown in the diagram :



When the program resulting from the linkage edit is loaded for execution the root module receives control and immediately invokes the initialisation routines. These routines perform three tasks :

- Initialisation of statistics cells or reference counts

Each notion's attribute list and associated secondary lists are scanned. Each notion that appears in the lists as a nature or value notion is accessed and its reference count incremented by one.

Here we have taken into account the paging environment of the program and attempted to avoid excessive paging. When the notions in the lists are accessed for updating of their reference counts they may be widely dispersed and in a heavily loaded system this could lead to frequent paging requests. The technique used to avoid this consists in discriminating between notions in the same page and notions in other pages. A notion in the same page is accessed and its reference count updated. For a notion in another page, the reference is noted in a table.

At the end of a page, the notions noted in the table are accessed and their reference counts updated. This is done by dividing the pages of the 3-graph into groups of contiguous pages. All the notions in a group of pages are accessed, then all the notions in the next group and so on until all the notions in the table have been accessed. In this way, we hope to keep the paging rate down.

- Creation of the dictionary of notion names

The dictionary is created only on the request of the user, who is asked at the beginning of the initialisation whether a dictionary is to be created or not. The dictionary is usually created the first time a particular 3-graph is simulated and need not be re-created each time.

To create the dictionary, the 3-graph is scanned to find those notions that have names (flagged in the statistics cell) and their names and addresses are transferred from the name blocks.

The dictionary is created as an indexed sequential file. The name of the notion is the retrieval key for a record in the file. Thus, the dictionary gives a notion's address when its name is known. The name blocks in the 3-graph give a notion's name when its address is known.

The scan of the 3-graph to find the name blocks is a much simpler scan than the one to initialise the statistics cells, so we preferred to use two scans. With separate scans, the initialisation routines are also more modular.

- Initialisation of the page table

The page table is contained in the root module (page 0) and contains an entry for each of the 256 pages in the program's virtual memory space. For each page containing part of the 3-graph the entry indicates :

. the section to which the contents of the page belongs;
. an indication of whether the free list of the page is empty or not.

It is used when, during simulation, space is required for allocation to an attribute or secondary list and the page in which the list is located is complete. The page table is scanned to find another page with space available. Preference is given to pages with the same section number to keep the notions of each section together. If all currently existing pages are full their the system is asked to allocate another page.

After initialisation is finished, the root module loads and hands control to the simulator routines. The first thing that happens is that the simulator asks for the name of the first action occurrence to be simulated. It consults the dictionary to find the corresponding notion address and then starts the simulation process.

- The input-output routines

The simulator routines do not provide any method of conserving or retrieving the resultant 3-graph after a simulation. It would be possible to provide a sort of dump, in readable form, of the 3-graph at the end of a simulation. We decided, instead, to provide a set of routines that permit a limited form of dialogue with the 3-graph during the simulation.

The reasons for this are twofold :

- the sheer volume of information that would be produced by a dump of the 3-graph. Most of it would be of little interest but a dump is not discriminating;
- the interesting part of the simulation is not necessarily the terminal results. It is also, the process which provides the results, and the intermediate stages.

The input-output routines are invoked when the user interrupts the simulation process from his terminal or when a particular point in the process is reached, due to an earlier call to these routines. When invoked they prompt the user to send a command to be executed. The command can be one of the following :

- display of an attribute or all the attributes with a given root notion. This command is

  D < notion name >

  or D < notion name 1> OF <notion name 2>

In the first case, all the attributes in the notion's list of attributes are listed. In the second case the attribute whose root notion is specified by notion name 2, and nature notion by notion name 1, is listed. The command :

D ?

provokes the listing of the attribute list of the notion that is the next action occurrence to be simulated.

- setting a halt point

  The command is AT <notion name>

  The notion specified must be an action occurrence. Its OCCDE attribute is saved and replaced by a halt point indicator. When the simulation process reaches the notion, the input-output routines are invoked (and produce the message 'HALT POINT'). Only one halt point at a time is currently permitted.

- return to the simulation process. This can be done in two ways. The command

R

indicates simulation is to resume where it left off. The command

BR <notion name>

indicates simulation is to resume at the notion specified.

Except when the command R or BR is given, the routines prompt the user for further commands after execution of the previous one. When a display command is given the output is in three column format. The left-hand column gives the name of the root notion, the central column the name of the nature notion and the third column the name(s) of the value notion(s) of the attributes. If a notion has no name, a unique name is generated by the routines, which can be subsequently used to identify the notion.

5. FUTURE VERSIONS OF THE IMPLEMENTATION

We have tried to show the main points of interest in the current version of the 3-graph implementation on the Siemens 4004 computer. It is, however, only a first version and there are several unsatisfactory points about it, in particular certain choices made during the implementation have proved to be questionable. Although, to date, the implementation has not been used very much to treat 3-graphs so it is difficult to say how it will stand up in practice or what errors exist, we can point out one or two modifications that will be necessary, or useful in future versions. These fall into three categories :

- those necessary to bring the implementation into line with the current theory;
- those necessary to correct certain doubtful choices in the implementation
- those that will extend the current possibilities of the implementation.

5.1. Implementation and theory

The modifications are required to bring the implementation into line involve the addition of the notion TODO and the treatment of the basic actions COMP, CREN, CRER, SUP, SUPR, LAT, TMV, REP. These do not appear as notions in the 3-graph implementation in the same way as other notions. They usually appear as particular values of the OCCDE attributes and the way in which they are handled means that they can not be root or nature of any attribute. The other special notions can be root, nature or value of attributes (though all those in which they are root notions must be defined in 3-graph section # 1 (see restrictions in the section on TGL)) and, since in the theory

the basic actions can also play any role in an attribute they must be included in the functions that declare the special notions in TGL, INIT and BEGIN. Also in TGL, the functions that detect references to the special notions will be slightly altered to take into account the new way in which the basic actions are represented. In the simulator too, this will mean slight changes, for example where the special notions have to be protected from removal from the 3-graph, or where they are treated as special cases in the creation of attributes.

The most important modifications to the simulator are those required for the addition of the notion TODO. The TGL reader is not greatly affected, the notion must be declared and recognised in the same way as the others. In the simulator, however, a new function must be added. At the end of the simulation of each action the attributes of TODO must be updated so that they specify the actions to be executed next. The function that indicates, on exit from each basic action simulation the successors of the current action will be altered to include the new function. The choice of successor and simulation of parallelism functions will be slightly altered.

5.2. Doubtful choices in the current version of the implementation

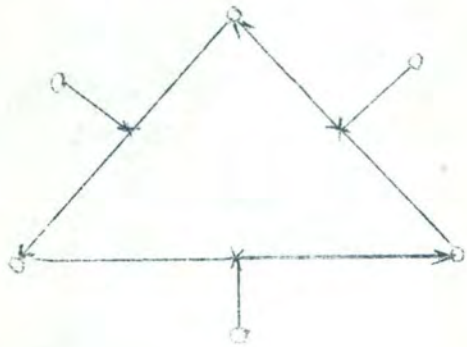
Independently of any modifications caused by the introduction of the notion TODO the simulation of parallelism is a prime candidate for change. The current method used is highly unsatisfactory for the following reason:

when an action occurrence initiates a set of parallel processes, that is, there is a secondary list of successors the contents of this list is copied onto a stack, and one of the processes is initiated. Only when this process terminates will another be initiated, and we end up with, in effect

a sequence of single processes. Furthermore, if one process never terminates, those left on the stack can never be initiated. Thus it is important to find another way of simulating parallelism in the 3-graph.

The routines that suppress inacessible substructures from the 3-graph can also be improved, from the user's point of view and in the space-saving aspect. These substructures are a waste of valuable space to the simulator but they are not necessarily uninteresting to the user. The introduction of an indicator that could be set to prevent their automatic suppression would be useful. It would allow the user to preserve them for later use or ask for them to be copied to a file before suppression.

Not all inaccessible structures are detected by the present methods. Structures such as the following :



where none of the notions is a special notion, are inacessible but the notions are all nature or value of an attribute and cannot be suppressed. To eliminate such cases, a rescan similar to the initialisation scan is required to mark all accessible notions and then remove those that are left unmarked (garbage collector).

5.3. Extensions to the possibilities

At the moment, these concern mainly the facilities for obtaining results from the 3-graph, conserving a resultant 3-graph from one simulation to another and interaction with the 3-graph during simulation.

The commands for input and output described in section 4 need to be extended. In particular, it would be interesting to have more than 1 halt point at a time and have a halt point that remained in the 3-graph until explicitly removed. It would also be interesting to be able to specify halt points in a TGL text. A special halt point at the end of a simulation process when final results are available is, in the present case, almost a necessity since the final state of the 3-graph is lost.

The possibilities of the display commands could be extended to permit the display of structures starting from a particular root notion and extending downward through its immediate attributes.

In the current version of the implementation it is impossible to halt the process of simulation and resume it at a later date or to conserve the state of a 3-graph from one simulation to another. This is an important obstacle to the use of the implementaton and its removal is high in the list of extensions to be made in future versions.

6. CONCLUSION

Of necessity, many details of the implementation have been left out of this description. We refer the interested reader to the program listings for the detailed description of every function, whether macro or routine.

In conclusion, we can only hope that the implementation in the current and future versions will be used for the study of semantics, and that it will prove to be as easy to maintain and modify as we have tried to make it.

BIBLIOGRAPHY

[1] A Proposal for the Formalisation of Semantics :
C. Cherton, M. Hewitt, F. Doyen

[2] Siemens Assembly System Reference Manual

[3] Siemens 4004/35, 45 and 55 Processors Reference Manual

[4] Siemens 4004 VMOS Utility Routines Reference Manual

[5] Siemens 4004 VMOS Executive Macros Reference Manual

°

° °

AN IMPLEMENTATION OF 3-GRAPHS

FIRST VERSION

APPENDIX : PROGRAM LISTINGS

M. HEWITT

```
GENERAL  01411500:01433600 12/20/74 12/20/74

         MACRO
         GENERAL
.*ORGANISATION OF A NOTION
.*       ************************************************************
.*EACH NOTION DEFINED CONSISTS OF AT LEAST 2 CONTIGUOUS 20-BYTE BLOCKS
.*WITH THE FOLLOWING LAYOUT:
.*          <---20-BYTE BLOCK--------->   4-BYTE 'CELL'
.*START OF |-----------------------------------------------|
.*NOTION   |  | | |  |  | | |  | | |  |  | | |  |  | | |  | |
.*         |--|---|-----|----|---|-----|---|-----|---|-----|-|
.*            | OCCDE |    | POINTER |  ARG2   |  ARG4   |
.*          STATISTICS |  SUC2 TO   ARG 1     ARG3       POINTER
.*                   SUC1     NEXT                       TO NEXT
.*                            (CONTIGUOUS)               ATTRIBUTE
.*                            BLOCK                      BLOCK IF THERE
.*                                                       IS ONE
.*
.*THIS MAY BE FOLLOWED BY ANY NO OF 20-BYTE ATTRIBUTE OR 'TAIL' BLOCKS
.*                 4-BYTE CELL   <ATTRIBUTE>
.* POINTED TO BY    |-|----------------------|
.* POINTER CELL OF>| |   |    | | | | | | |
.* PRECEDING BLOCK |-------------|---|----|-|
.*                  <ATTRIBUTE>  |   |   POINTER TO NEXT ATTRIBUTE
.*                           NATURE |   BLOCK IF THERE IS ONE
.*                             OF   VALUE
.*                         ATTRIBUTE OF
.*                                ATTRIBUTE
.*IF AN ATTRIBUTE HAS MORE THAN 1 VALUE THE VALUE CELL CONTAINS A POINTER
.*TO ONE OR MORE VALUE LIST BLOCKS
.*    ATTRIBUTE BLOCK
.*   -------------
.*    | NAT| |  |
.*   --------|-----COUNTER   1ST VALUE LIST
.*         VA|LUE   CELL          BLOCK
.*         CE|LL   ------------------------|
.*           ----->| |  | V A|L U |E S | |  |
.*                 |-|-------------------|--|
.*                 COUNT OF              PO|INTER CELL
.*          NO OF CELLS ALLOCATED          |
.*                 IN LIST                 |
.*               |-----------------------|
.*               |  2ND & SUCCELDING VALUE LIST BLOCKS
.*               |  |----------------------|
.*               |->| V A| L  |U E | S  | |  |
.*                  |---------------------|--|
.*                                        POINTER TO SUCCELDING BLOCK
.*THE FIRST 2 BLOCKS OF A NOTION WITH A NAME ARE FOLLOWED BY A
.*20-BYTE BLOCK CONTAINING THE NOTION'S NAME AND ADDRESS
.*       ************************************************************
.*ORGANISATION OF A CELL
.*       ************************************************************
```

GENERAL 01411500:01433600 12/20/74 12/20/74

```
.*              <-8-><---24--->(BITS)
.*              |---|---------|
.*              | | |     |   |
.*              |-|-|-----|---|
.*              FLAGS   ADDRESS OF NOTION(OR OBJECT OF THE UNIVERSE)
.*         ***********************************************************
.*SPECIAL CONTENTS OF A CELL
.*         ***********************************************************
.*A)THE OCCDE CELL CONTAINS AN OFFSET INTO A TABLE IN THE SIMULATOR
.*ORGANISED AS FOLLOWS:
.*START OF TABLE:            4 BYTE ENTRIES
.*                  ADDRESS COMP SIMULATION ROUTINE
.*                  ADDRESS CREN SIMULATION ROUTINE
.*                  BLANK ENTRY
.*                  ADDRESS LAT  SIMULATION ROUTINE
.*                  ADDRESS REP  SIMULATION ROUTINE
.*                  ADDRESS SUP  SIMULATION ROUTINE
.*                  BLANK ENTRY
.*                  ADDRESS SUPR SIMULATION ROUTINE
.*                  ADDRESS TMV  SIMULATION ROUTINE
.*                  ADDRESS CRER SIMULATION ROUTINE
.*B)A NATURE OF VALUE CELL IS UNOCCUPIED IF ITS CONTENTS = X'OFFFFFFF'
.*CORRESPONDS TO THE NOTION 'NIL'
.*C)THE END OF A CHAIN OF ATTRIBUTE OR VALUE LIST BLOCKS IS INDICATED
.*IN THE POINTER CELL WHOSE CONTENTS = X'OFFFFFFF'
.*         ***********************************************************
.*FLAG BITS
.*         ***********************************************************
.*   CELL                              BIT 7   6   5   4   3   2   1   0
.*OCCDE-BASIC ACTION                     1   0   -   -   -   -   -   -
.*      -NON-BASIC ACTION                0   0   -   -   -   -   -   -
.*      -POINTER TO LIST                 0   1   -   -   -   -   -   -
.*SUC1
.*SUC2  -NOTION                          0   0   -   -   -   -   -   -
.*ARG1  -POINTER TO LIST                 0   1   -   -   -   -   -   -
.*ARG2
.*ARG3
.*ARG4
.*NATURE OF ATTRIBUTE                   0   0   -   -   -   -   -   -
.*VALUE
.*OF    -NOTION                         0   0   -   -   -   -   -   -
.*ATTRIBUTE-POINTER TO LIST             0   1   -   -   -   -   -   -
.*VALUE-NATURE IS OCCDE
.*CELL          -BASIC ACTION           1   0   -   -   -   -   -   -
.*ID            -NON-BASIC              0   0   -   -   -   -   -   -
.*VALUE-NATURE ANY NOTION
.*LIST
.*              -NOTION                 0   0   -   -   -   -   -   -
.*POINTER TO ATTRIBUTE BLOCK            0   0   -   -   -   -   -   -
.*STATISTICS CELL-NAMED NOTION          1   -   -   -   -   -   -   -
```

GENERAL 01411500:01433600 12/20/74 12/20/74

```
.*                  -UNNAMED                    C   -   -   -   -   -   -   -
.*STATISTICS OR COUNTER CELL/EMPTY               0   0   0   1   -   -   -   -
.*BLOCK IN CHAIN OF BLOCKS
.*COUNTER IN LIST OF VALUES                     -   -   -   -   -   -   -   -
.*      ************************************************************
.*GLOBAL MACRO VARIABLES(CROSS-REFERENCE TABLE)
.*      ************************************************************
.*NAME     TYPE  TESTED IN       SET,RESET,UPDATED IN     USED IN
.*                (MACRO NAME)    (MACRO NAME)             (MACRO NAME)
.*&BASACT  BOOL  LISTVAL         DEF
.*                VALUE           TSOCC
.*&DEFCON  BOOL  DEF             DEF
.*                DEF             DEF
.*                NATRE           VALUE
.*&LAB     ARITH                 SFLAG                    SFLAG
.*                                                       TFLAG
.*&NAMGEN  ARITH                 DEF                      DEF
.*                                DEF                      DEF
.*                                                       VALUE
.*&NODY    ARITH                 ALLOC                    ALLOC
.*                                INIT                     TROON
.*                                TROON
.*&PASSNO  ARITH                 BEGIN                    TGEND
.*                                                       DEF
.*                                                       DEFS
.*                                                       LISTVAL
.*                                                       NATRE
.*                                                       SUBLST
.*                                                       TESDF
.*                                                       TNAME
.*                                                       VALST
.*                                                       VALUE
.*&POS     ARITH                 SUBLST                   DEF
.*                                TSNAT                    DEFS
.*                                                       SUBLST
.*                                                       TSNAT
.*&RMATB   ARITH NATRE           DEFS
.*                                DEF
.*                                NATRE
.*                                VALUE
.*&RMVAB   ARITH DEF             DEF
.*                DEFS            DEFS
.*                LISTVAL         LISTVAL
.*                SUBLST          SUBLST
.*                VALST           VALST
.*&SFMAT   BOOL  DEF             DEF
.*                DEFS            TSNAT
.*                NATRE
.*                VALUE
.*&SPOCCDEBOOL    ALLOC           DEF
```

GENERAL 01411500:01433600 12/20/74 12/20/74

```
.*                DEF                TSNAT
.*                DEFS
.*                VALUE
.*&SEVAL  BOOL    SUBLST             DEF
.*                TESDF              TSVAL
.*                VALST
.*                VALUE
.*&STAR   BOOL    TESDF              TSTAR
.*&SUCC1  BOOL    TFLAG              SFLAG
.*                                   TFLAG
.*&SUCC2  BOOL    TFLAG              SFLAG
.*                                   TFLAG
.*        ********************************************************
.*MACRO TABLE(CROSS-REFERENCE)
.*        ********************************************************
.*MACRO NAME |CALLED BY;                              |CALLS;
.*ALLOC      |DEF,DEFS,LISTVAL,NATRE,SUBLST|FLAG
.*          |VALST                         |
.*PLGIN      |3-GRAPH PSEUDO-OPERATION      | -
.*COMP       |3-GRAPH BASIC OPERATION       |FLAG,TNAME,TESDF
.*CREN       |3-GRAPH BASIC OPERATION       |FLAG,TNAME,TESDF
.*CRER       |3-GRAPH BASIC OPERATION       |FLAG,TNAME,TESDF
.*DEF        |3-GRAPH PSEUDO-OPERATION      |ALLOC,DEFNOT,LISTVAL,NATRE
.*          |                              |VALST,VALUE,SETCNT
.*DEFN       |TNAME                         |TROOM
.*DEFNOT     |DEF                           |FLAG,TROOM
.*DEFS       |3-GRAPH PSEUDO-OPERATION      |ALLOC,LISTVAL,NATRE,VALST,
.*          |                              |VALUE,SETCNT
.*EXT        |3-GRAPH PSEUDO-OPERATION      | -
.*FLAG       |ALLOC,COMP,CREN,CRER,DEFNOT,  | -
.*          |     LAT,LISTVAL,NATRE,OUTIT,|
.*          |REP,SUBLST,SUP,     SUPR,     |
.*          |TESDF,TFLAG,TMV,TSOCC,TSVAL,  |
.*          |VALST,VALUE                   |
.*INIT       |3-GRAPH PSEUDO-OPERATION      |OUTIT
.*LAT        |3-GRAPH BASIC OPERATION       |FLAG,TESDF,TNAME
.*LISTVAL    |DEF,DEFS                      |ALLOC,FLAG,TSOCC
.*NATRE      |DEF,DEFS                      |ALLOC,FLAG,TSNAT
.*OUTIT      |INIT                          |FLAG
.*REP        |3-GRAPH BASIC OPERATION       |FLAG,TESDF,TNAME
.*SETCNT     |DEF,DEFS,SUBLST               | -
.*SFLAG      |TESDF                         | -
.*SUBLST     |TESDF                         |ALLOC,FLAG,TSVAL,SETCNT
.*SUP        |3-GRAPH BASIC OPERATION       |FLAG,TESDF,TNAME
.*SUPR       |3-GRAPH BASIC OPERATION       |FLAG,TESDF,TNAME
.*TESDF      |COMP,CREN,CRER,     LAT,REP,  |FLAG,SFLAG,SUBLST,TSTAR,TSVAL
.*          |SUP,     ,SUPR,TMV            |
.*TFLAG      |TNAME                         |FLAG
.*TGEND      |3-GRAPH PSEUDO-OPERATION      | -
.*TMV        |3-GRAPH BASIC OPERATION       |FLAG,TESDF,TNAME
```

```
GENERAL  01411500:01433600 12/20/74 12/20/74

.*TNAME        |COMP,CREN,CREF,      LAT,REP,  |DEFN,TFLAG,TROOM
.*             |SUP,      SUPR,TMV             |
.*TROOM        |DEFN,DEFNOT,TNAME                | -
.*TSNAT        |NATRE                            | -
.*TSOCC        |LISTVAL,VALUE                    |FLAG
.*TSTAR        |TESDF                            | -
.*TSVAL        |SUBLST,TESDF,VALST,VALUE         |FLAG
.*VALST        |DEF,DEFS                         |ALLOC,FLAG,TSVAL
.*VALUE        |DEF,DEFS,                        |FLAG,TSOCC,TSVAL
.*      ***********************************************************
.*WRITING MACRO CALLS
.*      ***********************************************************
.*1)NOTION NAME-1 TO 6 ALPHANUMERIC CHARACTERS STARTING WITH AN
.*ALPHABETIC CHARACTER (EXCLUDING @)
.*THIS NAME IS CONCATENATED WITH THE SECTION NUMBER(SEE BEGIN MACRO)
.*2)IN BASIC ACTION MACROS A PARAMETER MAY BE REPLACED BY A LIST WHICH
.*IS ENCLOSED IN PARENTHESES WITH THE NOTION NAMES SEPARATED BY COMMAS.
.*THE WHOLE LIST INCLUDING PARENTHESES MAY NOT EXCEED 127 CHARACTERS.
.*3)A PARAMETER MAY BE REPLACED BY AN * WHICH INDICATES THAT THE VALUE(S)
.*OF THE ATTRIBUTE WILL BE SPECIFIED EITHER BY A DEFS MACRO (LIST OF
.*VALUES >127 CHARACTERS) OR DURING EVOLUTION OF THE 3-GRAPH.
.*4)IF THE SUC1 OR SUC2 OF A BASIC ACTION IS THE ACTION DIRECTLY
.*FOLLOWING IT,IT NEED NOT BE EXPLICITLY MENTIONED.IF SUC1 IS IMPLICIT
.*WHILE SUC2 IS EXPLICIT THE SEPARATING COMMA MUST BE WRITTEN E.G.
.*1)            COMP A,B,SUCC,NEXT
.*      NEXT    TMV  ....
.*IS EQUIVALENT TO
.*              COMP A,B,SUCC
.*              TMV  ....                    SUC2 NOT EXPLICIT
.*2)            COMP A,B,NEXT,SUCC
.*      NEXT    TMV  ....
.*IS EQUIVALENT TO
.*              COMP A,B,,SUCC
.*              TMV  ....                    SUC1 NOT EXPLICIT
.*5)THE STRING 'NIL' CAN BE USED TO INDICATE THE NOTION 'NIL'(TSTAR MACRO)
.*      ***********************************************************
        MEND
```

```
COMP     01333800:01340800 12/20/74 12/20/74

         MACRO
&NAME    COMP  &ARG1,&ARG2,&SUC1,&SUC2
.*GENERAL DESCRIPTION
.*       ***************************************************
.*COMP IS A BASIC OPERATION ON A 3-GRAPH;THE NOTION SPECIFIED BY THE
.*VALUE OF ARG1 IS COMPARED TO THAT SPECIFIED BY THE VALUE OF ARG2.IF
.*THEY ARE THE SAME THEN THE NEXT ACTION TO BE TAKEN IS THAT SPECIFIED
.*BY THE VALUE OF SUC1 OTHERWISE THE NEXT ACTION IS SPECIFIED BY THE
.*VALUE OF SUC2.
.*       ***************************************************
.*EXPANSION OF THIS MACRO AND THE MACROS IT CALLS GENERATES A NEW
.*NOTION IN THE 3-GRAPH WHICH IS AN OCCURRENCE OF THE BASIC ACTION-COMP
.*THIS NOTION MAY BE NAMED (&NAME PARAMETER).
.*       ***************************************************
.*DEFINITION OF THE NEW NOTION
         TNAME &NAME
.*EXPANSION OF THE TNAME MACRO ALLOCATES 2CONTIGUOUS 20-BYTE BLOCKS
.*FOR THE NEW NOTION AND DEFINES THE NAME IF THE NOTION IS NAMED.
.*       ***************************************************
.*STATISTICS CELL
         AIF   ('&NAME' EQ '').NONAM
         DC    XL4'80000000'
         AGO   .OCCDE
.NONAM   DC    F'0'
.*THE FIRST CELL IS RESERVED FOR STATISTICS ABOUT THE NOTION AND IS
.*INITIALISED TO 0. A FLAG BIT IS SET IF THE NOTION IS NAMED.
.*       ***************************************************
.*OCCDE CELL
.OCCDE   FLAG  BASE
         DC    AL3(0)
.*THE (2ND CELL) SPECIFY THE ACTION WHICH THE NOTION IS AN OCCURRENCE
.*OF.THE FLAG BITS SET INDICATE THAT THIS IS A BASIC ACTION.
.*       ***************************************************
.* SUC1 AND SUC2 CELLS
         TESDF &SUC1,F1
         TESDF &SUC2,F2
.*EXPANSION OF THE TESDF MACRO TESTS FOR THE ABSENCE OF AN EXPLICIT
.*SUCCESSOR IN WHICH CASE A FLAG IS SET TO INDICATE THAT THE SUCCESSOR
.*IS THE NEXT SEQUENTIAL ACTION.IF THE ACTUAL PARAMETER = '*' THEN THE
.*CELL IS LEFT EMPTY.OTHERWISE THE ADDRESS OF THE NOTION SPECIFIED IS
.* FILLED IN.
.*       ***************************************************
.*POINTER CELL
         FLAG  NOT
         DC    AL3(*+3)
.*THE FIRST CLOCK IS CHAINED  TO ITS SUCCESSOR WHICH IN THIS CASE IS
.*CONTIGUOUS.THE FLAG BITS IN A POINTER CELL ARE ALL ZEROES.
.*       ***************************************************
.*ARG1,ARG2 CELLS
         TESDF &ARG1,M1
```

COMP 01333800:01340800 12/20/74 12/20/74

```
          TLSDF &ARG2,Y2
.*EXPANSION OF THE TESDF MACRO,AND THE MACROS IT CALLS,DEAL WITH THE
.*POSSIBLE VALUES AN ARG CELL MAY CONTAIN.ABSENCE OF THE CORRESPONDING
.*ACTUAL PARAMETER PROVOKES AN ERROR MESSAGE.AN '*' OR THE NAME OF A
.*3-GRAPH NOTION OR A LIST OF SUCH NOTIONS ARE THE POSSIBLE OPTIONS.
.*        **********************************************
.*ARG3,ARG4 CELLS
          DC    2X'FFFFFFC'
*NO FLAGS IN EMPTY CELLS
.*THE CORRESPONDING CELLS ARE INITIALISED TO EMPTY SINCE THESE ARGUMENTS
.*DO NOT OCCUR IN THE BASIC ACTION OF WHICH THIS NOTION IS AN OCCURRENCE
.*        **********************************************
.*POINTER CELL
          DC    X'FFFFFFF'
*NO FLAGS IN POINTER CELL
.*AT PRESENT THIS NOTION HAS NO ATTRIBUTES OTHER THAN THOSE JUST
.*DEFINED SO THIS CELL IS INITIALISED TO INDICATE 'END OF NOTION'.
          ORG
          MEND
```

```
CREN     01340900:013482C0 12/20/74 12/20/74

         .MACRO
&NAME    CREN  &ARG1,&ARG2,&SUC1
.*GENERAL DESCRIPTION
.*       ********************************************************
.*CREN IS A BASIC OPERATION ON A 3-GRAPH.A NEW NOTION IS CREATED IN THE
.*3-GRAPH AND IT BECOMES A VALUE OF THE ATTRIBUTE WHOSE NATURE IS THE
.*VALUE OF ARG2 AND ROOT THE VALUE OF ARG1.THE NEXT ACTION TO BE EXECUT
.*ED IS SPECIFIED BY THE VALUE OF SUC1.
.*       ********************************************************
.*EXPANSION OF THIS MACRO AND THE MACROS IT CALLS GENERATES A NEW
.*NOTION IN THE 3-GRAPH WHICH IS AN OCCURRENCE OF THE BASIC ACTION-CREN
.*THIS NOTION MAY BE NAMED (&NAME PARAMETER).
.*       ********************************************************
.*DEFINITION OF THE NEW NOTION
         TNAME &NAME
.*EXPANSION OF THE TNAME MACRO ALLOCATES 2 CONTIGUOUS 20-BYTE BLOCKS
.*FOR THE NEW NOTION AND DEFINES THE NAME IF THE NOTION IS NAMED.
.*       ********************************************************
.*STATISTICS CELL
         AIF   ('&NAME' EQ '').NONAM
         DC    XL4'80000000'
         AGO   .OCCDE
.NONAM   DC    F'0'
.*THE FIRST CELL IS RESERVED FOR STATISTICS ABOUT THE NOTION AND IS
.*INITIALISED TO 0. A FLAG BIT IS SET IF THE NOTION IS NAMED.
.*       ********************************************************
.*OCCDE CELL
.OCCDE   FLAG  BASE
         DC    AL3(4)
.* THE (2ND CELL) SPECIFY THE ACTION WHICH THIS NOTION IS AN OCCURRENCE
.*OF.THE FLAG BITS SET INDICATE THAT THIS IS A BASIC ACTION.
.*       ********************************************************
.*SUC1 CELL
         TESDF &SUC1,F1
.*EXPANSION OF THE TESDF MACRO TESTS FOR THE ABSENCE OF AN EXPLICIT
.*SUCCESSOR IN WHICH CASE A FLAG IS SET TO INDICATE THAT THE SUCCESSOR
.*IS THE NEXT SEQUENTIAL ACTION.IF THE ACTUAL PARAMETER = '*' THEN THE
.*CELL IS LEFT EMPTY OTHERWISE THE ADDRESS OF THE NOTION SPECIFIED IS
.*FILLED IN.
.*       ********************************************************
.*SUC2 CELL
         DC    X'FFFFFFE'
.*THE CORRESPONDING CELL IS INITIALISED TO EMPTY SINCE THIS ACTION
.*CANNOT CAUSE A CHOICE OF SUCCESSOR.
.*       ********************************************************
.*POINTER CELL
         FLAG  NOT
         DC    AL3(*+3)
.*THE FIRST BLOCK IS CHAINED TO ITS SUCCESSOR WHICH IN THIS CASE IS
.*CONTIGUOUS.THE FLAG BITS IN A POINTER CELL ARE ALL ZEROES.
```

```
CREN     01340900:01348200 12/20/74 12/20/74

.*       ***********************************************************
.*ARG1,ARG2 CELLS
         TESDF &ARG1,*1
         TESDF &ARG2,*2
.*EXPANSION OF THE TESDF MACRO,AND THE MACROS IT CALLS,DEAL WITH THE
.*POSSIBLE VALUES AN ARG CELL MAY CONTAIN.ABSENCE OF THE CORRESPONDING
.*ACTUAL PARAMETER PROVOKES AN ERROR MESSAGE.AN '*' OR THE NAME OF A
.*3-GRAPH NOTION OR A LIST OF SUCH NOTIONS ARE THE POSSIBLE OPTIONS.
.*       ***********************************************************
.*ARG3,ARG4 CELLS
         DC    2X'FFFFFFL'
*NO FLAGS IN EMPTY CELLS
.*THE CORRESPONDING CELLS ARE INITIALISED TO EMPTY SINCE THESE ARGUMENTS
.*DO NOT OCCUR IN THE BASIC ACTION OF WHICH THIS NOTION IS AN OCCURRENCE
.*       ***********************************************************
.*POINTER CELL
         DC    X'FFFFFFF'
*NO FLAGS IN POINTER CELL
.*AT PRESENT THIS NOTION HAS NO ATTRIBUTES OTHER THAN THOSE JUST
.*DEFINED SO THIS CELL IS INITIALISED TO INDICATE 'END OF NOTION'.
         ORG
         MEND
```

```
CRER     01348300:01356100 12/20/74 12/20/74

         MACRO
&NAME    CRER  &ARG1,&ARG2,&ARG3,&SUC1
.*GENERAL DESCRIPTION
.*       ********************************************************
.*CRER IS A BASIC OPERATION ON A 3-GRAPH.A RELATION IS CREATED BETWEEN
.*3 EXISTING NOTIONS,IN EFFECT THE NOTION SPECIFIED BY THE VALUE OF
.*ARG1 IS THE ROOT NOTION TO WHICH AN ATTRIBUTE IS ADDED WHOSE NATURE
.*IS SPECIFIED BY (THE VALUE OF) ARG2 AND WHOSE VALUE IS SPECIFIED BY
.*(THE VALUE OF ) ARG3.THE NEXT ACTION TO BE EXECUTED IS SPECIFIED BY
.*THE VALUE OF SUC1.
.*       ********************************************************
.*EXPANSION OF THIS MACRO AND THE MACROS IT CALLS GENERATES A NEW
.*NOTION IN THE 3-GRAPH WHICH IS AN OCCURRENCE OF THE BASIC ACTION-CRER
.*THIS NOTION MAY BE NAMED (&NAME PARAMETER).
.*       ********************************************************
.*DEFINITION OF THE NEW NOTION
         TNAME &NAME
.*EXPANSION OF THE TNAME MACRO ALLOCATES 2 CONTIGUOUS 20-BYTE BLOCKS
.*FOR THE NEW NOTION AND DEFINES THE NAME IF THE NOTION IS NAMED.
.*       ********************************************************
.*STATISTICS CELL
         AIF   ('&NAME' EQ '').NONAM
         DC    XL4'80000000'
         AGO   .OCCDE
.NONAM   DC    F'0'
.*THE FIRST CELL IS RESERVED FOR STATISTICS ABOUT THE NOTION AND IS
.*INITIALISED TO 0. A FLAG BIT IS SET IF THE NOTION IS NAMED.
.*       ********************************************************
.*OCCDE CELL
.OCCDE   FLAG  BASE
         DC    AL3(36)
.* THE (2ND CELL) SPECIFY THE ACTION WHICH THIS NOTION IS AN OCCURRENCE
.*OF.THE FLAG BITS SET INDICATE THAT THIS IS A BASIC ACTION.
.*       ********************************************************
.*SUC1 CELL
         TESDF &SUC1,F1
.*EXPANSION OF THE TESDF MACRO TESTS FOR THE ABSENCE OF AN EXPLICIT
.*SUCCESSOR IN WHICH CASE A FLAG IS SET TO INDICATE THAT THE SUCCESSOR
.*IS THE NEXT SEQUENTIAL ACTION.IF THE ACTUAL PARAMETER = '*' THEN THE
.*CELL IS LEFT EMPTY OTHERWISE THE ADDRESS OF THE NOTION SPECIFIED IS
.*FILLED IN.
.*       ********************************************************
.*SUC2 CELL
         DC    X'FFFFFFE'
*NO FLAGS IN EMPTY CELL
.*THE CORRESPONDING CELL IS INITIALISED TO EMPTY SINCE THIS ACTION
.*CANNOT CAUSE A CHOICE OF SUCCESSOR.
.*       ********************************************************
.*POINTER CELL
         FLAG  NOT
```

```
CREF     C1348300:01356100 12/20/74 12/20/74

         DC    AL3(*+3)
.*THE FIRST BLOCK IS CHAINED TO ITS SUCCESSOR WHICH IN THIS CASE IS
.*CONTIGUOUS.THE FLAG BITS IN A POINTER CELL ARE ALL ZEROES.
.*       **********************************************************
.*ARG1,ARG2,ARG3 CELLS
         TESDF &ARG1,M1
         TESDF &ARG2,M2
         TESDF &ARG3,M3
.*EXPANSION OF THE TESDF MACRO,AND THE MACROS IT CALLS,DEAL WITH THE
.*POSSIBLE VALUES AN ARG CELL MAY CONTAIN.ABSENCE OF THE CORRESPONDING
.*ACTUAL PARAMETER PROVOKES AN ERROR MESSAGE.AN '*' OR THE NAME OF A
.*3-GRAPH NOTION OR A LIST OF SUCH NOTIONS ARE THE POSSIBLE OPTIONS.
.*       **********************************************************
.*ARG4 CELL
         DC    X'FFFFFFE'
*NO FLAGS IN EMPTY CELL
.*THE CORRESPONDING CELL IS INITIALISED TO EMPTY SINCE THIS ARGUMENT
.*DOES NOT OCCUR IN THE BASIC ACTION OF WHICH THIS NOTION IS AN
.*OCCURRENCE.
.*       **********************************************************
.*POINTER CELL
         DC    X'FFFFFFF'
*NO FLAGS IN POINTER CELL
.*AT PRESENT THIS NOTION HAS NO ATTRIBUTES OTHER THAN THOSE JUST
.*DEFINED SO THIS CELL IS INITIALISED TO INDICATE 'END OF NOTION'.
         ORG
         MEND
```

```
LAT       01442900:01450300 12/20/74 12/20/74

          MACRO
&NAME     LAT   &ARG1,&SUC1
.*GENERAL DESCRIPTION
.*        **********************************************
.*LAT IS A BASIC OPERATION ON A 3-GRAPH.AN ATTRIBUTE OF THE NOTION
.*SPECIFIED BY THE VALUE OF ARG1 IS CHOSEN AT RANDOM.THE NATURE OF THE
.*ATTRIBUTE BECOMES THE VALUE OF ARG2,AND THE VALUE OF THE ATTRIBUTE
.* THE VALUE OF ARG3.THE NEXT ACTION IN THE 3-GRAPH TO BE EXECUTED IS
.*SPECIFIED BY THE VALUE OF SUC1.
.*        **********************************************
.*EXPANSION OF THIS MACRO AND THE MACROS IT CALLS GENERATES A NEW
.*NOTION IN THE 3-GRAPH WHICH IS AN OCCURRENCE OF THE BASIC ACTION-LAT
.*THIS NOTION MAY BE NAMED (&NAME PARAMETER).
.*        **********************************************
.*DEFINITION OF THE NEW NOTION
          TNAME &NAME
.*EXPANSION OF THE TNAME MACRO ALLOCATES 2 CONTIGUOUS 20-BYTE BLOCKS
.*FOR THE NEW NOTION AND DEFINES THE NAME IF THE NOTION IS NAMED.
.*        **********************************************
.*STATISTICS CELL
          AIF   ('&NAME' EQ '').NONAM
          DC    XL4'80000000'
          AGO   .OCCDE
.NONAM    DC    F'0'
.*THE FIRST CELL IS RESERVED FOR STATISTICS ABOUT THE NOTION AND IS
.*INITIALISED TO 0. A FLAG BIT IS SET IF THE NOTION IS NAMED.
.*        **********************************************
.*OCCDE CELL
.OCCDE    FLAG  BASE
          DC    AL3(12)
.* THE (2ND CELL) SPECIFY THE ACTION WHICH THIS NOTION IS AN OCCURRENCE
.*OF.THE FLAG BITS SET INDICATE THAT THIS IS A BASIC ACTION.
.*        **********************************************
.*SUC1 CELL
          TESDF &SUC1,F1
.*EXPANSION OF THE TESDF MACRO TESTS FOR THE ABSENCE OF AN EXPLICIT
.*SUCCESSOR IN WHICH CASE A FLAG IS SET TO INDICATE THAT THE SUCCESSOR
.*IS THE NEXT SEQUENTIAL ACTION.IF THE ACTUAL PARAMETER = '*' THEN THE
.*CELL IS LEFT EMPTY OTHERWISE THE ADDRESS OF THE NOTION SPECIFIED IS
.*FILLED IN.
.*        **********************************************
.*SUC2 CELL
          DC    X'FFFFFFE'
*NO FLAGS IN EMPTY CELL
.*THE CORRESPONDING CELL IS INITIALISED TO EMPTY SINCE THIS ACTION
.*CANNOT CAUSE A CHOICE OF SUCCESSOR.
.*        **********************************************
.*POINTER CELL
          FLAG  NOT
          DC    AL3(*+3)
```

```
LAT      01442900:01450300 12/20/74 12/20/74

.*THE FIRST BLOCK IS CHAINED TO ITS SUCCESSOR WHICH IN THIS CASE IS
.*CONTIGUOUS.THE FLAG BITS IN A POINTER CELL ARE ALL ZEROES.
.*       **********************************************
.*ARG1 CELL
         TESDF &ARG1,M1
.*EXPANSION OF THE TESDF MACRO,AND THE MACROS IT CALLS,DEAL WITH THE
.*POSSIBLE VALUES AN ARG CELL MAY CONTAIN.ABSENCE OF THE CORRESPONDING
.*ACTUAL PARAMETER PROVOKES AN ERROR MESSAGE.AN '*' OR THE NAME OF A
.*3-GRAPH NOTION OR A LIST OF SUCH NOTIONS ARE THE POSSIBLE OPTIONS.
.*       **********************************************
.*ARG2,ARG3,ARG4 CELLS
         DC    3X'FFFFFFE'
*NO FLAGS IN EMPTY CELLS
.*THE CORRESPONDING CELLS ARE INITIALISED TO EMPTY SINCE THESE ARGUMENTS
.*DO NOT OCCUR IN THE BASIC ACTION OF WHICH THIS NOTION IS AN OCCURRENCE
.*       **********************************************
.*POINTER CELL
         DC    X'FFFFFFF'
*NO FLAGS IN POINTER CELL
.*AT PRESENT THIS NOTION HAS NO ATTRIBUTES OTHER THAN THOSE JUST
.*DEFINED SO THIS CELL IS INITIALISED TO INDICATE 'END OF NOTION'.
         ORG
         MEND
```

```
REF      01463000:01470100 12/20/74 12/20/74

         MACRO
&NAME    REF   &ARG1,&ARG2,&ARG3,&ARG4,&SUC1
.*GENERAL DESCRIPTION
.*       ***********************************************
.*REF IS A BASIC OPERATION ON A 3-GRAPH.THE ATTRIBUTE OF THE NOTION
.*SPECIFIED BY (THE VALUE OF) ARG3 WHOSE NATURE IS (THE VALUE OF) ARG4
.*IS REPLACED BY THE ATTRIBUTE OF THE NOTION SPECIFIED BY (THE VALUE OF
.*) ARG1 WHOSE NATURE IS (THE VALUE OF) ARG2.THE NEXT ACTION TO BE
.*EXECUTED IN THE 3-GRAPH IS SPECIFIED BY THE VALUE OF SUC1.
.*       ***********************************************
.*EXPANSION OF THIS MACRO AND THE MACROS IT CALLS GENERATES A NEW
.*NOTION IN THE 3-GRAPH WHICH IS AN OCCURENCE OF THE BASIC ACTION-REF
.*THIS NOTION MAY BE NAMED (&NAME PARAMETER).
.*       ***********************************************
.*DEFINITION OF THE NEW NOTION
         TNAME &NAME
.*EXPANSION OF THE TNAME MACRO ALLOCATES 2 CONTIGUOUS 20-BYTE BLOCKS
.*FOR THE NEW NOTION AND DEFINES THE NAME IF THE NOTION IS NAMED.
.*       ***********************************************
.*STATISTICS CELL
         AIF   ('&NAME' EQ '').NONAM
         DC    XL4'80000000'
         AGO   .OCCDE
.NONAM   DC    F'0'
.*THE FIRST CELL IS RESERVED FOR STATISTICS ABOUT THE NOTION AND IS
.*INITIALISED TO 0. A FLAG BIT IS SET IF THE NOTION IS NAMED.
.*       ***********************************************
.*OCCDE CELL
.OCCDE   FLAG  BASE
         DC    AL3(16)
.* THE (2ND CELL) SPECIFY THE ACTION WHICH THIS NOTION IS AN OCCURRENCE
.*OF.THE FLAG BITS SET INDICATE THAT THIS IS A BASIC ACTION.
.*       ***********************************************
.*SUC1 CELL
         TESDF &SUC1,F1
.*EXPANSION OF THE TESDF MACRO TESTS FOR THE ABSENCE OF AN EXPLICIT
.*SUCCESSOR IN WHICH CASE A FLAG IS SET TO INDICATE THAT THE SUCCESSOR
.*IS THE NEXT SEQUENTIAL ACTION.IF THE ACTUAL PARAMETER = '*' THEN THE
.*CELL IS LEFT EMPTY OTHERWISE THE ADDRESS OF THE NOTION SPECIFIED IS
.*FILLED IN.
.*       ***********************************************
.*SUC2 CELL
         DC    X'FFFFFFE'
*NO FLAGS IN EMPTY CELL
.*THE CORRESPONDING CELL IS INITIALISED TO EMPTY SINCE THIS ACTION
.*CANNOT CAUSE A CHOICE OF SUCCESSOR.
.*       ***********************************************
.*POINTER CELL
         FLAG  NOT
         DC    AL3(*+3)
```

```
REP     01463000:01470100 12/20/74 12/20/74

.*THE FIRST BLOCK IS CHAINED TO ITS SUCCESSOR WHICH IN THIS CASE IS
.*CONTIGUOUS.THE FLAG BITS IN A POINTER CELL ARE ALL ZEROES.
.*      **********************************************
.*ARG1,ARG2,ARG3,ARG4 CELLS
        TESDF &ARG1,M1
        TESDF &ARG2,M2
        TESDF &ARG3,M3
        TESDF &ARG4,M4
.*EXPANSION OF THE TESDF MACRO,AND THE MACROS IT CALLS,DEAL WITH THE
.*POSSIBLE VALUES AN ARG CELL MAY CONTAIN.ABSENCE OF THE CORRESPONDING
.*ACTUAL PARAMETER PROVOKES AN ERROR MESSAGE.AN '*' OR THE NAME OF A
.*3-GRAPH NOTION OR A LIST OF SUCH NOTIONS ARE THE POSSIBLE OPTIONS.
.*      **********************************************
.*POINTER CELL
        DC    X'FFFFFFF'
*NO FLAGS IN POINTER CELL
.*AT PRESENT THIS NOTION HAS NO ATTRIBUTES OTHER THAN THOSE JUST
.*DEFINED SO THIS CELL IS INITIALISED TO INDICATE 'END OF NOTION'.
        ORG
        MEND
```

```
SUP       01490200:01497600 12/20/74 12/20/74

          MACRO
&NAME     SUP   &ARG1,&ARG2,&SUC1
.*GENERAL DESCRIPTION
.*        **********************************************
.*SUP IS A BASIC OPERATION ON A 3-GRAPH.THE ATTRIBUTE WHOSE ROOT IS
.*SPECIFIED BY (THE VALUE OF) ARG1 AND NATURE BY (THE VALUE OF) ARG2 IS
.*REMOVED FROM THE 3-GRAPH.THE NEXT ACTION IN THE 3-GRAPH TO BE
.*EXECUTED IS SPECIFIED BY THE VALUE OF SUC1.
.*        **********************************************
.*EXPANSION OF THIS MACRO AND THE MACROS IT CALLS GENERATES A NEW
.*NOTION IN THE 3-GRAPH WHICH IS AN OCCURRENCE OF THE BASIC ACTION-SUP
.*THIS NOTION MAY BE NAMED (&NAME PARAMETER).
.*        **********************************************
.*DEFINITION OF THE NEW NOTION
          TNAME &NAME
.*EXPANSION OF THE TNAME MACRO ALLOCATES 2 CONTIGUOUS 20-BYTE BLOCKS
.*FOR THE NEW NOTION AND DEFINES THE NAME IF THE NOTION IS NAMED.
.*        **********************************************
.*STATISTICS CELL
          AIF   ('&NAME' EQ '').NONAM
          DC    XL4'80000000'
          AGO   .OCCDF
.NONAM    DC    F'0'
.*THE FIRST CELL IS RESERVED FOR STATISTICS ABOUT THE NOTION AND IS
.*INITIALISED TO 0. A FLAG BIT IS SET IF THE NOTION IS NAMED.
.*        **********************************************
.*OCCDF CELL
.OCCDE    FLAG  BASE
          DC    AL3(20)
.* THE (2ND CELL) SPECIFY THE ACTION WHICH THIS NOTION IS AN OCCURRENCE
.*OF.THE FLAG BITS SET INDICATE THAT THIS IS A BASIC ACTION.
.*        **********************************************
.*SUC1 CELL
          TESDF &SUC1,F1
.*EXPANSION OF THE TESDF MACRO TESTS FOR THE ABSENCE OF AN EXPLICIT
.*SUCCESSOR IN WHICH CASE A FLAG IS SET TO INDICATE THAT THE SUCCESSOR
.*IS THE NEXT SEQUENTIAL ACTION.IF THE ACTUAL PARAMETER = '*' THEN THE
.*CELL IS LEFT EMPTY OTHERWISE THE ADDRESS OF THE NOTION SPECIFIED IS
.*FILLED IN.
.*        **********************************************
.*SUC2 CELL
          DC    X'FFFFFFE'
*NO FLAGS IN EMPTY CELL
.*THE CORRESPONDING CELL IS INITIALISED TO EMPTY SINCE THIS ACTION
.*CANNOT CAUSE A CHOICE OF SUCCESSOR.
.*        **********************************************
.*POINTER CELL
          FLAG  NOT
          DC    AL3(*+3)
.*THE FIRST BLOCK IS CHAINED TO ITS SUCCESSOR WHICH IN THIS CASE IS
```

```
SUP       01490200:01497600 12/20/74 12/20/74

.*CONTIGUOUS.THE FLAG BITS IN A POINTER CELL ARE ALL ZEROES.
.*        **********************************************
.*ARG1,ARG2 CELLS
          TESDF &ARG1,1
          TESDF &ARG2,2
.*EXPANSION OF THE TESDF MACRO,AND THE MACROS IT CALLS,DEAL WITH THE
.*POSSIBLE VALUES AN ARG CELL MAY CONTAIN.ABSENCE OF THE CORRESPONDING
.*ACTUAL PARAMETER PROVOKES AN ERROR MESSAGE.AN '*' OR THE NAME OF A
.*3-GRAPH NOTION OR A LIST OF SUCH NOTIONS ARE THE POSSIBLE OPTIONS.
.*        **********************************************
.*ARG3,ARG4 CELLS
          DC    2X'FFFFFFC'
*NO FLAGS IN EMPTY CELLS
.*THE CORRESPONDING CELLS ARE INITIALISED TO EMPTY SINCE THESE ARGUMENTS
.*DO NOT OCCUR IN THE BASIC ACTION OF WHICH THIS NOTION IS AN OCCURRENCE
.*        **********************************************
.*POINTER CELL
          DC    X'FFFFFFF'
*NO FLAGS IN POINTER CELL
.*AT PRESENT THIS NOTION HAS NO ATTRIBUTES OTHER THAN THOSE JUST
.*DEFINED SO THIS CELL IS INITIALISED TO INDICATE 'END OF NOTION'.
          ORG
          MEND
```

```
SUPP      01505300:01512900 12/20/74 12/20/74

          MACRO
&NAME     SUPR  &ARG1,&ARG2,&ARG3,&SUC1
.*GENERAL DESCRIPTION
.*        ********************************************
.*SUPR IS A BASIC ACTION ON A 3-GRAPH.THE RELATION  SPECIFIED BY THE
.*VALUES OF ARG1 (ROOT NOTION),ARG2 (NATURE OF ATTRIBUTE),AND ARG3
.*(VALUE OF ATTRIBUTE) IS REMOVED FROM THE 3-GRAPH.THE NEXT ACTION TO
.*BE EXECUTED IS SPECIFIED BY THE VALUE OF SUC1.
.*        ********************************************
.*EXPANSION OF THIS MACRO AND THE MACROS IT CALLS GENERATES A NEW
.*NOTION IN THE 3-GRAPH WHICH IS AN OCCURRENCE OF THE BASIC ACTION-SUPR
.*THIS NOTION MAY BE NAMED (&NAME PARAMETER).
.*        ********************************************
.*DEFINITION OF THE NEW NOTION
          TNAME &NAME
.*EXPANSION OF THE TNAME MACRO ALLOCATES 2 CONTIGUOUS 20-BYTE BLOCKS
.*FOR THE NEW NOTION AND DEFINES THE NAME IF THE NOTION IS NAMED.
.*        ********************************************
.*STATISTICS CELL
          AIF   ('&NAME' EQ '').NONAM
          DC    XL4'80000000'
          AGO   .OCCDE
.NONAM    DC    F'0'
.*THE FIRST CELL IS RESERVED FOR STATISTICS ABOUT THE NOTION AND IS
.*INITIALISED TO 0. A FLAG BIT IS SET IF THE NOTION IS NAMED.
.*        ********************************************
.*OCCDE CELL
.OCCDE    FLAG  BASE
          DC    AL3(23)
.* THE (2ND CELL) SPECIFY THE ACTION WHICH THIS NOTION IS AN OCCURRENCE
.*OF.THE FLAG BITS SET INDICATE THAT THIS IS A BASIC ACTION.
.*        ********************************************
.*SUC1 CELL
          TESOF &SUC1,F1
.*EXPANSION OF THE TESOF MACRO TESTS FOR THE ABSENCE OF AN EXPLICIT
.*SUCCESSOR IN WHICH CASE A FLAG IS SET TO INDICATE THAT THE SUCCESSOR
.*IS THE NEXT SEQUENTIAL ACTION.IF THE ACTUAL PARAMETER = '*' THEN THE
.*CELL IS LEFT EMPTY OTHERWISE THE ADDRESS OF THE NOTION SPECIFIED IS
.*FILLED IN.
.*        ********************************************
.*SUC2 CELL
          DC    X'FFFFFFE'
*NO FLAGS IN EMPTY CELL
.*THE CORRESPONDING CELL IS INITIALISED TO EMPTY SINCE THIS ACTION
.*CANNOT CAUSE A CHOICE OF SUCCESSOR.
.*        ********************************************
.*POINTER CELL
          FLAG  NOT
          DC    AL3(*+3)
.*THE FIRST BLOCK IS CHAINED TO ITS SUCCESSOR WHICH IN THIS CASE IS
```

```
SUPR     01505300:01512900 12/20/74 12/20/74

.*CONTIGUOUS.THE FLAG BITS IN A POINTER CELL ARE ALL ZEROES.
.*       **********************************************
.*ARG1,ARG2,ARG3 CELLS
         TESDF &ARG1,*1
         TESDF &ARG2,*2
         TESDF &ARG3,*3
.*EXPANSION OF THE TESDF MACRO,AND THE MACROS IT CALLS,DEAL WITH THE
.*POSSIBLE VALUES AN ARG CELL MAY CONTAIN.ABSENCE OF THE CORRESPONDING
.*ACTUAL PARAMETER PROVOKES AN ERROR MESSAGE.AN '*' OR THE NAME OF A
.*3-GRAPH NOTION OR A LIST OF SUCH NOTIONS ARE THE POSSIBLE OPTIONS.
.*       **********************************************
.*ARG4 CELL
         DC    X'FFFFFFE'
*NO FLAGS IN EMPTY CELL
.*THE CORRESPONDING CELL IS INITIALISED TO EMPTY SINCE THIS ARGUMENT
.*DOES NOT OCCUR IN THE BASIC ACTION OF WHICH THIS NOTION IS AN
.*OCCURRENCE.
.*       **********************************************
.*POINTER CELL
         DC    X'FFFFFFF'
*NO FLAGS IN POINTER CELL
.*AT PRESENT THIS NOTION HAS NO ATTRIBUTES OTHER THAN THOSE JUST
.*DEFINED SO THIS CELL IS INITIALISED TO INDICATE 'END OF NOTION'.
         ORG
         MEND
```

```
TMV       01534400:01541500 12/20/74 12/20/74

          MACRO
&NAME     TMV   &ARG1,&ARG2,&ARG3,&ARG4,&SUC1
.*GENERAL DESCRIPTION
.*        **********************************************
.*TMV IS A BASIC ACTION ON A 3-GRAPH.THE VALUE OF THE ATTRIBUTE
.*SPECIFIED BY THE VALUES OF ARG1 (ROOT  NOTION) AND ARG2 (NATURE OF THE
.*ATTRIBUTE) BECOMES THE VALUE OF THE ATTRIBUTE SPECIFIED BY THE VALUES
.*OF ARG3 (ROOT NOTION) AND ARG4 (NATURE OF THE ATTRIBUTE).THE NEXT
.*ACTION TO BE TAKEN IS SPECIFIED  BY THE VALUE OF SUC1.
.*        **********************************************
.*EXPANSION OF THIS MACRO AND THE MACROS IT CALLS GENERATES A NEW
.*NOTION IN THE 3-GRAPH WHICH IS AN OCCURRENCE OF THE BASIC ACTION-TMV
.*THIS NOTION MAY BE NAMED (&NAME PARAMETER).
.*        **********************************************
.*DEFINITION OF THE NEW NOTION
          TNAME &NAME
.*EXPANSION OF THE TNAME MACRO ALLOCATES 2 CONTIGUOUS 20-BYTE BLOCKS
.*FOR THE NEW NOTION AND DEFINES THE NAME IF THE NOTION IS NAMED.
.*        **********************************************
.*STATISTICS CELL
          AIF   ('&NAME' EQ '').NONAM
          DC    XL4'80000000'
          AGO   .OCCDE
.NONAM    DC    F'0'
.*THE FIRST CELL IS RESERVED FOR STATISTICS ABOUT THE NOTION AND IS
.*INITIALISED TO 0. A FLAG BIT IS SET IF THE NOTION IS NAMED.
.*        **********************************************
.*OCCDE CELL
.OCCDE    FLAG  BASE
          DC    AL3(32)
.* THE (2ND CELL) SPECIFY THE ACTION WHICH THIS NOTION IS AN OCCURRENCE
.*OF.THE FLAG BITS SET INDICATE THAT THIS IS A BASIC ACTION.
.*        **********************************************
.*SUC1 CELL
          TESDF &SUC1,F1
.*EXPANSION OF THE TESDF MACRO TESTS FOR THE ABSENCE OF AN EXPLICIT
.*SUCCESSOR IN WHICH CASE A FLAG IS SET TO INDICATE THAT THE SUCCESSOR
.*IS THE NEXT SEQUENTIAL ACTION.IF THE ACTUAL PARAMETER = '*' THEN THE
.*CELL IS LEFT EMPTY OTHERWISE THE ADDRESS OF THE NOTION SPECIFIED IS
.*FILLED IN.
.*        **********************************************
.*SUC2 CELL
          DC    X'FFFFFFE'
*NO FLAGS IN EMPTY CELL
.*THE CORRESPONDING CELL IS INITIALISED TO EMPTY SINCE THIS ACTION
.*CANNOT CAUSE A CHOICE OF SUCCESSOR.
.*        **********************************************
.*POINTER CELL
          FLAG  NOT
          DC    AL3(*+3)
```

```
TMV      01534400:01541500 12/20/74 12/20/74

.*THE FIRST BLOCK IS CHAINED TO ITS SUCCESSOR WHICH IN THIS CASE IS
.*CONTIGUOUS.THE FLAG BITS IN A POINTER CELL ARE ALL ZEROES.
.*       **********************************************
.*ARG1,ARG2,ARG3,ARG4 CELLS
         TESDF &ARG1,M1
         TESDF &ARG2,M2
         TESDF &ARG3,M3
         TESDF &ARG4,M4
.*EXPANSION OF THE TESDF MACRO,AND THE MACROS IT CALLS,DEAL WITH THE
.*POSSIBLE VALUES AN ARG CELL MAY CONTAIN.ABSENCE OF THE CORRESPONDING
.*ACTUAL PARAMETER PROVOKES AN ERROR MESSAGE.AN '*' OR THE NAME OF A
.*3-GRAPH NOTION OR A LIST OF SUCH NOTIONS ARE THE POSSIBLE OPTIONS.
.*       **********************************************
.*POINTER CELL
         DC    X'FFFFFFF'
*NO FLAGS IN POINTER CELL
.*AT PRESENT THIS NOTION HAS NO ATTRIBUTES OTHER THAN THOSE JUST
.*DEFINED SO THIS CELL IS INITIALISED TO INDICATE 'END OF NOTION'.
         ORG
         MEND
```

```
DEF      01356200:01377300 12/20/74 01/07/75

         MACRO
&NAME    DEF   &NAT,&VAL
.*GENERAL DESCRIPTION
.*       ************************************************************
.*DEF IS A PSEUDO-OPERATION ON A 3-GRAPH.IT ADDS AN ATTRIBUTE TO A
.*NOTION.IN GENERAL,WE HAVE A GROUP OF DEF CALLS ALL REFERRING TO THE
.*SAME NOTION AND DEFINING SEVERAL ATTRIBUTES.AN EXCEPTION IS A DEF USED
.*TO DEFINE A NOTION WITHOUT ATTRIBUTES BECAUSE ITS NAME IS USED IN
.*OTHER PARTS OF THE 3-GRAPH AND ALL NOTIONS NAMES MUST BE EXPLICITLY
.*DEFINED.SUCH A DEF HAS THE NAME OF THE NOTION IN THE NAME COLUMN
.*(COL 1-8) AND AN * IN THE NATURE AND VALUE COLUMNS (CORRESPONDING
.*TO THE &NAME,&NAT,&VAL FORMAL PARAMETERS RESP).
.*THE FIRST (OR ONLY) DEF OPERATION IN A GROUP MUST NAME THE ROOT NO-
.*TION (&NAME PARAMETER).THIS IS DEFINED AS A NEW NOTION IN THE 3-
.*GRAPH,AND THE OTHER DEF(S) IN THE GROUP (WITHOUT AN &NAME PARAMETER)
.*REFER TO IT IMPLICITLY.
.*EACH DEF DEFINES ONE ATTRIBUTE OF THE ROOT NOTION WHOSE NATURE IS
.*THE 1ST PARAMETER AND WHICH MAY HAVE ONE OR MORE VALUES (NOTION NAMES
.*SEPARATED BY COMMAS).AN ATTRIBUTE IS DEFINED BY ONE AND ONLY ONE DEF
.*(IE THE SAME NATURE MAY NOT APPEAR TWICE IN A DEF GROUP) OTHERWISE
.*THERE ARE NO RESTRICTIONS ON THE NOTIONS WHICH MAY APPEAR.
.*       ************************************************************
.*MACRO VARIABLES.
         GBLB  &DEFCON,&SPOCCDE,&SPNAT,&PASACT,&SPVAL
         GBLA  &PASSNO
         GBLA  &NAMGEN,&NMATB,&NMVAB,&POS
         LCLA  &COUNT,&CNT
.*&DEFCON IS A BOOLEAN INDICATOR.AT THE BEGINNING OF A DEF (DEFS) GROUP
.*IT IS SET TO 0 (WHEN THE MACRO CALL HAS A  NAME (COL 1-8)) AND IT IS
.*SET TO 1 FOR THE OTHER MACRO CALLS BELONGING TO THE SAME GROUP.SO WE
.*KNOW,FOR ANY DEF (DEFS) MACRO CALL WITHOUT A NAME THAT WE ARE
.*CORRECTLY POSITIONED WITHIN A NOTION WHEN DEFINING A NEW ATTRIBUTE.
.*&SPOCCDE INDICATES WHETHER THE NATURE OF THE CURRENT  ATTRIBUTE IS
.*'OCCDE' OR NOT (TSNAT MACRO)
.*&SPNAT INDICATES WHETHER THE NATURE OF THE CURRENT ATTRIBUTE IS ONE OF
.*THE SPECIAL NOTIONS OF THE 3-GRAPH (TSNAT MACRO)
.*&PASACT INDICATES WHETHER THE VALUE OF THE ATTRIBUTE WHOSE NATURE IS
.*'OCCDE' IS A BASIC ACTION OF THE 3-GRAPH (TSOCC MACRO)
.*&SPVAL INDICATES WHETHER THE VALUE OF THE ATTRIBUTE IS ONE OF THE
.*SPECIAL NOTIONS OF THE 3-GRAPH (TSVAL MACRO)
.*&PASSNO IS THE NO OF THE CURRENT 3-GRAPH SECTION
.*&NAMGEN AND &POS ARE USED FOR GENERATING UNIQUE SYMBOLS TO BE
.*ATTACHED TO LOCATION COUNTER  POSITIONS
.*&NMATB INDICATES THE NO OF EMPTY ATTRIBUTES IN THE CURRENT ATTRIBUTE
.*BLOCK
.*&NMVAB INDICATES THE NO OF EMPTY CELLS IN THE CURRENT VALUE LIST
.*BLOCK
.*&COUNT,&CNT ARE COUNTER VARIABLES USED FOR LISTS OF VALUES
.*       ************************************************************
.*NOTE ON GENERATED SYMBOLS.
```

```
DEF       01356200:01377300 12/20/74 01/07/75

.*A)@NO&NAMGEN.THE NATURE OF AN ATTRIBUTE MAY BE ANY NOTION INCLUDING
.*ONE OF THE SPECIAL NOTIONS OF THE 3-GRAPH.WHEN ONE OF THESE OCCURS
.*THE LOCATION COUNTER IS REPOSITIONED RELATIVE TO THE BEGINNING OF THE
.*ROOT NOTION WHOSE ADDRESS MUST ALWAYS BE AVAILABLE EVEN WHEN ITS
.*NAME IS NOT,SO A UNIQUE SYMBOL IS ATTACHED TO IT.
.*B)@NA&POS.AN ATTRIBUTE WHOSE NATURE IS ONE OF THE SPECIAL NOTIONS
.*MAY OCCUR ANYWHERE IN A DEF (DEFS) GROUP AND MAY BE FOLLOWED BY AN
.*ATTRIBUTE WHOSE NATURE IS NOT SPECIAL.SO A UNIQUE SYMBOL IS ATTACHED
.*TO THE CURRENT POSITION IN THE ATTRIBUTE BLOCK WHEN THE NEW ATTRIBUTE
.*IS 'SPECIAL' THEN THE LOCATION COUNTER CAN BE CORRECTLY REPOSITIONED
.*AFTERWARDS.
.*C)@VA&SYSNDX.WHEN AN ATTRIBUTE HAS A LIST OF VALUES THE CURRENT
.*ATTRIBUTE BLOCK IS ABANDONED WHILE A VALUE LIST IS SET UP.THE LOCATION
.*COUNTER CAN BE REPOSITIONED FOR THE FOLLOWING ATTRIBUTE BECAUSE A
.*UNIQUE SYMBOL IS ATTACHED TO THE VALUE CELL POINTING
.*TO THE VALUE LIST.
.*        **********************************************************
.*TEST FOR NEW DEF GROUP,IF NOT GO TO .L1
          AIF   ('&NAME' EQ '').L1
.*        **********************************************************
.*TEST IF NOTION TO BE DEFINED IS ATTRIBUTELESS,IF SO,GO TO .DUMMY
          AIF   (('&NAT' EQ '*') AND ('&VAL' EQ '*')).DUMMY
.*        **********************************************************
.*INITIALISATION AT START OF A DEF GROUP
&NAMGEN   SETA  &NAMGEN+1
&DEFCON   SETB  0
&RMATH    SETA  0
.*        **********************************************************
.*DEFINE THE NEW NOTION WITH THE NAME GIVEN,POSITION AT START OF NOTION
.*AND ATTACH A UNIQUE GENERATED SYMBOL TO THE START OF THE NOTION
.*(TO BE USED FOR POSITIONING IF NECESSARY,SEE NATRE MACRO)
          DEFNOT &NAME&PASSNO
          ORG   &NAME&PASSNO
@NO&NAMGEN EQU &NAME&PASSNO
          AGO   .CONT
.*        **********************************************************
.*IF NO NAME GIVEN TO CURRENT DEF MACRO CALL AND FLAG NOT SET TO INDIC
.*-ATE THAT A DEF GROUP HAS BEEN STARTED,OUTPUT ERROR MESSAGE
.L1       AIF   (&DEFCON NE 1).ERR1
.*        **********************************************************
.*NATURE OF ATTRIBUTE
          NATRE &NAT
          AGO   .CONT1
.CONT     ANOP
&NAME&PASSNO NATRE &NAT
.*NATRE MACRO DEALS WITH THE DIFFERENT POSSIBILITIES E.G. SPECIAL NOTION.
.*AT THE END OF THE EXPANSION THE LOCATION COUNTER IS POSITIONED ON THE
.*VALUE CELL OF THE NEW ATTRIBUTE,AND INDICATORS (&SPOCCDE,&SPNAT,ETC)
.*HAVE BEEN SET.
.*        **********************************************************
```

```
DEF       01356200:01377300 12/20/74 01/07/75

.*TEST FOR LIST OF VALUES ASSOCIATED WITH THE ATTRIBUTE (IF THE 3RD
.*PARAMETER IS NOT A NULL STRING THEN THERE IS A LIST).
.CONT1    AIF    ('&SYSLIST(3)' NE '').LIST
.*        ********************************************************
.*VALUE OF ATTRIBUTE
          AIF    ('&NAME' NE '').N1
          VALUE &VAL
          AGO    .N2
.N1       ANOP
&NAME&PASSNO VALUE &VAL
.*THE VALUE MACRO DEALS WITH THE DIFFERENT POSSIBILITIES AND FILLS IN
.*THE VALUE CELL OF THE ATTRIBUTE
.*        ********************************************************
.*IF THE ATTRIBUTE IS 'SPECIAL' (IE THE NATURE IS ONE OF THE SPECIAL
.*NOTIONS OF THE 3-GRAPH,THE LOCATION COUNTER IS REPOSITIONED (SEE
.*NATRE MACRO)
.N2       AIF    (&SPNAT EQ 0).OUT
          ORG    @NA&POS
.OUT      MEXIT
.*        ********************************************************
.*LIST OF VALUES
.*LOOP WHICH COUNTS THE NO. OF VALUES IN THE LIST.WHEN &COUNT=N
.*&SYSLIST(N) PICKS OUT THE NTH PARAMETER OF THE MACRO CALL.WHEN THIS
.*IS A NULL STRING THE LIST IS EXHAUSTED
.LIST     ANOP
&COUNT    SETA   2
.LOOP     AIF    ('&SYSLIST(&COUNT)' EQ '').FIN
&COUNT    SETA   &COUNT+1
          AGO    .LOOP
.*        ********************************************************
.FIN      ANOP
@VA&SYSNDX EQU *
.*SAVE ADDRESS OF CURRENT VALUE CELL BY ATTACHING A UNIQUE LABEL TO IT
.*        ********************************************************
          ALLOC VAL,20
          SETCNT &COUNT,2
&COUNT    SETA   &COUNT-1
&RMVAR    SETA   3
*ALL FLAG BITS 0 IN COUNT CELL
.*ALLOCATE 20-BYTE BLOCK FOR VALUE LIST,FILL IN COUNTER CELL OF THIS
.*LIST;UPDATE &RMVAR TO INDICATE 3 EMPTY CELLS IN CURRENT VALUE LIST
.*BLOCK
.*        ********************************************************
          AIF    (&SPOCCDE EQ 1).ERA
.*TEST IF NATURE OF ATTRIBUTE WAS 'OCCDE'.TREATMENT OF THE VALUE LIST
.*IN THIS CASE IS DIFFERENT AS THE LIST MAY CONTAIN BASIC ACTIONS OF
.*THE 3-GRAPH
.*        ********************************************************
&CNT      SETA   2
.VALOOP   VALST &SYSLIST(&CNT)
```

```
DEF       01356200:01377300 12/20/74 01/07/75

&CNT      SETA  &CNT+1
          AIF   (&CNT LE &COUNT).VALOOP
.*LOOP DEALS WITH EACH ITEM IN THE VALUE LIST IN TURN CALLING THE VALST
.*MACRO TO FILL IN THE CURRENT VALUE CELL.&SYSLIST (N) PICKS OUT THE
.*NTH PARAMETER OF THE MACRO CALL (IN THIS CASE THE (N-1)ST VALUE IN THE
.*LIST OF VALUES (THE NATURE OF THE ATTRIBUTE IS THE 1ST PARAMETER)).
.*        **********************************************************
.LAB3     AIF   (&RMVAR NE 0).LAB2
          DC    X'FFFFFFFE'
*ALL FLAGS IN POINTER CELL 0
.*AT THE END OF THE VALUE LIST THE CURRENT VALUE LIST BLOCK IS COMPLETED
.*WITH VALUE CELLS INITIALISED TO EMPTY IF THERE ARE ANY LEFT AND THE
.*POINTER CELL INITIALISED TO 'LAST POINTER'
.*        **********************************************************
          ORG   @VA&SYSNDX+4
&DEFCON   SETB  1
.*THE LOCATION COUNTER IS REPOSITIONED TO THE CELL FOLLOWING THE VALUE
.*CELL OF THE ATTRIBUTE JUST DEALT WITH.(@VA&SYSNDX IS ASSOCIATED WITH
.*THE VALUE CELL .&DEFCON IS SET TO INDICATE A DEF (DEFS)
.*GROUP STARTED.
.*        **********************************************************
          AIF   (&SPNAT EQ 1).L2
&RMATR    SETA  &RMATR-1
.L2       AIF   (&SPNAT EQ 0).MEXIT
          ORG   @NA&POS
.MEXIT    MEXIT
.*IF &SPNAT=1 THE NATURE OF THE ATTRIBUTE WAS ONE OF THE SPECIAL NOTIONS
.*SO A)&RMATR IS NOT UPDATED (NO ATTRIBUTE SPACE HAS BEEN USED)
.*   B)THE LOCATION COUNTER IS REPOSITIONED ON THE NEXT FREE ATTRIBUTE
.*IN THE CURRENT ATTRIBUTE BLOCK WHOSE ADDRESS WAS SAVED WHEN THE NATURE
.*WAS IDENTIFIED AS SPECIAL (NATRE AND TSNAT MACROS).
.*        **********************************************************
.*INITIALISE CELL TO EMPTY AND UPDATE &RMVAR
.LAB2     DC    X'FFFFFFFE'
*ALL FLAGS 0 IN EMPTY CELL
&RMVAR    SETA  &RMVAR-1
          AGO   .LAB3
.*        **********************************************************
.*NATURE OF ATTRIBUTE IS OCCDE WITH A LIST OF VALUES ASSOCIATED.
.*LOOP ITERATES OVER THE LIST.THE MACRO LISTVAL COMPLETES THE
.*CORRESPONDING CELL IN THE 3-GRAPH.THE END OF THE LIST IS GIVEN BY
.*THE VALUE OF &COUNT .
.ERA      AMOP
&CNT      SETA  2
.TSLOOP   LISTVAL &SYSLIST(&CNT)
&CNT      SETA  &CNT+1
          AIF   (&CNT LE &COUNT).TSLOOP
          AGO   .LAB3
.*        **********************************************************
.*ERROR MESSAGE
```

```
DEF       01356200:01377300 12/20/74 01/07/75

.ERR1     MNOTE 'FIRST DEF IN A GROUP MUST HAVE A NAME'
          MEXIT
.*        ********************************************************
.*DEFINE NCTION WITH NO ATTRIBUTES,SET ALL INDICATORS TO 0 (FALSE).
.DUMMY    DEFNOT &NAME&PASSNO
&DEFCON   SETB  0
&SPOCCDE  SETB  0
&SPNAT    SETB  0
&BASACT   SETB  0
&SPVAL    SETC  0
&RMATB    SETA  0
&PMVAD    SETA  0
          MEND
```

```
DEFS     01383500:01402900 12/20/74 12/20/74

         MACRO
         DEFS  &NAME,&NAT,&VAL
.*GENERAL DESCRIPTION
.*       ********************************************************
.*THE DEFS PSEUDO-OPERATION PERFORMS THE SAME FUNCTIONS AS THE DEF
.*MACRO FOR A ROOT NOTION  ALREADY  DEFINED EITHER AS AN
.*OCCURRENCE OF A BASIC ACTION OR BY A PREVIOUS DEF GROUP.
.*THIS MEANS THAT THE FIRST DEFS OF A GROUP MUST EXPLICITLY NAME THE
.*ROOT NOTION CONCERNED (&NAME PARAMETER) WHICH MUST PREVIOUSLY HAVE
.*BEEN DEFINED AND THAT THE NATURE OF ANY ATTRIBUTE OCCURRING IN A DEFS
.*MUST NOT HAVE ALREADY APPEARED.THIS APPLIES IN PARTICULAR TO THE
.*SPECIAL NOTIONS WHERE THERE ARE TOO MANY VALUES FOR THE ATTRIBUTE
.*TO APPEAR IN A BASIC ACTION MACRO CALL.THE VALUES CANNOT BE PARTLY
.*LISTED IN THE BASIC ACTION MACRO CALL AND PARTLY IN A DEFS.
.*HOWEVER,UNLIKE A DEF GROUP,A SAME ROOT NOTION CAN HAVE ATTRIBUTES
.*DEFINED BY MORE THAN ONE DEFS GROUP.
.*       ********************************************************
.*MACRO VARIABLES.
         GBLB  &DEFCON,&SPOCCDE,&SPNAT,&BASACT,&SPVAL
         GBLA  &NAMGEN,&RMATB,&PMVAB,&POS
         GBLA  &PASSNO
         LCLA  &COUNT,&CNT
.*&DEFCON IS A BOOLEAN INDICATOR.AT THE BEGINNING OF A DEF (DEFS) GROUP
.*IT IS SET TO 0 (WHEN THE MACRO CALL HAS A  NAME ) AND IT IS
.*SET TO 1 FOR THE OTHER MACRO CALLS BELONGING TO THE SAME GROUP.SO WE
.*KNOW,FOR ANY DEF (DEFS) MACRO CALL WITHOUT A NAME THAT WE ARE
.*CORRECTLY POSITIONED WITHIN A NOTION WHEN DEFINING A NEW ATTRIBUTE.
.*&SPOCCDE INDICATES WHETHER THE NATURE OF THE CURRENT  ATTRIBUTE IS
.*'OCCDE' OR NOT (TSNAT MACRO)
.*&SPNAT INDICATES WHETHER THE NATURE OF THE CURRENT ATTRIBUTE IS ONE OF
.*THE SPECIAL NOTIONS OF THE 3-GRAPH (TSNAT MACRO)
.*&BASACT INDICATES WHETHER THE VALUE OF THE ATTRIBUTE WHOSE NATURE IS
.*'OCCDE' IS A BASIC ACTION OF THE 3-GRAPH (TSOCC MACRO)
.*&SPVAL INDICATES WHETHER THE VALUE OF THE ATTRIBUTE IS ONE OF THE
.*SPECIAL NOTIONS OF THE 3-GRAPH (TSVAL MACRO)
.*&NAMGEN AND &POS ARE USED FOR GENERATING UNIQUE SYMBOLS TO BE
.*ATTACHED TO LOCATION COUNTER  POSITIONS
.*&RMATB INDICATES THE NO OF EMPTY ATTRIBUTES IN THE CURRENT ATTRIBUTE
.*BLOCK
.*&RMVAB INDICATES THE NO OF EMPTY CELLS IN THE CURRENT VALUE LIST
.*BLOCK
.*&COUNT,&CNT ARE COUNTER VARIABLES USED FOR LISTS OF VALUES
.*&PASSNO IS THE NO OF THE CURRENT 3-GRAPH SECTION
.*       ********************************************************
.*NOTE ON GENERATED SYMBOLS.
.*A)&NO&NAMGEN.THE NATURE OF AN ATTRIBUTE MAY BE ANY NOTION INCLUDING
.*ONE OF THE SPECIAL NOTIONS OF THE 3-GRAPH.WHEN ONE OF THESE OCCURS
.*THE LOCATION COUNTER IS REPOSITIONED RELATIVE TO THE BEGINNING OF THE
.*ROOT NOTION WHOSE ADDRESS MUST ALWAYS BE AVAILABLE EVEN WHEN ITS
.*NAME IS NOT,SO A UNIQUE SYMBOL IS ATTACHED TO IT.
```

```
DEFS     01383500:01402900 12/20/74 12/20/74

.*B)&NAGPOS.AN ATTRIBUTE WHOSE NATURE IS ONE OF THE SPECIAL NOTIONS
.*MAY OCCUR ANYWHERE IN A DEF (DEFS) GROUP AND MAY BE FOLLOWED BY AN
.*ATTRIBUTE WHOSE NATURE IS NOT SPECIAL.SO A UNIQUE SYMBOL IS ATTACHED
.*TO THE CURRENT POSITION IN THE ATTRIBUTE BLOCK WHEN THE NEW ATTRIBUTE
.*IS 'SPECIAL' THEN THE LOCATION COUNTER CAN BE CORRECTLY REPOSITIONED
.*AFTERWARDS.
.*C)&VA&SYSNDX.WHEN AN ATTRIBUTE HAS A LIST OF VALUES THE CURRENT
.*ATTRIBUTE BLOCK IS ABANDONED WHILE A VALUE LIST IS SET UP.THE LOCATION
.*COUNTER CAN BE REPOSITIONED FOR THE FOLLOWING ATTRIBUTE BECAUSE A
.*UNIQUE SYMBOL IS ATTACHED TO THE VALUE CELL POINTING
.*TO THE VALUE LIST.
.*       ************************************************************
.*TEST IF CURRENT DEFS MACRO CALL IS FIRST IN A GROUP.
         AIF   ('&NAME' EQ '').DEFS1
.*       ************************************************************
.*IF SO,POSITION LOCATION COUNTER TO START OF NOTION (NOTE:IF THE
.*CONDITIONS FOR THE USE OF THIS MACRO HAVE NOT BEEN FULFILLED AN
.*ASSEMBLER ERROR MESSAGE WILL BE PRODUCED HERE).
         ORG   &NAME&PASSNO
.*       ************************************************************
.*INITIALISATION AT START OF A DEFS GROUP.
&NAMGEN  SETA  &NAMGEN+1
&DEFCON  SETB  0
&RMATB   SETA  0
.*       ************************************************************
.*ATTACH UNIQUE GENERATED SYMBOL TO START OF NOTION (USED FOR
.*REPOSITIONING,SEE NATRE MACRO)
@NO&NAMGEN EQU &NAME&PASSNO
         AGO   .DEFS2
.*       ************************************************************
.*IF &NAME PARAMETER IS NULL AND INDICATOR SHOWS THAT A DEFS GROUP HAS
.*NOT BEEN STARTED OUTPUT ERROR MESSAGE.
.DEFS1   AIF   (&DEFCON NE 1).ERR
         NATRL &NAT
         AGO   .DEFS7
.*       ************************************************************
.*NATURE OF ATTRIBUTE
.DEFS2   ANOP
&NAME&PASSNO NATRE &NAT
.*NATRE MACRO DEALS WITH THE DIFFERENT POSSIBILITIES E.G. SPECIAL NOTION.
.*AT THE END OF THE EXPANSION THE LOCATION COUNTER IS POSITIONED ON THE
.*VALUE CELL OF THE NEW ATTRIBUTE,AND INDICATORS (&SPOCCDE,&SPNAT,ETC)
.*HAVE BEEN SET.
.*       ************************************************************
.*TEST FOR LIST OF VALUES ASSOCIATED WITH THE ATTRIBUTE (IF THE 3RD
.*PARAMETER IS NOT A NULL STRING THEN THERE IS A LIST).
.DEFS7   AIF   ('&SYSLIST(4)' NE '').DEFS3
.*       ************************************************************
.*VALUE OF ATTRIBUTE
         AIF   ('&NAME' EQ '').DEFS8
```

```
DEFS     01383500:01402900 12/20/74 12/20/74

&NAME&PASSNO VALUE &VAL
         AGO   .DEFS9
.DEFS8   VALUE &VAL
.*THE VALUE MACRO DEALS WITH THE DIFFERENT POSSIBILITIES AND FILLS IN
.*THE VALUE CELL OF THE ATTRIBUTE
.*       ********************************************************
.*IF NATURE OF ATTRIBUTE IS ONE OF THE SPECIAL NOTIONS THE LOCATION COUNTER
.*IS REPOSITIONED TO WHERE IT WAS BEFORE THIS ATTRIBUTE WAS DEALT WITH
.*(SEE GENERAL DESCRIPTION).
.DEFS9   AIF   (&SPNAT EQ 0).L1
         ORG   @NA&POS
.L1      MEXIT
.*       ********************************************************
.*LIST OF VALUES
.*LOOP WHICH COUNTS THE NO. OF VALUES IN THE LIST.WHEN &COUNT=N
.*&SYSLIST(N) PICKS OUT THE NTH PARAMETER OF THE MACRO CALL.WHEN THIS
.*IS A NULL STRING THE LIST IS EXHAUSTED
.DEFS3   ANOP
&COUNT   SETA  3
.FOUC    AIF   ('&SYSLIST(&COUNT)' EQ '').END
&COUNT   SETA  &COUNT+1
         AGO   .FOUC
.*       ********************************************************
.END     ANOP
@VA&SYSNDX EQU *
.*SAVE ADDRESS OF CURRENT VALUE CELL BY ATTACHING A UNIQUE LABEL TO IT
.*       ********************************************************
         ALLOC VAL,20
         SETCNT &COUNT,3
&COUNT   SETA  &COUNT-1
&RMVAR   SETA  3
*ALL FLAG BITS 0 IN COUNT CELL
.*ALLOCATE 20-BYTE BLOCK FOR VALUE LIST,FILL IN COUNTER CELL OF THIS
.*LIST;UPDATE &RMVAR TO INDICATE 3 EMPTY CELLS IN CURRENT VALUE LIST
.*BLOCK
.*       ********************************************************
         AIF   (&SPOCCDE EQ 1).DEFS4
.*TEST IF NATURE OF ATTRIBUTE WAS 'OCCDE'.TREATMENT OF THE VALUE LIST
.*IN THIS CASE IS DIFFERENT AS THE LIST MAY CONTAIN BASIC ACTIONS.
.*       ********************************************************
&CNT     SETA  3
.VALOOP  VALST &SYSLIST(&CNT)
&CNT     SETA  &CNT+1
         AIF   (&CNT LE &COUNT).VALOOP
.*LOOP DEALS WITH EACH ITEM IN THE VALUE LIST IN TURN CALLING THE VALST
.*MACRO TO FILL IN THE CURRENT VALUE CELL.&SYSLIST (N) PICKS OUT THE
.*NTH PARAMETER OF THE MACRO CALL (IN THIS CASE THE (N-1)ST VALUE IN THE
.*LIST OF VALUES (THE NATURE OF THE ATTRIBUTE IS THE 1ST PARAMETER)).
.*       ********************************************************
.DEFS5   AIF   (&RMVAR NE 0).DEFS6
```

```
DEFS     01383500:01402900 12/20/74 12/20/74

         DC    X'FFFFFFF'
*ALL FLAGS IN POINTER CELL 0
.*AT THE END OF THE VALUE LIST THE CURRENT VALUE LIST BLOCK IS COMPLETED
.*WITH VALUE CELLS INITIALISED TO EMPTY IF THERE ARE ANY LEFT AND THE
.*POINTER CELL INITIALISED TO 'LAST POINTER'
.*       ********************************************************
         ORG   @VA&SYSNDX+4
&DEFCON  SETB  1
.*THE LOCATION COUNTER IS REPOSITIONED TO THE CELL FOLLOWING THE VALUE
.*CELL OF THE ATTRIBUTE JUST DEALT WITH.(@VA&SYSNDX IS ASSOCIATED WITH
.*THE VALUE CELL .&DEFCON IS SET TO INDICATE A DEF (DEFS)
.*GROUP STARTED.
.*       ********************************************************
         AIF   (&SPNAT EQ 1).L3
&RMATB   SETA  &RMATB-1
.L3      AIF   (&SPNAT EQ 0).L2
         ORG   @NA&POS
.L2      MEXIT
.*IF &SPNAT=1 THE NATURE OF THE ATTRIBUTE WAS ONE OF THE SPECIAL NOTIONS
.*SO A)&RMATB IS NOT UPDATED (NO ATTRIBUTE SPACE HAS BEEN USED)
.*    B)THE LOCATION COUNTER IS REPOSITIONED ON THE NEXT FREE ATTRIBUTE
.*IN THE CURRENT ATTRIBUTE BLOCK WHOSE ADDRESS WAS SAVED WHEN THE NATURE
.*WAS IDENTIFIED AS SPECIAL (NATRE AND TSNAT MACROS).
.*       ********************************************************
.*INITIALISE CELL TO EMPTY AND UPDATE &RMVAR
.DEFS6   DC    X'FFFFFFF'
*ALL FLAGS 0 IN EMPTY CELL
&RMVAR   SETA  &RMVAR-1
         AGO   .DEFS5
.*       ********************************************************
.*NATURE OF ATTRIBUTE IS OCCDE WITH A LIST OF VALUES ASSOCIATED.
.*LOOP ITERATES OVER THE LIST.THE MACRO LISTVAL COMPLETES THE
.*CORRESPONDING CELL IN THE 3-GRAPH.THE END OF THE LIST IS GIVEN BY
.*THE VALUE OF &COUNT .
.DEFS4   ANOP
&CNT     SETA  3
.TSBOUC  LISTVAL &SYSLIST(&CNT)
&CNT     SETA  &CNT+1
         AIF   (&CNT LE &COUNT).TSBOUC
         AGO   .DEFS5
.*       ********************************************************
.*ERROR MESSAGE
.ERR     MNOTE 'FIRST DEFS IN A GROUP MUST HAVE A NAME'
         MEND
```

```
BEGIN     01330500:01333700 12/20/74 12/20/74

          MACRO
          BEGIN &SECTN,&PASS
.*GENERAL DESCRIPTION
.*        ***********************************************************
.*THIS MACRO HEADS ANY DEFINITION OF A 3-GRAPH  OR PART OF A 3-GRAPH
.*WHICH IS AUTONOMOUS IN THE SENSE THAT IT CONSTITUTES ONE SEPARATE
.*ASSEMBLY.THE PARAMETER &PASS  IS A DECIMAL NUMBER N WHICH INDICATES
.*THAT THIS IS THE NTH SEPARATE ASSEMBLY OF THE 3-GRAPH.THE PARAMETER
.*&SECTN ALLOWS THE USER TO NAME THIS PART OF A 3-GRAPH(AN ALPHANUMERIC
.*NAME WITH A MAX OF 127 CHARACTERS).
.*        ***********************************************************
.*MACRO VARIABLE
          GBLA  &PASSNO
&PASSNO   SETA  &PASS
.*&PASSNO IS SET EQUAL TO &PASS FOR USE IN OTHER MACROS
.*        ***********************************************************
TRGR&PASS CSECT PAGE
          USING *,12
.*THE OUTPUT OF AN ASSEMBLY MUST CONSIST OF AT LEAST ONE CSECT AND EACH
.*CSECT MUST HAVE AT LEAST ONE BASE REGISTER.THE CSECT IS GIVEN THE
.*ATTRIBUTE PAGE,IT WILL BE ALIGNED ON A PAGE BOUNDARY ADDRESS IN
.*MEMORY(MULTIPLE OF 4K).
.*        ***********************************************************
          AIF   (&PASS EQ 1).END
          EXTRN OCCDE,SUC1,SUC2,ARG1,ARG2,ARG3,ARG4
.*THE POLICY OF SEPARATELY ASSEMBLED 3-GRAPH SECTIONS MEANS THAT
.*NOTIONS DEFINED IN OTHER SECTIONS AND REFERRED TO IN THE PRESENT ONE
.*MUST BE DECLARED AS EXTRN.THIS APPLIES IN PARTICULAR TO THE SPECIAL
.*NOTIONS-DEFINED IN THE
.*1ST SEPARATE ASSEMBLY(SEE MACRO INIT) AND AUTOMATICALLY DECLARED AS
.*EXTRN IN ALL OTHERS.
.*        ***********************************************************
.END      MEND
```

```
INIT     01440C00:01442800 12/20/74 12/20/74

         MACRO
         INIT
.*GENERAL DESCRIPTION
.*       **********************************************************
.*EXPANSION OF THIS MACRO AND THE MACRO IT CALLS IS DESIGNED TO DEFINE
.*THE SPECIAL NOTIONS.THESE NOTIONS
.*WILL BE DEFINED AT THE VERY BEGINNING OF THE WHOLE 3-GRAPH AND
.*THEREFORE THE SOLE CALL OF THIS MACRO IS TO BE INSERTED AFTER THE
.*CALL  TO THE BEGIN MACRO(WITH &PASS=1) IN THE 1ST 3-GRAPH SECTION.
.*THE MACRO OUTIT IS CALLED ONCE FOR EACH NOTION TO BE DEFINED,THEN THE
.*COUNT (&NOBY) OF THE NUMBER OF BYTES USED IN THE CURRENT PAGE IS
.*UPDATED.
.*       **********************************************************
         GBLA  &NOBY
         OUTIT OCCDE
         OUTIT SUC1
         OUTIT SUC2
         OUTIT ARG1
         OUTIT ARG2
         OUTIT ARG3
         OUTIT ARG4
&NOBY    SETA  420
         MEND


OUTIT    01591600:01594000 01/07/75 01/07/75

         MACRO
         OUTIT &A1
.*GENERAL DESCRIPTION
.*       **********************************************************
.*EXPANSION OF THIS MACRO DEFINES A NEW NOTION WITH THE NAME SPECIFIED
.*BY THE PARAMETER &A1.
.*       **********************************************************
.*STATISTICS CELL,INITIALISED TO 0, WITH FLAG BIT FOR NAMED NOTION
&A1      DC    XL4'80000000'
.*OCCDE,SUC1,SUC2 CELLS,INITIALISED TO EMPTY.
         DC    3X'FFFFFFE'
.L2      ANOP
.*POINTER CELL,CHAIN TO NEXT (CONTIGUOUS) BLOCK.
         FLAG  NOT
         DC    AL3(*+3)
.*ARG1,ARG2,ARG3,ARG4 CELLS,INITIALISED TO EMPTY.
         DC    4X'FFFFFFE'
.*POINTER CELL,INITIALISED TO 'END OF NOTION'.
         DC    X'FFFFFFF'
.*NAME OF NOTION
         DC    CL8'&A1'
         DC    A(&A1)
         DS    CL8
.*NAME OF NOTION DECLARED AS ENTRY POINT,SO THAT IT MAY BE REFERRED TO
.*IN OTHER ASSEMBLIES.
         ENTRY &A1
         MEND
```

```
TGEND     01594100:01600000 01/07/75 01/07/75

          MACRO
          TGEND
.*GENERAL DESCRIPTION
.*        ***********************************************************
.*THIS MACRO INDICATES THE END OF A 3-GRAPH SECTION WHICH IS TO BE
.*ASSEMBLED SEPARATELY.THE FIRST UNUSED BYTE  AT THE END OF THE 3-GRAPH
.*IS GIVEN A UNIQUE NAME
.*THE REST OF THE CURRENT PAGE IS INITIALISED INTO A CHAINED LIST OF
.*EMPTY BLOCKS,AND THE ADDRESS OF THE FIRST BLOCK IS STORED IN ONE OF
.*THE UNUSED CELLS AT THE END OF THE PAGE.
.*        ***********************************************************
.*MACRO VARIABLES
          GBLA  &NOBY
          GBLA  &PASSNO
.*&NOBY IS A COUNT OF THE NO. OF BYTES USED IN THE CURRENT PAGE.
.*&PASSNO IDENTIFIES THE SECTION OF THE 3-GRAPH BEING TERMINATED.IT IS
.*USED TO GENERATE UNIQUE NAMES.
.*        ***********************************************************
          ORG
.*SET LOCATION COUNTER TO END OF 3-GRAPH.
.*        ***********************************************************
@TGEND&PASSNO EQU *
.*UNIQUE NAME FOR END OF 3-GRAPH SECTION.
.*        ***********************************************************
          AIF   (&NOBY NE 4080).TGEND2
          DC    X'FFFFFFFE'
          AGO   .TGEND3
.*IF THE END OF THE 3-GRAPH SECTION COINCIDES WITH THE END OF THE CURRENT
.*PAGE THE CELL WHICH SHOULD CONTAIN THE ADDRESS OF THE FIRST FREE BLOCK
.*IN THE PAGE IS INITIALISED TO EMPTY (&NOBY=4080) + FLAGS FOR END OF
.*SECTION.
.*        ***********************************************************
.TGEND2   DC    4X'FFFFFFFE'
.*THE CELLS OF A 20-BYTE FREE BLOCK ARE INITIALISED
.*        ***********************************************************
          AIF   ((&NOBY+20) EQ 4080).TGEND1
          FLAG  NOT
          DC    AL3(*+3)
.*IF THE CURRENT FREE BLOCK IS NOT THE LAST IN THE PAGE (&NOBY+20 <
.*4080) IT IS CHAINED TO THE FOLLOWING ONE
.*        ***********************************************************
&NOBY     SETA  &NOBY+20
          AGO   .TGEND2
.*UPDATE &NOBY AND LOOP
.*        ***********************************************************
.TGEND1   DC    X'FFFFFFFF'
.*END OF CHAIN OF FREE BLOCKS INDICATED IN POINTER CELL OF LAST ONE
.*        ***********************************************************
          DC    A(@TGEND&PASSNO)
.*ADDRESS OF 1ST FREE BLOCK STORED AT END OF PAGE
```

```
TGEND     01594100:01609000 01/07/75 01/07/75

.*        ********************************************************************
.TGEND3   DC      3X'FFFFFFE'
.*LAST UNUSED CELLS IN PAGE INITIALISED
.*        ********************************************************************
          ENTRY @TGEND&PASSNO
.*DECLARE GENERATED NAMES SO THAT THEY MAY BE REFERRED TO IN THE
.*SIMULATOR.
.*        ********************************************************************
          END   TRGF&PASSNO
          MEND



EXT       01403000:01405200 12/20/74 12/20/74

          MACRO
          EXT   &SECT,&PAROLS
.*GENERAL DESCRIPTION
.*        ********************************************************************
.*THIS MACRO ALLOWS THE USER TO DECLARE THE NOTIONS DEFINED IN OTHER
.*SEPARATELY ASSEMBLED 3-GRAPH SECTIONS TO WHICH HE WILL REFER IN THE
.*PRESENT ONE.THE PARAMETER &PAROLS  IS REPLACED,IN THE MACRO CALL,BY A
.*LIST OF THE EXTERNAL NOTIONS SEPARATED BY COMMAS.EXPANSION OF THIS
.*MACRO ITERATES OVER THE LIST DECLARING EACH NOTION AS EXTRN.
.*&SECT IS THE NO OF THE SECTION CONTAINING THE NOTIONS
.*        ********************************************************************
.*MACRO VARIABLE
          LCLA  &CNT
.*&CNT IS USED TO INDICATE THE CURRENT PARAMETER IN THE LIST. &SYSLIST
.*(N) IDENTIFIES THE NTH PARAMETER OF THE MACRO CALL.&SYSLIST IS A
.*SYSTEM MACRO VARIABLE.
.*        ********************************************************************
&CNT      SETA  1
.EXT2     ANOP
&CNT      SETA  &CNT+1
          AIF   ('&SYSLIST(&CNT)' EQ '').EXT1
          EXTRN &SYSLIST(&CNT)&SECT
          AGO   .EXT2
.EXT1     MEND
```

```
TESDF     01513000:01520800 12/20/74 12/20/74

          MACRO
          TESDF &NOT,&M
.*GENERAL DESCRIPTION
.*        **********************************************
.*THIS MACRO IS CALLED BY THE COMP,GREN,GPER,IFOU,LAT,SUP,SUPN,SUPP
.*MACROS ONCE FOR EACH OF THE POSSIBLE ARGUMENTS IN THE OUTER MACRO CALL
.*(IE THE PARAMETERS &ARG1,&ARG2,&ARG3,&ARG4,&SUC1,&SUC2 WHICH MAY HAVE
.*VALUES IN THE PARTICULAR CALL).THE PARAMETER &NOT IS REPLACED BY THE
.*ACTUAL ARGUMENT CONCERNED WHICH CORRESPONDS TO THE VALUE OF ONE OF
.*THE SPECIAL NOTIONS OF THE 3-GRAPH (ARG1,ARG2,ARG3,ARG4,SUC1,SUC2).
.*THE VALUE OF THE 2ND PARAMETER (&M) IS USED TO GENERATE A MACRO
.*SEQUENCE SYMBOL TO WHICH CONTROL TRANSFERS IF THE ACTUAL PARAMETER
.*TURNS OUT TO BE A NULL STRING (IE ARGUMENT MISSING OR SUCCESSOR NOT
.*EXPLICIT).
.*        **********************************************
.*MACRO VARIABLES USED
          GBLB  &STAR,&SPVAL
          GBLA  &PASSNO
          GSEQ  LF(2),LM(4)
.*&STAR SET BY THE TSTAR MACRO INDICATES ACTUAL PARAMETER = '*' OR 'NIL'.
.*&SPVAL SET BY TSVAL MACRO INDICATES ACTUAL PARAMETER = ONE OF
.*SPECIAL NOTIONS.
.*&PASSNO IS THE NO OF THE CURRENT 3-GRAPH SECTION
.*GSEQ LISTS THE SEQUENCE SYMBOLS TO BE GENERATED USING THE &M
.*PARAMETER.
.*        **********************************************
.*TESTS MADE ON VALUE OF &NOT
.*THE FIRST TEST IS FOR A NULL STRING WHICH CAUSES AN ERROR MESSAGE IF
.*THE MISSING ACTUAL PARAMETER CORRESPONDS TO ONE OF ARG1,ARG2,ARG3 OR
.*ARG4.IF IT CORRESPONDS TO SUC1 OR SUC2 THEN THE SFLAG MACRO WILL SET
.*A FLAG TO INDICATE THAT THE SUCCESSOR TO THE NOTION DEFINED BY THE
.*CALLING MACRO IS THE NEXT ACTION IN SEQUENCE (SEE SFLAG MACRO).
          AIF   ('&NOT' EQ '').L&M
.*        **********************************************
.*THE NEXT TEST IS FOR '*' OR 'NIL' AS ACTUAL PARAMETER,THE TSTAR MACRO
.*DEALS WITH THIS CASE.
          TSTAR &NOT
          AIF   (&STAR EQ 1).L2
.*        **********************************************
.*NEXT,THE ACTUAL PARAMETER IS TESTED TO SEE IF IT CONSISTS OF ONE OR
.*A LIST OF NOTIONS (ENCLOSED IN PARENTHESES AND SEPARATED BY COMMAS).
.*A LIST OF NOTIONS IS DEALT WITH BY THE SUBLST MACRO SETTING UP A
.*LIST OF VALUES CHAINED TO THE ORIGINAL VALUE CELL AS DESCRIBED IN THE
.*GENERAL DESCRIPTION OF A 3-GRAPH.
          AIF   (&NOT(2) NE '').L1
.*        **********************************************
.*FINALLY,HAVING ESTABLISHED THAT THE ACTUAL PARAMETER IS A SINGLE
.*NOTION THIS IS TESTED TO SEE IF IT IS ONE OF THE SPECIAL 3-GRAPH
.*NOTIONS (TSVAL MACRO) AND THE CORRESPONDING VALUE CELL IS FILLED IN
.*WITH THE FLAG BITS AND THE ADDRESS OF THE NOTION.
```

```
TESDF     01513000:01520800 12/20/74 12/20/74

          TSVAL &NOT
          AIF   (&SEVAL EQ 1).L2
          FLAG  NOT
.*IF NOTION INTERNAL CONCATENATE NAME WITH CURRENT SECTION #
          AIF   (T'&NOT NE 'M').EXTRN
          DC    AL3(&NOT&PASSNO)
          MEXIT
.EXTRN    DC    AL3(&NOT)
.L2       MEXIT
.*        ********************************************
.*PARAMETER IS A SUBLST
.L1       SUBLST &NOT
          MEXIT
.*        ********************************************
.*SUC1 NOT EXPLICIT
.LF1      SFLAG 1
          MEXIT
.*        ********************************************
.*SUC2 NOT EXPLICIT
.LF2      SFLAG 2
          MEXIT
.*        ********************************************
.*ERROR MESSAGES
.LM1      MNOTE 'FIRST ARGUMENT MISSING'
          MEXIT
.LM2      MNOTE 'SECOND ARGUMENT MISSING'
          MEXIT
.LM3      MNOTE 'THIRD ARGUMENT MISSING'
          MEXIT
.LM4      MNOTE 'FOURTH ARGUMENT MISSING'
          MEND
```

```
SUBLST   01480400:01490100 12/20/74 12/20/74

         MACRO
         SUBLST &PAR
.*GENERAL DESCRIPTION
.*       ********************************************
.*THIS MACRO IS CALLED BY THE TESOF MACRO WHEN AN ACTUAL PARAMETER
.*CORRESPONDING TO ONE OF THE SPECIAL NOTIONS TAKES THE FORM OF A LIST .
.*THIS LIST (&PAR) CONSISTS OF A NUMBER OF NOTION NAMES SEPARATED BY
.*COMMAS AND ENCLOSED IN PARENTHESES (MAX OF 127 CHARACTERS).
.*THIS MACRO CREATES A VALUE LIST IN THE 3-GRAPH
.*CHAINED TO THE CORRESPONDING VALUE CELL IN THE NOTION BEING DEFINED.
.*       ********************************************
.*MACRO VARIABLES
         GBLA  &RMVAR,&POS
         GBLA  &PASSNO
         GBLB  &SPVAL
         LCLA  &COUNT,&CNT
.*&RMVAR IS AN INDICATOR OF THE NO OF CELLS LEFT IN A VALUE LIST BLOCK
.*&POS IS USED TO CREATE UNIQUE SYMBOLS FOR LABELLING LOCATION
.*COUNTER POSITIONS.
.*&PASSNO IS THE NO OF THE CURRENT 3-GRAPH SECTION
.*&SPVAL (SEE TSVAL MACRO) IS A BOOLEAN INDICATOR USED IN CONNECTION
.*WITH THE SPECIAL NOTIONS-OCCDE,SUC1,SUC2,ARG1,ARG2,ARG3,ARG4.
.*&COUNT,&CNT ARE USED FOR IDENTIFYING AND KEEPING TRACK OF THE NOTIONS
.*IN THE LIST.
.*       ********************************************
.*INITIALISATION OF VARIABLES
&CNT     SETA  1
&POS     SETA  &POS+1
.*       ********************************************
.*DETERMINE THE NO. OF NOTIONS IN THE LIST BY TESTING EACH ACTUAL
.*PARAMETER IN THE LIST UNTIL A NULL STRING IS ENCOUNTERED AND
.*INCREMENTING A COUNTER (&CNT)
.SUB2    AIF   ('&PAR(&CNT)' EQ '').SUB1
&CNT     SETA  &CNT+1
         AGO   .SUB2
.SUB1    ANOP
.*       ********************************************
.*SAVE THE CURRENT LOCATION POSITION (IE LABEL IT WITH A UNIQUE
.*GENERATED SYMBOL)
&NA&POS  EQU   *
.*       ********************************************
.*ALLOCATE A NEW 20-BYTE BLOCK AND CHAIN IT TO THE CURRENT CELL,
.*POSITION THE LOCATION COUNTER AT THE START OF THE NEW BLOCK.
         ALLOC VAL,20
.*       ********************************************
.*SET &RMVAR TO INDICATE 3 CELLS LEFT,FILL IN COUNTER CELL.
&RMVAR   SETA  3
         SETCNT &CNT,1
&CNT     SETA  &CNT-1
.*       ********************************************
```

```
SUBLST   01480400:01490100 12/20/74 12/20/74

.*INITIALISE A NEW COUNTER
&COUNT   SETA  1
.*       ********************************************************
.*TEST IF THERE IS ROOM IN THE CURRENT BLOCK FOR THE NEXT NOTION IN
.*THE LIST.IF SO GO TO .SKIP.
.SUB4    AIF   (&RMVAR NE 0).SKIP
.*       ********************************************************
.*OTHERWISE ALLOCATE A NEW 20-BYTE BLOCK,CHAIN IT TO THE LAST ONE
.*AND POSITION THE LOCATION COUNTER TO THE START OF THE NEW BLOCK.
         ALLOC SUB1,20
.*       ********************************************************
.*SET &RMVAR TO INDICATE 4 CELLS LEFT IN BLOCK.
&RMVAR   SETA  4
.*       ********************************************************
.*TEST NEXT NOTION IN LIST (INDEXED BY &COUNT) TO SEE IF IT IS ONE OF
.*THE SPECIAL NOTIONS.(SEE TSVAL MACRO) IF SO GO TO .SUB3
.SKIP    TSVAL  &PAR(&COUNT)
         AIF   (&SPVAL LO 1).SUB3
.*       ********************************************************
.*OTHERWISE COMPLETE CELL WITH ADDRESS OF NOTION AND FLAG BITS.
         FLAG  NOT
.*IF NOTION INTERNAL CONCATENATE NAME WITH CURRENT SECTION #
         AIF   (T'&PAR(&COUNT) NE 'N').EXTRN
         DC    AL3(&PAR(&COUNT)&PASSNO)
         AGO   .SUB3
.EXTRN   DC    AL3(&PAR(&COUNT))
.SUB3    ANOP
.*       ********************************************************
.*UPDATE &RMVAR & &COUNT
&RMVAR   SETA  &RMVAR-1
&COUNT   SETA  &COUNT+1
.*       ********************************************************
.*TEST FOR END OF LIST,IF NOT END GO TO .SUB4
         AIF   (&COUNT LE &CNT).SUB4
.*       ********************************************************
.*OTHERWISE COMPLETE CURRENT VALUE LIST BLOCK BY INITIALISING ANY
.*LEFT-OVER CELLS TO EMPTY AND THE POINTER CELL TO 'END OF LIST'
.SUB6    AIF   (&RMVAR NE 0).SUB5
         DC    X'FFFFFFF'
*NO FLAGS IN EMPTY CELL
.*       ********************************************************
.*RESET LOCATION COUNTER TO NEXT CELL IN ORIGINAL NOTION.
         ORG   @NA&POS+4
         MEXIT
.*       ********************************************************
.*INITIALISE CELL TO EMPTY AND UPDATE &RMVAR
.SUB5    DC    X'FFFFFFE'
&RMVAR   SETA  &RMVAR-1
         AGO   .SUB6
         MEND
```

```
TNAME     01541600:01545200 12/20/74 12/20/74

          MACRO
          TNAME &NAM
.*GENERAL DESCRIPTION
.*        **********************************
.*THIS MACRO AND THE MACROS IT CALLS DEFINE THE NEW NOTION (IF IT
.*IS NAMED) AND ALLOCATE SPACE FOR IT.IF THE LAST ACTION DEFINED DID
.*NOT HAVE AN EXPLICIT SUCCESSOR (SUC1 OR SUC2) THE ADDRESS OF
.*THE NEW NOTION IS STORED AS THE SUCCESSOR ACTION WHICH IS REQUIRED.
.*        **********************************
.*THIS MACRO IS CALLED BY THE COMP,CREN,CRER,IFOU,LAT,RLF,SUP,SUFN,
.*SUPR,TMV MACROS.ITS ONLY PARAMETER IS THE NAME GIVEN (COL 1-8)
.* TO THE NEW NOTION OTHERWISE THE PARAMETER IS NULL.
.*        **********************************
          GBLA  &PASSNO
.*&PASSNO IS THE NO OF THE CURRENT 3-GRAPH SECTION
          AIF   ('&NAM' EQ '').L1
.*IF THE NEW NOTION HAS A NAME THE DEFN MACRO IS CALLED TO DEFINE IT
.*AND ALLOCATE SPACE,THE LOCATION COUNTER IS SET TO THE START OF THE
.*NEW NOTION.
          DEFN  &NAM&PASSNO
.*        **********************************
.*THE TFLAG MACRO IS CALLED TWICE TO SEE IF THE LAST ACTION DEFINED
.*REQUIRES A SUC1 AND/OR SUC2,IF SO,THE APPROPRIATE CELL IS
.*COMPLETED WITH THE ADDRESS OF THE NEW NOTION AND THE FLAG IS
.*RESET.
.L3       TFLAG 1
          TFLAG 2
          MEXIT
.*        **********************************
.*WHEN THE NEW NOTION IS UNNAMED SPACE IS ALLOCATED FOR IT AND THE
.*LOCATION COUNTER IS SET TO THE BEGINNING OF THE NEW NOTION.
.L1       ORG
          TROOM 40
          DS    CL20
          DS    CL20
          ORG   *-40
          AGO   .L3
          MEND
```

```
NATRE    01454300:01460500 12/20/74 12/20/74

         MACRO
&NAME    NATRE &NAT
.*GENERAL DESCRIPTION
.*       ***********************************************************
.*CALLED BY DEF AND DEFS MACROS,EXPANSION OF THE NATRE MACRO FILLS IN
.*THE NATURE CELL OF A NEW ATTRIBUTE,ALLOWING FOR SPECIAL CASES.
.*       ***********************************************************
.*MACRO VARIABLES
         GBLB  &SPNAT,&DEFCON
         GBLA  &PASSNO
         GBLA  &RMATB,&NAMGEN
.*&SPNAT IS SET BY THE TSNAT MACRO TO 1,IF THE NATURE OF THE ATTRIBUTE
.*IS ONE OF THE SPECIAL NOTIONS IN THE 3-GRAPH,TO 0 OTHERWISE.
.*&DEFCON IS SET TO INDICATE THAT THE CALLING MACRO (DEF OR DEFS) IS
.*PART OF A DEF OR DEFS GROUP.
.*&RMATB INDICATES THE NO OF ATTRIBUTES WHICH CAN STILL BE STORED IN
.*THE CURRENT BLOCK (EG 2,1,0).IT IS SET AFTER A NEW TAIL BLOCK HAS
.*BEEN ADDED,AND UPDATED AFTER AN ATTRIBUTE HAS BEEN ADDED TO THE
.*NOTION.
.*&NAMGEN IS USED FOR GENERATING UNIQUE SYMBOLS
.*&PASSNO IS THE NO OF THE CURRENT 3-GRAPH SECTION
.*       ***********************************************************
.*THE FIRST THING IS TO TEST FOR A SPECIAL
.*NOTION.THIS IS DONE BY THE TSNAT MACRO WHICH THEN POSITIONS THE
.*LOCATION COUNTER ON THE CORRECT CELL IN THE ROOT NOTION. FOR
.*THIS,IT REQUIRES A SYMBOLIC ADDRESS FOR THE START OF THE ROOT NOTION
.*WHICH IS THE NAME OF THE NOTION WHEN THE CALLING DEF OR DEFS MACRO IS
.*THE FIRST IN A GROUP OF SUCH MACRO CALLS,AND IS GIVEN BY A GENERATED
.*SYMBOL IN THE OTHERS. SO THE &NAME PARAMETER IS TESTED.IF NOT NULL
.*THE NAME OF THE NOTION CAN BE USED,OTHERWISE THE GENERATED SYMBOL IS
.*USED (AT .NAT1)
         AIF   ('&NAME' LQ '').NAT1
.*       ***********************************************************
.*TEST FOR SPECIAL NOTION
         TSNAT &NAT,&NAME
.*       ***********************************************************
.*IF THE TSNAT TEST  IS POSITIVE (&SPNAT=1) THE NATRE MACRO HAS NOTHING
.*TO DO.(NOTE:LOCATION COUNTER POSITIONED BY TSNAT MACRO)
.NAT2    AIF   (&SPNAT EQ 1).NAT3
.*       ***********************************************************
.*IF CALLING DEF OR DEFS MACRO IS THE FIRST IN A GROUP AND NATURE OF
.*ATTRIBUTE NOT SPECIAL,POSITION ON POINTER CELL AT END OF 1ST 2 BLOCKS
.*OF NOTION
         AIF   (&DEFCON NL 0).NAT4
         ORG   *+36
.*       ***********************************************************
.*IF THE CURRENT ATTRIBUTE BLOCK IS FULL (OR NO ATTRIBUTES YET ADDED)
.*ALLOCATE A NEW TAIL BLOCK AND CHAIN TO END OF OLD ONE.RESET &RMATB
.NAT4    AIF   (&RMATB NE 0).SKIP
         ALLOC SUI,20
```

```
NATRE     01454300:01460500 12/20/74 12/20/74

&RNATB    SETA  2
.*        ********************************************************
.*FILL IN NATURE CELL (LOCATION COUNTER NOW POSITIONED ON VALUE CELL)
.SKIP     FLAG  NOT
.*IF NOTION INTERNAL CONCATENATE NAME WITH CURRENT SECTION #
          AIF   (T'&NAT NE 'M').EXTRN
          DC    AL3(&NAT&PASSNO)
          MEXIT
.EXTRN    DC    AL3(&NAT)
.NAT3     MEXIT
.*        ********************************************************
.*TEST FOR SPECIAL NOTION
.NAT1     TSNAT &NAT,@NO&NAMGEN
          AGO   .NAT2
          MEND



DEFN      01538900:01591500 01/07/75 01/07/75

          MACRO
          DEFN  &PAR1
.*GENERAL DESCRIPTION
.*        ********************************************************
.*THIS MACRO IS CALLED BY THE TNAME MACRO TO DEFINE THE NEW NOTION
.*WHICH HAS BEEN GIVEN A NAME.THE VALUE OF &PAR1 IS THE NAME OF THE NEW
.*NOTION.
.*        ********************************************************
.*SET LOCATION COUNTER TO 1ST AVAILABLE SPACE (IE HIGHEST VALUE YET
.*ENCOUNTERED)
          ORG
.*        ********************************************************
.*TEST IF THERE IS ROOM LEFT IN THE CURRENT PAGE FOR A NEW NOTION
          TROOM 60
.*        ********************************************************
.*DEFINE THE NEW NOTION AND RESET LOCATION COUNTER TO THE BEGINNING OF
.*IT.
&PAR1     DS    CL40
.*NAME OF NOTION
          DC    CL8'&PAR1'
          DC    A(&PAR1)
          DS    CL8
          ORG   &PAR1
.*        ********************************************************
.*DEFINE THE NAME GIVEN AS AN ENTRY POINT SO THAT IT MAY BE REFERRED TO
.*IN SUBSEQUENT 3-GRAPH SECTIONS.
          ENTRY &PAR1
          MEND
```

```
DEFNOT   01585300:01588600 01/07/75 01/07/75

         MACRO
         DEFNOT &NOTNAME
.*GENERAL DESCRIPTION
.*       ************************************************************
.*CALLED BY THE DEF MACRO,THE EXPANSION OF THE DEFNOT MACRO DEFINES A
.*NEW NOTION IN THE 3-GRAPH WITH A NAME. THE
.*NOTION IS DEFINED WITHOUT ATTRIBUTES (ALL CELLS INITIALISED TO
.*'EMPTY'.
.*       ************************************************************
.*POSITION LOCATION COUNTER ON FIRST FREE ADDRESS IN 3-GRAPH (IE
.*HIGHEST ADDRESS SO FAR).
         ORG
.*       ************************************************************
.*TEST IF ENOUGH ROOM IN CURRENT PAGE.
         TROOM 60
.*       ************************************************************
.*DEFINE NOTION
.*STATISTICS CELL,INITIALISED TO 0 FLAG BIT FOR NAMED NOTION
&NOTNAME DC    XL4'80000000'
.*OCCDE,SUC1,SUC2 CELLS INITIALISED TO 'EMPTY'
         DC    3X'FFFFFFE'
.*POINTER CELL,CHAIN TO NEXT (CONTIGUOUS) BLOCK
         FLAG  NOT
         DC    AL3(*+3)
.*ARG1,ARG2,ARG3,ARG4 CELLS,INITIALISED TO 'EMPTY'
         DC    4X'FFFFFFE'
.*POINTER CELL,INITIALISED TO 'END OF NOTION'.
         DC    X'FFFFFFF'
.*NAME OF NOTION
         DC    CL8'&NOTNAME'
         DC    A(&NOTNAME)
         DS    CL8
.*       ************************************************************
.*DEFINE NAME OF NOTION AS ENTRY POINT (PERMITS REFERENCES TO THIS
.*NOTION IN SUBSEQUENT 3-GRAPH SECTIONS)
         ENTRY &NOTNAME
         MEND
```

```
VALUE     01577600:01585200 12/20/74 12/20/74

          MACRO
&NAME     VALUE &VAL
.*GENERAL DESCRIPTION.
.*        ************************************************************
.*CALLED BY THE DEF AND DEFS MACROS,EXPANSION OF THE VALUE MACRO DEALS
.*WITH THE DIFFERENT CASES FOR THE VALUE OF THE ATTRIBUTE (DEPENDENT
.*ON THE NATURE).
.*        ************************************************************
.*MACRO VARIABLES
          GBLB  &DEFCON,&SPOCCDE,&SPNAT,&BASACT,&SPVAL
          GBLA  &PASSNO
          GBLA  &RMATB,&NAMGEN
.*&DEFCON INDICATES WHETHER THE CALLING DEF OR DEFS MACRO IS THE FIRST
.*IN A GROUP (&DEFCON=0) OR NOT (&DEFCON=1).
.*&SPOCCDE INDICATES WHETHER THE NATURE OF THE ATTRIBUTE IS ONE OF THE
.*'SPECIAL' NOTIONS (SET BY TSNAT MACRO).
.*&BASACT (SET BY TSOCC MACRO) INDICATES WHETHER THE VALUE OF THE
.*ATTRIBUTE (IN THE CASE WHERE THE NATURE IS 'OCCDE') IS ONE OF THE
.*BASIC ACTIONS OF THE 3-GRAPH.
.*&SPVAL (SET BY TSVAL MACRO) INDICATES WHETHER THE VALUE OF THE
.*ATTRIBUTE IS ONE OF THE 'SPECIAL' NOTIONS.
.*&RMATB SPECIFIES THE NO OF ATTRIBUTE SPACES LEFT IN AN ATTRIBUTE
.*BLOCK.
.*&NAMGEN IS USED FOR GENERATING UNIQUE SYMBOLS
.*&PASSNO IS THE NO OF THE CURRENT 3-GRAPH SECTION
.*        ************************************************************
.*TEST &SPOCCDE.IF =1 NATURE OF ATTRIBUTE IS 'OCCDE' SO TSOCC MACRO
.*IS CALLED TO TEST VALUE OF ATTRIBUTE.IF THIS IS ONE OF BASIC ACTIONS
.*OF 3-GRAPH,THE CELL IS COMPLETED AND &BASACT IS SET TO 1.
          AIF   (&SPOCCDE NE 1).VAL1
          TSOCC &VAL
          AIF   (&BASACT NE 1).VAL2
.*        ************************************************************
.*TEST &SPNAT.IF =1 THEN THE LOCATION COUNTER IS TO BE REPOSITIONED
          AIF   (&SPNAT EQ 1).RESET
.*        ************************************************************
.*UPDATE &RMATB AND SET &DEFCON TO 1 TO INDICATE THAT AT LEAST ONE OF A
.*GROUP OF DEF OR DEFS MACRO CALLS HAS BEEN ENCOUNTERED.
&RMATB    SETA  &RMATB-1
&DEFCON   SETB  1
          MEXIT
.*        ************************************************************
.*VALUE OF ATTRIBUTE IS AN ORDINARY NOTION OF THE 3-GRAPH SO COMPLETE
.*CELL IN USUAL WAY.
.VAL2     FLAG  NOT
.*IF NOTION INTERNAL CONCATENATE NAME WITH CURRENT SECTION #
          AIF   (T'&VAL NE 'N').EXTRN
          DC    AL3(&VAL&PASSNO)
          AGO   .NEXT
.EXTRN    DC    AL3(&VAL)
```

```
VALUE    01577600:01585200 12/20/74 12/20/74

.*       ********************************************************************
.*IF &SPNAT=1 LOCATION COUNTER IS TO BE RESET
.NEXT    AIF   (&SPNAT EQ 1).RESET
.*       ********************************************************************
.*UPDATE &RMATB AND SET &DEFCON.
&RMATB   SETA  &RMATB-1
&DEFCON  SETB  1
         MEXIT
.*       ********************************************************************
.*NATURE OF ATTRIBUTE NOT 'OCCDE'.TSVAL MACRO IS CALLED TO TEST VALUE
.*OF ATTRIBUTE.IF IT CORRESPONDS TO ONE OF THE 'SPECIAL' NOTIONS,THE
.*CELL IS FILLED IN AND &SPVAL IS SET TO 1.
.VAL1    TSVAL &VAL
.*       ********************************************************************
.*IF VALUE NOT 'SPECIAL' GO TO .VAL2
         AIF   (&SPVAL NE 1).VAL2
.*       ********************************************************************
.*IF &SPNAT=1 LOCATION COUNTER IS TO BE RESET
         AIF   (&SPNAT EQ 1).RESET
.*       ********************************************************************
.*UPDATE &RMATB AND SET &DEFCON.
&RMATB   SETA  &RMATB-1
&DEFCON  SETB  1
         MEXIT
.*       ********************************************************************
.*RESET LOCATION COUNTER TO START OF NOTION AND SET &DEFCON
.RESET   ORG   @NO&NAMGEN
&DEFCON  SETB  1
         MEND
```

```
LISTVAL  0145040):01454200 12/20/74 12/20/74

         MACRO
         LISTVAL &VAL
.*GENERAL DESCRIPTION
.*       ********************************************************
.*CALLED BY THE DEF AND DEFS MACROS,EXPANSION OF THE LISTVAL MACRO
.*DEALS WITH ONE OF A LIST OF VALUES OF AN ATTRIBUTE WHEN THE NATURE
.*IS 'OCCDE'.IT IS CALLED ONCE FOR EACH OF THE VALUES IN THE LIST.
.*       ********************************************************
.*MACRO VARIABLES
         GBLA  &PASSNO
         GBLA  &RMVAR
         GBLB  &BASACT
.*&RMVAR INDICATES THE NO OF EMPTY CELLS LEFT ON THE CURRENT BLOCK
.*&BASACT INDICATES WHETHER THE VALUE IS A BASIC ACTION OF THE 3-GRAPH
.*&PASSNO IS THE NO OF THE CURRENT 3-GRAPH SECTION
.*       ********************************************************
.*IF THE CURRENT BLOCK IS FULL (&RMVAR=0) ALLOCATE A NEW BLOCK AND
.*RESET &RMVAR
         AIF   (&RMVAR NE 0).LST1
         ALLOC SUI,20
&RMVAR   SETA  4
.*       ********************************************************
.*TEST IF VALUE IS A BASIC ACTION.TSOCC MACRO SETS &BASACT=1 AND FILLS
.*VALUE CELL IF TEST SUCCESSFUL.
.LST1    TSOCC &VAL
         AIF   (&BASACT NE 1).LST2
.*       ********************************************************
.*IF TEST SUCCESSFUL UPDATE &RMVAR
&RMVAR   SETA  &RMVAR-1
         MEXIT
.*       ********************************************************
.*VALUE IS AN ORDINARY NOTION.FILL IN VALUE CELL AND UPDATE &RMVAR
.*THE NAME OF THE NOTION IS CONCATENATED WITH THE CURRENT SECTION NUMBER
.*IF IT IS NOT AN EXTERNAL NOTION (TYPE OF NOTION = 'M' IF INTERNAL)
.LST2    FLAG  NOT
         AIF   (T'&VAL NE 'M').EXTRN
         DC    AL3(&VAL&PASSNO)
         AGO   .NEXT
.EXTRN   DC    AL3(&VAL)
.NEXT    ANOP
&RMVAR   SETA  &RMVAR-1
         MEND
```

```
VALST    01573600:01577500 12/20/74 12/20/74

         MACRO
         VALST &VAL
.*GENERAL DESCRIPTION
.*       ********************************************************
.*CALLED BY THE DEF AND DLFS MACROS TO DEAL WITH A VALUE OF A VALUE
.*LIST EXCEPT WHEN THE NATURE OF THE ATTRIBUTE IS 'OCODE'.IT IS CALLED
.*ONCE FOR EACH VALUE IN THE LIST AND FILLS IN THE CORRESPONDING
.*VALUE CELL,ADDING NEW BLOCKS TO THE NOTION AS NECESSARY.
.*       ********************************************************
.*MACRO VARIABLES
         GBLA  &PASSNO
         GBLA  &RMVAB
         GBLB  &SPVAL
.*&RMVAB INDICATES THE NO OF EMPTY CELLS LEFT ON THE CURRENT BLOCK
.*&SPVAL INDICATES WHETHER THE VALUE IS ONE OF THE 'SPECIAL' NOTIONS
.*OF THE 3-GRAPH
.*&PASSNO IS THE CURRENT 3-GRAPH SECTION NO.
.*       ********************************************************
.*IF THE CURRENT BLOCK IS FULL (&RMVAB=0) ALLOCATE A NEW BLOCK AND
.*RESET &RMVAB
         AIF   (&RMVAB NE 0).JUMP
         ALLOC SUI,20
&RMVAB   SETA  4
.*       ********************************************************
.*TEST IF THE VALUE IS ONE OF THE 'SPECIAL' NOTIONS OF THE 3-GRAPH.
.*THE TSVAL MACRO FILLS IN THE CELL IF IT IS AND SETS &SPVAL=1.UPDATE
.*&RMVAB
.JUMP    TSVAL &VAL
         AIF   (&SPVAL NE 1).VALST1
&RMVAB   SETA  &RMVAB-1
         MEXIT
.*       ********************************************************
.*THE VALUE IS AN ORDINARY NOTION.FILL IN VALUE CELL AND UPDATE &RMVAB.
.VALST1  FLAG  NOT
.*IF NOTION INTERNAL CONCATENATE NAME WITH CURRENT SECTION #
         AIF   (T'&VAL NE 'M').EXTRN
         DC    AL3(&VAL&PASSNO)
         AGO   .NEXT
.EXTRN   DC    AL3(&VAL)
.NEXT    ANOP
&RMVAB   SETA  &RMVAB-1
         MEND
```

```
TSNAT     01549300:01558500 12/20/74 12/20/74

          MACRO
          TSNAT &NAT,&NAME
.*GENERAL DESCRIPTION
.*        ********************************************************
.*CALLED BY THE NATRE MACRO TO DEAL WITH THE NATURE
.*OF AN ATTRIBUTE BEING DEFINED BY A DEF OF DEFS MACRO CALL
.*        ********************************************************
.*PARAMETERS
.*&NAT IS REPLACED BY THE NAME OF THE NOTION WHICH IS NATURE OF THE
.*ATTRIBUTE
.*&NAME IS REPLACED BY THE NAME OF THE ROOT NOTION OR BY A GENERATED
.*SYMBOL ATTACHED TO THE BEGINNING OF THE ROOT NOTION.
.*        ********************************************************
.*VARIABLE SYMBOLS
          GBLB  &SPNAT,&SPOCCDE
          GBLA  &POS
.*&SPNAT IS SET IN THIS MACRO TO INDICATE THAT THE VALUE OF &NAT IS ONE
.*OF THE SPECIAL NOTIONS (&SPNAT = 1) OR NOT (&SPNAT = 0)
.*&SPOCCDE IS SET = 1 IF THE VALUE OF &NAT IS 'OCCDE' AND IS SET = 0
.*OTHERWISE.
.*&POS IS USED TO GENERATE UNIQUE SYMBOLS.
.*        ********************************************************
.*ERROR MESSAGES
          AIF   ('&NAME' EQ '').ERR
          AIF   ('&NAT' EQ '').ERR1
.*IF EITHER &NAT OR &NAME HAS BEEN REPLACED BY A NULL STRING AN ERROR
.*MESSAGE IS OUTPUT.
.*NOTE:THE LOCATION COUNTER VALUE IS NOT ALTERED IN THIS CASE.
.*        ********************************************************
&POS      SETA  &POS+1
.*&POS IS INCREMENTED TO ENSURE THAT GENERATED SYMBOLS WILL BE UNIQUE.
.*        ********************************************************
.*TESTS ON &NAT
.*THE VALUE OF &NAT IS TESTED AGAINST THE SPECIAL NOTION NAMES
.*IN TURN.IF A TEST IS SUCCESSFUL THE CURRENT VALUE OF THE LOCATION
.*COUNTER IS SAVED,&SPNAT IS SET TO 1 (IF 'OCCDE' IS DETECTED &SPOCCDE
.*IS ALSO SET TO 1 OTHERWISE TO 0) AND THE LOCATION COUNTER IS
.*POSITIONED AT THE VALUE CELL IN THE ROOT NOTION CORRESPONDING TO THE
.*NOTION DETECTED
.*NOTE:THIS POSITIONING RELIES ON THE FACT THAT THE
.*FIRST TWO BLOCKS OF A NOTION ARE CONTIGUOUS AND ON THE ORDER OF THE
.*CELLS CORRESPONDING TO THE SPECIAL ATTRIBUTES.
.*        ********************************************************
          AIF   ('&NAT' NE 'OCCDE').L1
&SPNAT    SETB  1
&SPOCCDE  SETB  1
@NA&SYS   EQU   *
          ORG   &NAME+4
          MEXIT
.L1       ANOP
```

```
TSNAT     01549300:01558500 12/20/74 12/20/74

&SPGCODE  SETB  0
          AIF   ('&NAT' NE 'SUC1').L2
&SPNAT    SETB  1
@NA&POS   EQU   *
          ORG   &NAME+8
          MEXIT
.L2       AIF   ('&NAT' NE 'SUC2').L3
&SPNAT    SETB  1
@NA&POS   EQU   *
          ORG   &NAME+12
          MEXIT
.L3       AIF   ('&NAT' NE 'ARG1').L4
&SPNAT    SETB  1
@NA&POS   EQU   *
          ORG   &NAME+20
          MEXIT
.L4       AIF   ('&NAT' NE 'ARG2').L5
&SPNAT    SETB  1
@NA&POS   EQU   *
          ORG   &NAME+24
          MEXIT
.L5       AIF   ('&NAT' NE 'ARG3').L6
&SPNAT    SETB  1
@NA&POS   EQU   *
          ORG   &NAME+28
          MEXIT
.L6       AIF   ('&NAT' NE 'ARG4').L7
&SPNAT    SETB  1
@NA&POS   EQU   *
          ORG   &NAME+32
          MEXIT
.*        **********************************************************
.*IF ALL TESTS ARE NEGATIVE &SPNAT IS SET TO 0.
.L7       ANOP
&SPNAT    SETB  0
          MEXIT
.*        **********************************************************
.*ERROR MESSAGES
.ERR      MNOTE 'NOTION HAS NO NAME,TSNAT MACRO'
          MEXIT
.ERR1     MNOTE 'NATURE IS NULL,TSNAT MACRO'
          MEND
```

```
TSVAL     01568700:01573500 12/20/74 12/20/74

          MACRO
          TSVAL &A1
.*GENERAL DESCRIPTION
.*        ***********************************************
.*THIS MACRO IS CALLED BY THE TESDF,SUBLST,VALUE AND VALST MACROS.IT
.*TESTS WHETHER THE ACTUAL PARAMETER CORRESPONDING TO &A1 IS ONE OF THE
.*SPECIAL NOTIONS OCCDE,SUC1,SUC2,ARG1,ARG2,ARG3 OR ARG4.IF SO THE
.*CURRENT CELL IS FILLED IN WITH THE ADDRESS OF THE SPECIAL NOTION (AND
.*CORRESPONDING FLAG BITS) AND THE GLOBAL BOOLEAN VARIABLE &SPVAL IS
.*SET TO 1.OTHERWISE &SPVAL IS SIMPLY SET TO 0.
.*        ***********************************************
          GBLB  &SPVAL
          AIF   ('&A1' NE 'OCCDE').L1
          FLAG  NOT
          DC    AL3(OCCDE)
          AGO   .SET
.L1       AIF   ('&A1' NE 'SUC1').L2
          FLAG  NOT
          DC    AL3(SUC1)
          AGO   .SET
.L2       AIF   ('&A1' NE 'SUC2').L3
          FLAG  NOT
          DC    AL3(SUC2)
          AGO   .SET
.L3       AIF   ('&A1' NE 'ARG1').L4
          FLAG  NOT
          DC    AL3(ARG1)
          AGO   .SET
.L4       AIF   ('&A1' NE 'ARG2').L5
          FLAG  NOT
          DC    AL3(ARG2)
          AGO   .SET
.L5       AIF   ('&A1' NE 'ARG3').L6
          FLAG  NOT
          DC    AL3(ARG3)
          AGO   .SET
.L6       AIF   ('&A1' NE 'ARG4').L7
          FLAG  NOT
          DC    AL3(ARG4)
.*        ***********************************************
.*TESTS SUCCESSFUL
.SET      ANOP
&SPVAL    SETB  1
          MEXIT
.*        ***********************************************
.*TESTS UNSUCCESSFUL
.L16      ANOP
&SPVAL    SETB  0
          MEND
```

```
TSOCC     01558600:01566600 12/20/74 12/20/74

          MACRO
          TSOCC &VAL
.*GENERAL DESCRIPTION
.*        ********************************************************
.*THIS MACRO IS CALLED BY THE VALUE,VALST AND LISTVAL MACROS TO TEST
.*THE VALUE OF AN ATTRIBUTE BEING DEFINED WHEN THE NATURE OF THE
.*ATTRIBUTE WAS THE SPECIAL NOTION CCCDE.
.*        ********************************************************
.*PARAMETER
.*&VAL IS REPLACED BY THE NAME OF THE VALUE OF THE ATTRIBUTE IN
.*QUESTION.
.*        ********************************************************
.*VARIABLE SYMBOL
          GBLB  &BASACT
.*&BASACT IS SET = 1 IF THE ACTUAL PARAMETER CORRESPONDS TO ONE OF THE
.*BASIC ACTIONS OF THE 3-GRAPH.
.*        ********************************************************
.*ERROR MESSAGE
          AIF   (('&VAL' EQ '') OR ('&VAL' EQ '*')).ERR
.*IF THE ACTUAL PARAMETER IS A NULL STRING OR '*' AN ERROR MESSAGE IS
.*OUTPUT.NOTE:THE CURRENT CELL IS NOT FILLED IN AND THE LOCATION
.*COUNTER DOES NOT MOVE.
.*        ********************************************************
.*TESTS ON &VAL
.*THE ACTUAL PARAMETER IS TESTED AGAINST EACH OF THE BASIC ACTION NAMES
.*IN TURN.IF A TEST IS SUCCESSFUL THE CURRENT CELL IS FILLED IN WITH
.*THE FLAG BITS + THE OFFSET IN THE TABLE OF BASIC ACTIONS
.*CORRESPONDING TO THE ACTION DETECTED,&BASACT IS SET TO 1.
.*        ********************************************************
          AIF   ('&VAL' NE 'COMP').L0
          FLAG  BASE
          DC    AL3(0)
          AGO   .L9
.L0       AIF   ('&VAL' NE 'CREN').L1
          FLAG  BASE
          DC    AL3(4)
          AGO   .L9
.L1       ANOP
.L2       AIF   ('&VAL' NE 'LAT').L3
          FLAG  BASE
          DC    AL3(12)
          AGO   .L9
.L3       AIF   ('&VAL' NE 'REP').L4
          FLAG  BASE
          DC    AL3(16)
          AGO   .L9
.L4       AIF   ('&VAL' NE 'SUP').L5
          FLAG  BASE
          DC    AL3(20)
          AGO   .L9
```

```
TSOCC    01558600:01566600 12/20/74 12/20/74

.L5      ANOP
.L6      AIF   ('&VAL' NE 'SUPR').L7
         FLAG  BASE
         DC    AL3(28)
         AGO   .L9
.L7      AIF   ('&VAL' NE 'TMV').L8
         FLAG  BASE
         DC    AL3(32)
         AGO   .L9
.L8      AIF   ('&VAL' NE 'CREF').L10
         FLAG  BASE
         DC    AL3(36)
.*       ********************************************************************
.*TEST SUCCESSFUL
.L9      ANOP
&BASACT  SETB  1
         MEXIT
.*       ********************************************************************
.*ERROR MESSAGE
.ERR     MNOTE 'ILLEGAL OPERAND,TSOCC MACRO'
.*       ********************************************************************
.*IF ALL TESTS FAIL &BASACT IS SET TO 0.
.L10     ANOP
&BASACT  SETB  0
         MEND



TSTAR    01566700:01568600 12/20/74 12/20/74

         MACRO
         TSTAR &A1
.*GENERAL DESCRIPTION
.*       ********************************************************************
.*THIS MACRO IS CALLED BY THE TESDF MACRO TO TEST WHETHER THE VALUE OF
.*THE PARAMETER &A1 (CORRESPONDING TO THE VALUE OF ONE OF THE SPECIAL
.*NOTIONS) IS '*' OR 'NIL'.IF SO THE GLOBAL BOOLEAN VARIABLE &STAR IS
.*SET TO 1 AND THE CURRENT CELL IS INITIALISED TO EMPTY,OTHERWISE &STAR
.*IS SET TO 0.IT IS ASSUMED THAT THE LOCATION COUNTER IS CORRECTLY
.*POSITIONED BEFORE ENTRY TO THIS MACRO.
.*       ********************************************************************
         GBLB  &STAR
         AIF   ('&A1' EQ '*').L1
         AIF   ('&A1' EQ 'NIL').L1
&STAR    SETB  0
         MEXIT
.L1      DC    X'FFFFFFFF'
*NO FLAGS IN EMPTY CELL
&STAR    SETB  1
         MEND
```

```
ALLOC     0132160O:01330400 12/20/74 12/20/74

          MACRO
          ALLOC &WF,&LNG
.*GENERAL DESCRIPTION
.*        ********************************************************
.*THIS MACRO IS CALLED BY THE SUPLST,DEF,DEFS,NATRE,VALST & LISTVAL
.*MACROS.IT IS USED TO ALLOCATE AND CHAIN NEW BLOCKS IN THE 3-GRAPH.
.*        ********************************************************
.*PARAMETERS
.*&WF CAN BE REPLACED BY THE STRING 'SUI' OR THE STRING 'VAL' WHICH
.*INDICATES THAT THE NEW BLOCK IS TO BE CHAINED TO THE END OF THE
.*CURRENT BLOCK (TO THE POINTER CELL) OR TO THE CURRENT CELL (IF NEW
.*BLOCK IS TO CONTAIN THE BEGINNING OF A VALUE LIST).
.*&LNG SPECIFIES THE LENGTH OF THE NEW BLOCK (IN BYTES).
.*        ********************************************************
.*VARIABLE SYMBOLS
          GBLB  &SPOCCDE
          GBLA  &NOBY
          LCLA  &LOCNOBY,&REM
.*&SPOCCDE SPECIFIES WHETHER THE NATURE OF THE ATTRIBUTE CONCERNED IS
.*OCCDE OR NOT
.*&NOBY IS A COUNT OF THE NUMBER OF BYTES USED IN THE CURRENT PAGE.
.*&LOCNOBY,&REM ARE WORKING VARIABLES.
.*        ********************************************************
          AIF   (('&WF' NE 'SUI') AND ('&WF' NE 'VAL')).L7
.*THE VALIDITY OF &WF IS CHECKED AND AN ERROR MESSAGE IS OUTPUT IF IT
.*IS NOT O.K.NOTE:IN THIS CASE NO NEW BLOCK IS DEFINED.
.*        ********************************************************
@LCT&SYSNDX EQU *
.*SAVE ADDRESS OF CURRENT CELL (TO WHICH NEW BLOCK WILL BE CHAINED).
.*        ********************************************************
          ORG
.*MOVE TO 1ST FREE LOCATION IE HIGHEST ADDRESS SO FAR ENCOUNTERED.
.*        ********************************************************
&LOCNOBY  SETA  &NOBY+&LNG
          AIF   (&LOCNOBY GT 4096).L1
.*TEST IF ENOUGH ROOM IN CURRENT PAGE,IF NOT GOTO .L1
.*        ********************************************************
.L4       ANOP
&NOBY     SETA  &NOBY+&LNG
.*UPDATE &NOBY TO INCLUDE NEW BLOCK.
.*        ********************************************************
@LCP&SYSNDX EQU *
.*SAVE ADDRESS OF NEW BLOCK.
.*        ********************************************************
          DC    4X'FFFFFFFF'
          DC    X'FFFFFFFF'
.*INITIALISE NEW BLOCK.
.*        ********************************************************
          ORG   @LCT&SYSNDX
.*PREPARE TO CHAIN NEW BLOCK TO POINTER CELL
```

```
ALLOC    01321600:01330400 12/20/74 12/20/74

.*       ********************************************************
         AIF    ('&WF' EQ 'SUI').L2
         AIF    ('&WF' EQ 'VAL').L3
.*TEST WHETHER BLOCK IS A TAIL BLOCK OR A VALUE LIST BLOCK FOR FLAG BITS
.*WHICH ARE DIFFERENT.
.*       ********************************************************
.*ERROR MESSAGE
.L7      MNOTE 'PAR INVALID,ALLOC MACRO'
         MEXIT
.*       ********************************************************
.*VALUE LIST FLAGS
.L3      FLAG  LV
         AGO   .L5
.*       ********************************************************
.*TAIL BLOCK FLAGS
.L2      FLAG  NOT
.*       ********************************************************
.L5      DC    AL3(&LCP&SYSNDX)
         ORG   &LCP&SYSNDX
.*FILL IN ADDRESS OF NEW BLOCK AND POSITION LOCATION COUNTER AT START
.*OF NEW BLOCK.
         MEXIT
.*       ********************************************************
.L1      ANOP
&REM     SETA  (4096-&NOBY)/4
         DC    &REM.X'FFFFFFE'
&NOBY    SETA  0
.*THE CELLS WHICH ARE LEFT AT THE END OF THE PAGE ARE INITIALISED TO
.*'EMPTY' (FOR COMPATIBILITY).&NOBY IS RESET TO 0.
         AGO   .L4
.*       ********************************************************
         MEND
```

```
TROOM    01545300:015492C0 12/20/74 12/20/74

         MACRO
         TROOM &A1
.*GENERAL DESCRIPTION
.*       ********************************************************
.*TROOM MACRO IS CALLED BY THE TNAME,DEFN,DEFNCT,MACROS.
.* PARAMETER &A1 IS THE LENGTH OF THE NEW BLOCK REQUIRED BY THE
.*CALLING MACRO.A BLOCK IN THE 3-GRAPH IS NOT ALLOWED TO SIT ASTRIDE
.*THE BOUNDARY OF TWO  PAGES SO THE TROOM MACRO IS DESIGNED TO TEST
.*WHETHER A NEW BLOCK OF THE REQUIRED LENGTH WILL FIT INTO THE CURRENT
.*PAGE.
.*       ********************************************************
.*MACRO VARIABLES
         GBLA  &NOBY
         LCLA  &REM,&LOCNOBY
.*THE GLOBAL ARITHMETIC VARIABLE &NOBY IS USED TO KEEP TRACK OF THE
.*NUMBER OF BYTES USED IN THE CURRENT PAGE.IT IS INITIALISED TO 0 AT
.*THE START OF EACH ASSEMBLY AND IS UPDATED BY THE INIT,TROOM AND ALLOC
.*MACROS.
.*LOCAL VARIABLE &LOCNOBY IS SET TO THE SUM OF &NOBY (NO. BYTES USED)
.*AND THE LENGTH OF THE BLOCK REQUIRED.IF THE SUM IS NOT GREATER THAN
.*4080 BYTES THEN THE NEW BLOCK FITS INTO THE CURRENT PAGE (THE LIMIT
.*IS 4080 SINCE,BECAUSE ALL BLOCKS IN THE 3-GRAPH ARE EITHER 20 OR 40
.*BYTES THE LAST 16 BYTES OF EACH 4K PAGE WILL NEVER BE USED).
&LOCNOBY SETA  &NOBY+&A1
         AIF   (&LOCNOBY LE 4080).TR1
.*       ********************************************************
.*THE NEW BLOCK WILL NOT FIT SO THE NO. OF 4-BYTE CELLS REMAINING IS
.*CALCULATED AND EACH OF THESE IS THEN INITIALISED TO EMPTY (FOR
.*COMPATIBILITY).
&REM     SETA  (4096-&NOBY)/4
         DC    &REM.X'FFFFFFFF'
.*       ********************************************************
.*&NOBY IS REINITIALSED AND A NEW PAGE IS STARTED.
&NOBY    SETA  &A1
         MEXIT
.*       ********************************************************
.*&NOBY IS UPDATED BY THE LENGTH OF THE NEW BLOCK.
.TR1     ANOP
&NOBY    SETA  &NOBY+&A1
         MEND
```

```
SETCNT   01470200:01474700 12/20/74 12/20/74

         MACRO
         SETCNT &CNT,&NO
.*GENERAL DESCRIPTION
.*       ********************************************************************
.*SETCNT IS CALLED BY THE DLF,DEFS,SUBLST MACROS TO FILL IN THE COUNTER
.*CELL OF A VALUE LIST.IT ASSUMES THAT THE LOCATION COUNTER IS SET TO THE
.*ADDRESS OF THE COUNTER CELL INTO WHICH IT PLACES THE NO OF CELLS
.*AVAILABLE IN THE VALUE LIST.
.*       ********************************************************************
.*PARAMETERS
.*&CNT IS REPLACED BY THE VALUE OF THE COUNTER USED IN THE CALLING
.*MACRO TO COUNT THE NO OF VALUES IN THE VALUE LIST.
.*&NO IS REPLACED BY '1' IF THE CALLING MACRO IS SUBLST,THE COUNTER
.*CONTAINS 1 MORE THAN THE ACTUAL NO OF VALUES IN THE LIST;BY '2' IF
.*THE CALLING MACRO IS DEF,BY '3' IF THE CALLING MACRO IS DEFS
.*       ********************************************************************
.*MACRO VARIABLES
         LCLA  &CT1,&CT2
.*&CT1,&CT2 ARE WORKING VARIABLES
.*       ********************************************************************
.*&CT1 IS SET TO N+1 WHERE N IS THE ACTUAL NO OF VALUES IN THE VALUE
.*LIST.
&CT1     SETA  &CNT-&NO+1
.*       ********************************************************************
.*THE NO OF CELLS ALLOCATED WILL BE THE MULTIPLE OF 4 THAT IS GREATER
.*THAN OR EQUAL TO THE NO OF VALUES IN THE LIST,-1 FOR THE COUNTER CELL
.*SO THE FOLLOWING LOOP ALTERNATELY TESTS TO SEE IF &CT1 IS A MULTIPLE
.*OF 4 AND INCREASES IT BY 1 IF IT IS NOT.
.L1      ANOP
&CT2     SETA  (&CT1/4)*4
         AIF   (&CT2 EQ &CT1).L2
&CT1     SETA  &CT1+1
         AGO   .L1
.*       ********************************************************************
.*WHEN &CT1 HAS REACHED THE MULTIPLE OF 4 REQUIRED IT IS DECREASED TO
.* ACCCOUNT FOR THE COUNTER CELL AND THEN THE COUNTER CELL IS FILLED IN.
.L2      ANOP
&CT1     SETA  &CT1-1
.L3      DC    F'&CT1'
.*       ********************************************************************
         MEND
```

```
SFLAG     01474800:01480300 12/20/74 12/20/74

          MACRO
          SFLAG &A1
.*GENERAL DESCRIPTION
.*        **********************************************************
.*THIS MACRO IS CALLED BY TESDF WHEN AN
.*EXPLICIT SUCCESSOR TO THE CURRENT 3-GRAPH ACTION IS ABSENT,THE VALUE
.*OF THE PARAMETER &A1 INDICATES WHETHER IT IS SUC1 OR SUC2 AND THIS
.*MACRO ASSUMES THAT THE LOCATION COUNTER IS  CURRENTLY POSITIONED AT
.*THE CELL CONCERNED.
.*        **********************************************************
.*MACRO VARIABLES
          GBLB  &SUCC1,&SUCC2
          GBLA  &LAB
.*&SUCC1,&SUCC2 ARE GLOBAL BOOLEAN VARIABLES WHICH INDICATE THE
.*ABSENCE OF AN EXPLICIT SUCCESSOR.
.*THE GLOBAL ARITHMETIC VARIABLE &LAB IS USED FOR GENERATING UNIQUE
.*SYMBOLS.IT IS INCREMENTED JUST BEFORE EXIT FROM THIS MACRO.
.*        **********************************************************
.*SUC1 NOT EXPLICIT
          AIF   ('&A1' NE '1').L2
&SUCC1    SETB  1
.*&SUCC1 IS SET TO 1 TO INDICATE THAT A SUC1 IS
.*NEEDED (SEE TNAME AND TFLAG MACROS).
.*        **********************************************************
.*THE CURRENT VALUE OF THE LOCATION COUNTER IS ASSOCIATED WITH A
.*SPECIAL GENERATED SYMBOL WHICH WILL BE USED FOR REPOSITIONING WHEN
.*THE SUCCESSOR IS LATER FILLED IN.(SEE TFLAG MACRO)
&S1&LAB   EQU   *
          ACO   .L3
.*        **********************************************************
.*SUC2 NOT EXPLICIT
.L2       AIF   ('&A1' NE '2').L1
&SUCC2    SETB  1
.*&SUCC2 IS SET TO INDICATE THAT A SUC2 IS NEEDED.
.*(SEE TNAME AND TFLAG MACROS).
.*        **********************************************************
.*THE CURRENT VALUE OF THE LOCATION COUNTER IS ASSOCIATED WITH A
.*SPECIAL GENERATED SYMBOL WHICH WILL BE USED FOR REPOSITIONING WHEN
.*THE SUCCESSOR IS LATER FILLED IN.(SEE TFLAG MACRO)
&S2&LAB   EQU   *
.*        **********************************************************
.*FINALLY THE CELL IS INITIALISED TO EMPTY
.*(THIS ENSURES THAT THE CONTENTS OF THE
.*CELL ARE COMPATIBLE WITH THE REST OF THE 3-GRAPH IN THE EVENTUALITY
.*OF THERE BEING NO IMPLICIT SUCCESSOR).
.L3       DC    X'FFFFFFFF'
*NO FLAGS IN EMPTY CELL
&LAB      SETA  &LAB+1
          MEXIT
.*        **********************************************************
.*ERROR MESSAGE
.L1       MNOTE 'INCORRECT CALL:MACRO SFLAG'
.*AN ERROR MESSAGE IS GENERATED IF THE VALUE OF THE PARAMETER IS NOT
.*'1' OR '2'.NOTE:THE VALUE OF THE LOCATION COUNTER IS NOT ALTERED
.*WHEN THIS OCCURS.
          MEND
```

```
TFLAG     01520900:01528400 12/20/74 12/20/74

          MACRO
          TFLAG &NR
.*GENERAL DESCRIPTION
.*        **********************************************************
.*THIS MACRO IS CALLED BY TNAME  ONCE THE NEW NOTION HAS BEEN
.*DEFINED.DEPENDING ON THE VALUE OF THE PARAMETER &NR IT TESTS THE
.*VALUE OF THE GLOBAL BOOLEAN VARIABLE &SUCC1 OR &SUCC2.IF THIS IS =1
.*(IE THE CURRENT NOTION IS THE IMPLICIT SUCCESSOR OF A PREVIOUS NOTION)
.*IT COMPLETES THE APPROPRIATE SUCCESSOR CELL OF THE PRECEDING NOTION
.*        **********************************************************
.*MACRO VARIABLES
          GBLB  &SUCC1,&SUCC2
          GBLA  &LAB
          LCLA  &LOCLAB
.*THE GLOBAL BOOLEAN VARIABLES &SUCC1 AND &SUCC2 ARE SET BY THE SFLAG
.*MACRO TO INDICATE WHEN THE SUCCESSOR TO AN ACTION (THE NOTION BEING
.*DEFINED WHEN THE MACRO IS CALLED) IS NOT EXPLICIT.
.*THE GLOBAL ARITHMETIC VARIABLE &LAB IS USED IN THE SFLAG MACRO TO
.*GENERATE A UNIQUE LABEL ATTACHED TO THE SUCCESSOR CELL
.* TO BE COMPLETED BY THE TFLAG MACRO.
.*THE LOCAL ARITHMETIC VARIABLE &LOCLAB IS USED TO REGENERATE THIS LABEL.
.*        **********************************************************
.*&SUCC1 TO BE TESTED (&NR='1')
          AIF   ('&NR' NE '1').L2
          AIF   (&SUCC1 NE 1).END
.*IF &SUCC1 = 0 CONTROL RETURNS TO THE CALLING MACRO.OTHERWISE A LABEL
.*IS GENERATED AND ATTACHED TO THE CURRENT LOCATION COUNTER VALUE.
@LCT&SYSNDX EQU *
.*        **********************************************************
.*LABEL GENERATED BY THE SFLAG MACRO IS REGENERATED AND THE LOCATION
.*COUNTER VALUE RESET TO THAT ADDRESS.THE CELL IS FILLED IN WITH THE
.*ADDRESS OF THE NEW NOTION (ASSUMING THAT THE LOCATION COUNTER WAS SET
.*TO THE BEGINNING OF THIS NOTION AT THE START OF THE TFLAG MACRO) +
.*FLAGS.&SUCC1 IS RESET TO 0.THE LOCATION COUNTER IS RESET TO THE START
.*OF THE NEW NOTION (USING LABEL GENERATED IN THIS MACRO).
&LOCLAB   SETA  &LAB-1
          ORG   @S1&LOCLAB
          FLAG  NOT
          DC    AL3(@LCT&SYSNDX)
&SUCC1    SETB  0
          ORG   @LCT&SYSNDX
          MEXIT
.*        **********************************************************
.*&SUCC2 TO BE TESTED (&NR='2')
.L2       AIF   ('&NR' NE '2').L1
.*IF &SUCC2 = 0 EXIT.
          AIF   (&SUCC2 NE 1).END
.*        **********************************************************
.*OTHERWISE SAVE CURRENT VALUE OF LOCATION COUNTER IE GENERATE LABEL
.*ATTACHED TO THIS VALUE
```

```
TFLAG     01520900:01528400 12/20/74 12/20/74

@LCT&SYSNDX EQU *
.*        **********************************************************
.*RECOVER LABEL GENERATED BY SFLAG MACRO AND RESET LOCATION COUNTER TO
.*CORRESPONDING LOCATION.
&LOCLAB   SETA  &LAB-1
          ORG   @S2&LOCLAB
.*        **********************************************************
.*FILL IN CORRESPONDING SUC2 CELL WITH FLAG BITS AND ADDRESS OF NEW
.*NOTION (ASSUMING THAT AT THE START OF THIS MACRO THE LOCATION COUNTER
.*WAS POSITIONED AT THAT ADDRESS).
          FLAG  NOT
          DC    AL3(@LCT&SYSNDX)
.*        **********************************************************
.*RESET &SUCC2 TO 0
&SUCC2    SETD  0
.*        **********************************************************
.*RESET LOCATION COUNTER TO START OF NEW NOTION (&SYSNDX IS SYSTEM
.*VARIABLE).
          ORG   @LCT&SYSNDX
          MEXIT
.*        **********************************************************
.*IF &NR NE 1 OR 2 THEN OUTPUT ERROR MESSAGE.
.*NOTE:LOCATION COUNTER NOT CHANGED.
.L1       MNOTE 'TFLAG MACRO CALL INCORRECT'
.END      MEND
```

```
FLAC      01405300:01411400 12/20/74 12/20/74

          MACRO
          FLAG  &B1
.*GENERAL DESCRIPTION
.*        ********************************************************
.*EACH CELL (4 BYTES) IN A NOTION WITH THE EXCEPTION OF THE STATISTICS
.*CELL AT THE BEGINNING OF A NOTION AND THE COUNTER CELL AT THE
.*BEGINNING OF A LIST OF VALUES HAS 4 FLAG BITS  INDICATING THE
.*NATURE OF THE CONTENTS OF THE CELL(DEPENDING ON THE POSITION OF THE
.*CELL IN THE NOTION).
.*        ********************************************************
.*THE MACRO  TESTS THE VALIDITY OF THE ACTUAL PARAMETERS
.*AND IF OK SETS THE LEFTHAND BYTE OF THE CELL ACCORDING TO THE
.*INDICATIONS GIVEN.
.*NOTE:THE 4 LOW-ORDER BITS ARE ALL SET TO ONES;THEY
.*CAN EQUALLY WELL BE SET TO ZEROES AS LONG AS NO CONFUSION IS POSSIBLE
.*ABOUT THE CONTENTS OF THE CELL.
.*        ********************************************************
.*&B1 CAN BE:
.*BASE - FOR BASIC ACTION OCCURRENCE (BIT 2:7)
.*LV - FOR LIST OF VALUES POINTER (BIT 2:6)
.*NOT - FOR NOTION (NO FLAG BITS)
.*        ********************************************************
          AIF ('&B1' NE 'BASE').L1
          DC    X'8F'
          AGO   .L8
.L1       AIF   ('&B1' NE 'LV').L2
          DC    X'4F'
          AGO   .L8
.L2       ANOP
.L3       AIF   ('&B1' NE 'NOT').ERR
          DC    X'0F'
          AGO   .L8
.ERR      MNOTE 'ERROR IN FLAG PARAMETERS'
.L8       MEND
```

```
MXREF     C10o1700:01067300 04/30/75 04/30/75

          MACRO
          MXREF  ,XREF OF SIMULATOR MACROS
.**********************************************************************
.*ROUTINE        MACROS CALLED
.**********************************************************************
.*ROOT PHASE
.*                ERRCALL
.*INITIALISER
.*MAIN            RDACHK
.*SCANPAGE        INCR,TSTFLAG
.*TPTPOUT         INCR
.*CPREDICT        PAGNO
.*SIMULATOR
.*SIMCOMP         TSTARG,MVARG
.*SIMCREN         TSTARG,CREMOT,FNDATTR,ADDVAL,MVARG,SPATTR,CRATTR
.*SIMCRER         TSTARG,FNDATTR,FNDVAL,MVARG,ADDVAL,SPATTR,CRATTR
.*SIMIFOU         TSTARG,MVARG
.*SIMLAT          TSTARG,CHATRAN,TSLIST,CHVAL,LATNAT,LATVAL,MVARG,MVAL
.*SIMREP          TSTARG,FNDATTR,TSLIST,CHVAL,MVARG,REPVAL,MVAL
.*SIMSUP          TSTARG,FNDATTR,MVARG,TSLIST,SUPATTR,CHVAL,SUPVAL
.*SIMSUPR         TSTARG,FNDATTR,MVARG,TSLIST,CVAL,SUPATTR,FNDVAL,SUPVAL
.*SIMTMV          TSTARG,FNDATTR,MVARG,TSLIST,CHVAL,SPATTR,CRATTR,MVAL,
.*                FNDVAL,ADDVAL
.*FNDATSUB        GBLCK
.*FNDVALSU        GBLCK
.*FSTATSUB        ERRCALL
.*GETDLK          ERRCALL
.*PARLSUP         ERRCALL
.*RETNOTSE        PAGNO,TSLIST,RETVC,DECRF,CPAGE,NAMELK,TSTFLAG
.*RETVALST        PAGNO,DECRF,CPAGE,TSTFLAG
.*TSTPLY          ERRCALL
.*I/O ROUTINES
.*ANALYSIS        FNDNAME,ERRCALL,WRERR
.*#ATHALT         WRERR
.*#BRANCH         GETADD
.*#NEXTVAL        GETNAM,PUTNAM,PUTLINE
.*#PRINOT         GETADD,PUTNAM,OUTATT,PUTLINE
.*#PRTATTR        GETADD,GETATTR,PUTNAM,TSLIST,GETNAM,PUTLINE,WRERR
.*#SETHALT        GETADD
.**********************************************************************
.*MACRO           MACROS CALLED
.**********************************************************************
.*ADDVAL          TSLIST,INCR,TSTFLAG
.*CRATTR          INCR,TSTFLAG
.*DECRF           ERRCALL
.*FNDNAME         WRERR
.*FNDVAL          TSLIST
.*GETADD          CRNAM
.*GETNAM          TSTFLAG,CRNAM
.*LATNAT          INCR,TSLIST,TSTFLAG
.*OUTATT          GETNAM,PUTNAM,TSLIST,PUTLINE
.*REPVAL          DECR,MVAL,INCR,TSTFLAG
.*SPATTR          INCR,TSTFLAG
.*SUPATTR         DECR,TSTFLAG
.*SUPVAL          DECR,TSTFLAG
.**********************************************************************
          MEND
```

```
ADDVAL   01000100:01003400 04/30/75 04/30/75

         MACRO
&LAB     ADDVAL &ADDR,&VAL
*********************************************************************
**
**
.*(&ADDR)= ADDRESS OF THE VALUE CELL OF AN ATTRIBUTE,(&VAL) ARE ADDED
.*TO THE VALUES OF THIS ATTRIBUTE.A VALUE LIST IS CREATED IF NECESSARY
&LAB     TSLIST &ADDR,L2&SYSNDX
*IF A VALUE LIST ALREADY EXISTS CONTROL PASSES TO L2&SYSNDX,OTHERWISE
*THE NEXT INSTRUCTIONS CREATE A VALUE LIST FOR THE ATTRIBUTE
         MVI   NOBLCKS,X'14'        NOBLCKS=20,PARAM FOR GETBLK ROUTINE
         LR    NEWADD,&ADDR         START ADDR FOR SEARCH,PARAM GETBLK
         BAL   LINK,GETBLK          CALL ROUTINE TO ALLOCATE NEW BLOCK
*ADDRESS OF NEW BLOCK RETURNED IN NEWADD
         MVC   4(4,NEWADD),0(&ADDR) MOVE EXISTING VALUE INTO LIST
         ST    NEWADD,0(&ADDR)      CHAIN LIST TO VALUE CELL
         MVI   0(&ADDR),X'4F'       SET 'VALUE LIST' FLAGS
         MVC   0(4,NEWADD),=F'3'    COUNTER CELL,NO OF CELLS ALLOCATED
         MVC   8(4,NEWADD),&VAL     ADD NEW VALUE TO LIST
         B     EX&SYSNDX
L2&SYSNDX L    VALADD,EMPTY         VALUE REQUIRED,PARAM FOR FNDVALSU
         AIF   ('&ADDR' EQ 'ATADD').L1
         LR    ATADD,&ADDR          ADDR OF VALUE LIST,PARAM FNDVALSU
.L1      BAL   LINK,FNDVALSU        CALL ROUTINE TO SEARCH VALUE LIST
*FOR CELL CONTAINING GIVEN VALUE,RESULT OF SEARCH RETURNED IN VALADD
         MVC   0(4,VALADD),&VAL     STORE NEW VALUE IN EMPTY CELL
*INCREMENT REF COUNT IF NECESSARY
EX&SYSNDX TSTFLAG (OBJUN,&VAL,EX1&SYSNDX),(BASIC,&VAL,EX1&SYSNDX)
         INCR  &VAL
EX1&SYSNDX EQU *
**
**
*********************************************************************
         MEND
```

```
CHATRAN  01003500:01005800 04/30/75 04/30/75

         MACRO
&LAB     CHATRAN &NOT,&ATADD,&NAT,&EXIT
*********************************************************************
**
**
.*(&NOT)= ADDRESS OF A NOTION.AN ATTRIBUTE IS CHOSEN AT RANDOM,ITS
.*(VALUE CELL) ADDRESS IS RETURNED IN &ATADD,ITS NATURE (NOTION
.*ADDRESS) IN &NAT AND IF THE ATTRIBUTE CHOSEN IS 'NIL' CONTROL GOES TO
.*&EXIT.
&LAB     L     ROOTAD,&NOT          ADDRESS OF NOTION,CHRATSUB PARAMETER
         BAL   LINK,CHRATSUB        CALL ROUTINE TO CHOOSE AT RANDOM,AN
*ATTRIBUTE OF A GIVEN NOTION ADDR OF ATTRIBUTE'S VALUE CELL RETURNED IN
*ATADD,ADDRESS OF NOTION (NATURE OF ATTR) IN NATADD
         AIF   ('&ATADD' EQ 'ATADD').L1
         LR    &ATADD,ATADD         STORE ATTRIBUTE ADDRESS
.L1      ST    NATADD,&NAT          STORE NATURE OF ATTRIBUTE
         CLC   &NAT,EMPTY
         BE    &EXIT
*IF THE ATTRIBUTE IS NIL,THE VALUE OF THE ATTRIBUTE IS ALSO NIL AND
*CONTROL GOES TO &EXIT
**
**
*********************************************************************
         MEND

CHVAL    01005900:01008700 04/30/75 04/30/75

         MACRO
&LAB     CHVAL &ATADD,&RET,&ZONE,&EXIT
*********************************************************************
**
**
.*(&ATADD)= ADDRESS OF VALUE CELL OF AN ATTRIBUTE WITH A LIST OF VALUES
.*A VALUE CELL IN THE LIST IS CHOSEN AT RANDOM,ITS ADDRESS IS RETURNED
.*IN &RET,ITS CONTENTS IN &ZONE,IF THE VALUE CHOSEN IS 'NIL' &EXIT IS
.*TAKEN.
         AIF   ('&ATADD' EQ 'ATADD').L1
&LAB     LR    ATADD,&ATADD         ADDRESS OF VALUE LIST,RANSUB PARA"
         BAL   LINK,RANSUB          CALL ROUTINE WHICH CHOOSES A VALUE
*AT RANDOM,ADDRESS OF CELL CHOSEN RETURNED IN VALADD
.L3      AIF   ('&RET' EQ 'VALADD').L2
         LR    &RET,VALADD          STORE ADDRESS OF CELL CHOSEN
.L2      AIF   ('&EXIT' EQ '').L4
         C     VALADD,EMPTY         VALUE RETURNED IS NIL?
         BE    &EXIT                YES->EXIT
.L4      AIF   ('&ZONE' EQ '').END
         MVC   &ZONE,0(VALADD)   STORE VALUE CHOSEN
.END     MEXIT
.L1      ANOP
&LAB     BAL   LINK,RANSUB          CALL ROUTINE WHICH CHOOSES A VALUE
*AT RANDOM,ADDRESS OF CELL CHOSEN RETURNED IN VALADD
         AGO   .L3
**
**
*********************************************************************
         MEND
```

```
CPAGE    01006800:01011200 04/30/75 04/30/75

         MACRO
&LAB     CPAGE &REG,&ZONE,&EXIT,&SAVE2
*********************************************************************
**
**
.*&REG ADDRESSES A POINTER CELL,THE PAGE NO. OF THE BLOCK POINTED TO IS
.*COMPARED WITH THE PAGE NO. IN &ZONE.IF THE SAME,&EXIT IS TAKEN,
.*OTHERWISE THE CHAIN OF BLOCKS UP TO AND INCLUDING THE CURRENT ONE IS
.*RETURNED TO THE FREE LIST.&SAVE2 IS A 2-WORD SAVE AREA.
&LAB     L       WKREG1,0(&REG)        NEXT BLOCK
         SRL     WKREG1,12             ISOLATE PAGE NO
         STC     WKREG1,WORK
         CLC     &ZONE(1),WORK         =CURRENT PAGE NO?
         BE      &EXIT                 YES->BRANCH
*RETURN BLOCKS TO FREE LIST,ASSUME ROOTAD O.K.
         MVC     &SAVE2.(4),0(&REG)     SAVE NEXT BLOCK ADDR
         MVC     0(4,11),LSTPT         END OF CHAIN FOR RETBLK
         ST      LINK,&SAVE2+4         SAVE CURRENT RETURN ADDR
         BAL     LINK,RETBLK           REMOVE BLOCKS
         L       LINK,&SAVE2+4         RESTORE RETURN ADDR
         L       ROOTAD,&SAVE2         START ADDRESS OF LIST
**
**
*********************************************************************
         MEND

PAGNO    01073300:01074500 04/30/75 04/30/75

         MACRO
&LAB     PAGNO &REG,&ZONE
*********************************************************************
**
**
.*ISOLATE PAGE NO. OF ADDRESS IN &REG AND STORE IN &ZONE.
&LAB     LR      WKREG1,&REG           ADDRESS IN PAGE
         SRL     WKREG1,12             REMOVE REL ADDRESS IN PAGE
         STC     WKREG1,&ZONE          STORE PAGE NUMBER
**
**
*********************************************************************
         MEND
```

```
CRENOT   01013300:01014900 04/30/75 04/30/75

         MACRO
&LAB     CRENOT &NOTADDR,&CHADDR
*********************************************************************
**
**
.*SPACE FOR A NEW NOTION IS ALLOCATED AND ITS ADDRESS RETURNED IN
.*&NOTADDR.&CHADDR SERVES AS A BASIS FOR THE ALLOCATION SEARCH.
&LAB     MVI   NOBLCKS,X'28'        NOBLCKS SET TO 40,PARAM FOR GETBLK
         L     NEWADD,&CHADDR       START ADDR FOR SEARCH,PAR FOR GETBLK
         BAL   LINK,GETBLK          CALL ROUTINE TO ALLOCATE SPACE FOR
*NEW NOTION,ADDRESS OF NEW BLOCK RETURNED IN NEWADD
         ST    NEWADD,&NOTADDR      SAVE ADDRESS OF NEW NOTION
         MVC   0(4,NEWADD),=F'0'    INITIALISE REF COUNT
**
**
*********************************************************************
         MEND
CRATTR   01011300:01013200 04/30/75 04/30/75

         MACRO
&LAB     CRATTR &ADD,&NAT,&VAL
*********************************************************************
**
**
.*&NAT AND &VAL CONTAIN THE NATURE AND VALUE OF A NEW ATTRIBUTE RESP.
.*&ADD POINTS TO THE VALUE CELL OF AN EMPTY ATTRIBUTE WHERE THE NEW
.*ATTRIBUTE IS STORED.
&LAB     MVC   0(4,&ADD),&VAL       VALUE OF NEW ATTRIBUTE STORED
         S     &ADD,=F'4'           POSITION ON NATURE CELL
         MVC   0(4,&ADD),&NAT       NATURE OF NEW ATTRIBUTE STORED
*INCREMENT REF COUNT OF NATURE &,IF NECESSARY,VALUE OF ATTRIBUTE
         INCR  &NAT
         TSTFLAG (OPJUN,&VAL,CR&SYSNDX),(BASIC,&VAL,CR&SYSNDX)
         INCR  &VAL
CR&SYSNDX EQU  *
**
**
*********************************************************************
         MEND
CVAL     01017000:01018400 04/30/75 04/30/75

         MACRO
&LAB     CVAL  RADDR,&VAL,&EXIT1,&EXIT2
*********************************************************************
**
**
.*IF (&ADDR)=(&VAL) THEN &EXIT2 ELSE &EXIT1.
&LAB     CLC   0(4,&ADDR),&VAL      DOES CELL CONTAIN GIVEN VALUE?
         AIF   ('&EXIT1' EQ '').L1
         BNE   &EXIT1               NO,TRANSFER CONTROL
.L1      AIF   ('&EXIT2' EQ '').END
         BE    &EXIT2               YES,TRANSFER CONTROL
**
**
*********************************************************************
.END     MEND
```

```
DECR     01098800:01101800 05/02/75 05/02/75

         MACRO
&LAB     DECR  &INDAD
*********************************************************************
**
**
.*DECREMENT REF COUNT OF NOTION REFERENCED IN &INDAD,IF 0 REMOVE NOTION
.*FROM GRAPH (UNLESS A SPECIAL NOTION).
.*NOTE:(&INDAD)= ADDR OF NOTION
&LAB     L     WKREG1,0(&INDAD)     ADDRESS OF STATISTICS CELL
         L     WKREG2,0(WKREG1)     CONTENTS (REFERENCE COUNT)
         OC    SAVE,0(WKREG1)       SAVE FLAGS
         LA    WKREG2,0(WKREG2)     MASK FLAGS
         BCT   WKREG2,DECR&SYSNDX   DECREASE REFERENCE COUNT,WHILE IT IS
*GREATER THAN 0 CONTROL TRANSFERS TO DECR&SYSNDX
*WHEN REFERENCE COUNT IS ZERO NOTION IS NO LONGER ACCESSIBLE SO THE
*SPACE IT USES CAN BE RETURNED TO THE FREE LIST
         LA    WKREG1,0(WKREG1)     REMOVE FLAG BITS
         C     WKREG1,=A(ARG4)      SPECIAL NOTION?
         BNH   DECR&SYSNDX          YES,NOTION NOT REMOVED
         LR    ROOTAD,WKREG1        START ADDRESS OF NOTION,RETURN PARAM
         BAL   LINK,RETNOTSB        CALL ROUTINE TO RETURN BLOCKS OF
*                                   NOTION TO FREE LIST
         B     DCR1&SYSNDX          SKIP NEXT INSTRUCTIONS
DECR&SYSNDX ST WKREG2,0(WKREG1)     STORE REFERENCE COUNT
         OC    0(1,WKREG1),SAVE     RESTORE FLAGS
DCR1&SYSNDX EQU *
         XC    SAVE,SAVE            REINITIALISE
**
**
*********************************************************************
         MEND


INCR     01047900:01049600 04/30/75 04/30/75

         MACRO
&LAB     INCR  &NOT
*********************************************************************
**
**
.*&NOT INDIRECTLY ADDRESSES THE REFERENCE COUNT OF A NOTION.
.*THIS IS INCREMENTED TO REFLECT AN EXTRA REFERENCE TO THE NOTION.
&LAB     L     WKREG1,&NOT          ADDRESS OF STATISTICS CELL
         L     WKREG2,0(WKREG1)     CONTENTS (REFERENCE COUNT)
         OC    SAVE,0(WKREG1)       SAVE FLAGS
         LA    WKREG2,1(WKREG2)     INCREMENT REFERENCE COUNT
         ST    WKREG2,0(WKREG1)     STORE IN STATISTICS CELL
         OC    0(1,WKREG1),SAVE     RESTORE FLAGS
         XC    SAVE,SAVE            REINITIALISE
**
**
*********************************************************************
         MEND
```

```
DECRF     01021200:01024900 04/30/75 04/30/75

          MACRO
&LAB      DECRF &REG,&EXIT
          GBLB  &DFLAG
***********************************************
**
**
.*DECREMENT REF COUNT OF NOTION REFERENCED IN &REG.IF REF COUNT=0 ADD
.*NOTION TO NOTSTACK FOR LATER REMOVAL FROM THE GRAPH,THEN GO TO &EXIT.
&LAB      L       WKREG1,0(&REG)        ADDRESS OF NOTION
          LA      WKREG2,&EXIT          RETURN ADDRESS
          AIF     (NOT &DFLAG).L1
.*THE 1ST CALL TO THIS MACRO GENERATES A SMALL ROUTINE
.*SUBSEQUENT CALLS OF THE MACRO GENERATE CALLS
.*TO THE ROUTINE.
          B       @DECR1                TO ROUTINE
          MEXIT
.L1       ANOP
@DECR1    ST      WKREG2,@DECRS
          L       WKREG2,0(WKREG1)      REF COUNT
          OC      SAVE,0(WKREG1)        SAVE FLAGS
          LA      WKREG2,0(WKREG2)      MASK FLAGS
          BCT     WKREG2,@DECR2         DECR REF COUNT:WHILE >0 BRANCH
          ST      WKREG1,0(10)          ADD NOTION ADDRESS TO STACK
          LA      10,4(10)              UPDATE STACK POINTER
          C       10,=A(NOTSTEND)       STACK FULL?
          BNE     @DECR3                NO->EXIT
          ERRCALL MESS                  ERROR MESSAGE & TERM'D
@DECR2    ST      WKREG2,0(WKREG1)      STORE UPDATED REFERENCE COUNT
          OC      0(1,WKREG1),SAVE      RESTORE FLAGS
          XC      SAVE,SAVE             REINITIALISE
@DECR3    L       WKREG2,@DECRS         RETURN ADDRESS
          BR      WKREG2                EXIT
@DECRS    DS      F
&DFLAG    SETB    1
**
**
***********************************************
          MEND
```

```
FNDATTR  01026400:01028100 04/30/75 04/30/75

         MACRO
&LAB     FNDATTR &ROOT,&NAT,&ATADD
*********************************************************************
**
**
.*&ROOT AND &NAT SPECIFY THE ROOT AND NATURE OF AN ATTRIBUTE RESP.&ROOT
.*IS SEARCHED FOR THIS ATTRIBUTE.(&ATADD)=VALUE CELL OF THE ATTRIBUTE
.*OR INDICATE 'NO ATTRIBUTE FOUND' AFTER SEARCH.
&LAB     L       ROOTAD,&ROOT          ADDR OF ROOT NOTION,PAR FOR FNDATSUB
         L       NATADD,&NAT           ADDR OF NATURE NOTION,PAR FNDATSUB
         BAL     LINK,FNDATSUB         CALL ROUTINE TO FIND ATTRIBUTE,
*RESULT OF SEARCH RETURNED IN ATADD
         AIF     ('&ATADD' EQ 'ATADD').END
         LR      &ATADD,ATADD          STORE RESULT OF SEARCH
**
**
*********************************************************************
.END     MEND


FNDVAL   01031400:01033900 04/30/75 04/30/75

         MACRO
&LAB     FNDVAL &ADDR,&VAL,&INDIC
*********************************************************************
**
**
.*(&ADDR)= VALUE CELL OF AN ATTRIBUTE.THE VALUE(S) OF THE ATTRIBUTE IS
.*(ARE) COMPARED WITH (&VAL) TO SEE IF THE VALUE SPECIFIED IS A VALUE
.*OF THE ATTRIBUTE.&INDIC INDICATES THE RESULT OF THE COMPARISON.
&LAB     TSLIST &ADDR,L2&SYSNDX
         CLC     &VAL,0(&ADDR)         VALUE CORRESPONDS?
         BE      L1&SYSNDX             YES
         BNE     L3&SYSNDX             NO
L2&SYSNDX L      VALADD,&VAL            VALUE REQUIRED,PARAM FOR FNDVALSU
         AIF     ('&ADDR' EQ 'ATADD').L1
         LR      ATADD,&ADDR           ADDR OF VALUE LIST,PARAM FNDVALSU
.L1      BAL     LINK,FNDVALSU         CALL ROUTINE TO SEARCH VALUE LIST
*FOR CELL CONTAINING GIVEN VALUE,RESULT OF SEARCH RETURNED IN VALADD
         C       VALADD,LSTFT           SEARCH FAILED?
         BNE     L1&SYSNDX             NO->L1&SYSNDX
L3&SYSNDX MVI     &INDIC,X'00'          INDICATE SEARCH FAILED
         B       L1&SYSNDX+4
L1&SYSNDX MVI   &INDIC,X'01'           INDICATE SEARCH SUCCESSFUL
**
**
*********************************************************************
         MEND
```

```
GBLCK     01034000:01036600 04/30/75 04/30/75

          MACRO
&LAB      GBLCK &SAVE,&REG,&ROOT,&RET
**********************************************************************
**
**
.*A NEW BLOCK IS ALLOCATED AND CHAINED TO THE POINTER CELL ADDRESSED BY
.*&REG.(&ROOT SERVES AS A BASIS FOR THE ALLOCATION SEARCH).THE ADDRESS
.*OF THE NEW BLOCK IS RETURNED IN &RET.&SAVE IS A 1-WORD SAVE AREA.
&LAB      MVI   NOBLCKS,X'14'        1 BLOCK REQUIRED,GETBLK PARAM
          LR    NEWADD,&ROOT         SEARCH ADDR,GETBLK PARAM
          ST    LINK,&SAVE           SAVE CURRENT RETURN ADDRESS
          BAL   LINK,GETBLK          CALL ROUTINE TO ALLOCATE NEW BLOCK,
*ADDRESS OF NEW BLOCK RETURNED IN NEWADD
          ST    NEWADD,0(&REG)       CHAIN NEW BLOCK TO END OF LIST
          MVI   0(&REG),X'0F'        SET POINTER CELL FLAGS
          AIF   ('&ROOT' EQ 'ROOTAD').L1
          L     &ROOT,0(&ROOT)       COUNTER CELL ADDRESS
          L     WKREG1,0(&ROOT)      UPDATE
          LA    WKREG1,4(WKREG1)     COUNTER
          ST    WKREG1,0(&ROOT)      CELL
.L1       NI    0(&ROOT),X'EF'       INDICATE NO EMPTY BLOCKS IN LIST
          LR    &RET,NEWADD          RETURN ADDRESS OF NEW BLOCK
          L     LINK,&SAVE           RESTORE RETURN ADDRESS
**
**
**********************************************************************
          MEND


MVARG     01060300:01061600 04/30/75 04/30/75

          MACRO
&LAB      MVARG &ACT,&DISP,&LINK,&SAVE
**********************************************************************
**
**
.*(&ACT)= NOTION,(&DISP)= DISPLACEMENT IN THE NOTION TO POINT TO A
.*SPECIAL ATTRIBUTE.(&LINK)= RETURN ADDRESS STORED IN &SAVE.
&LAB      LA    &ACT,&DISP.(&ACT)    MOVE TO SPECIAL  ATTR CELL INDICATED
          AIF   ('&SAVE' EQ '').END
          L     &LINK,&SAVE          RESTORE RETURN ADDR TO CALLING ROUT
**
**
**********************************************************************
.END      MEND
```

```
LATNAT   01049700:01054100 04/30/75 04/30/75

         MACRO
&LAB     LATNAT &ACT,&DISP,&ZONE,&EXIT
**********************************************************************
**
**
.*(&ACT)= ADDRESS OF CURRENT LAT ACTION OCCURRENCE.
.*(&DISP)= DISPLACEMENT FROM (&ACT) FOR ARG2(3) ATTRIBUTE.(&ZONE) ARE
.*ADDED AS A VALUE TO THE ATTRIBUTE INDICATED.
.*NOTE:FROM LATATCLC TO LATSW IS A SUBROUTINE ALSO CALLED FROM THE
.*EXPANSION OF LATVAL MACRO WHICH RESETS LATSW TO A 'B'.
&LAB     MVI   LATSW+1,X'00'          BRANCH ADDRESS
         LA    ATADD,&DISP.(&ACT)   ADDRESS OF ARG CELL
         LA    NATADD,SPATTAB+&DISP-8 ADDR OF ARG2 NOTION
         MVC   ADDSAVE,&ZONE        STORE VALUE TO BE ADDED TO ARG
LATATCLC CLC   0(4,ATADD),EMPTY    VALUE OF ARG ALREADY EXISTS?
         BNE   LATATEX              YES->LATATEX
         MVC   0(4,ATADD),ADDSAVE   STORE VALUE
         INCR  0(NATADD)
*UPDATE STATISTICS CELL OF ARG NOTION CONCERNED
         B     LATINC
LATATEX  TSLIST ATADD,,LATATCR
*IF LIST OF VALUES DOES NOT YET EXIST,GO TO LATATCR
         L     VALADD,EMPTY         EMPTY CELL REQUIRED,FNDVALSU PARAM
*                                   ATADD ALREADY SET,FNDVALSU PARAM
         BAL   LINK,FNDVALSU        CALL RCUTINE TO SEARCH VALUE LIST
*FOR EMPTY CELL,RESULT OF SEARCH RETURNED IN VALADD
         MVC   0(4,VALADD),ADDSAVE  STORE VALUE IN LIST
         B     LATINC
LATATCR  MVI   NOBLCKS,X'14'        NOBLCKS=20,GETBLK PARAM
         LR    NEWADD,ATADD         SEARCH ADDRESS,GETBLK PARAM
         BAL   LINK,GETBLK          CALL ROUTINE TO ALLOCATE NEW BLOCK,
*ADDRESS OF NEW BLOCK RETURNED IN NEWADD
         MVC   4(4,NEWADD),0(ATADD) STORE EXISTING VALUE IN LIST
         ST    NEWADD,0(ATADD)      CHAIN NEW BLOCK TO VALUE CELL
         MVI   0(ATADD),X'4F'       SET 'VALUE LIST' FLAGS
         MVC   8(4,NEWADD),ADDSAVE  STORE VALUE
         MVC   0(4,NEWADD),=F'3'    COUNTER CELL,NO OF CELLS ALLOCATED
LATINC   TSTFLAG (OBJUN,ADDSAVE,LATSW),(BASIC,ADDSAVE,LATSW)
         INCR  ADDSAVE
*UPDATE STATISTICS CELL OF NOTION (VALUE OF ARG ATTRIBUTE) IF NECESSARY
LATSW    B     LATSUC1              EXIT
**
**
**********************************************************************
         MEND
```

```
LATVAL   01054200:01055700 04/30/75 04/30/75

         MACRO
&LAB     LATVAL &ACT,&DISP,&ZONE,&EXIT
*********************************************************************
**
**
.*NOTE:EXPANSION OF THIS MACRO MERELY SETS PARAMETERS FOR THE
.*SUBROUTINE GENERATED BY LATNAT MACRO.
&LAB     LA    ATADD,&DISP.(&ACT)  ADDR OF ARG CELL
         LA    MATADD,SPATTAB+&DISP-8 ADDR OF ARG3 NCTION
         MVC   ADDSAVE,&ZONE       VALUE TO BE ADDED TO ARG
         MVI   LATSW+1,X'F0'          SET RETURN ADDRESS
         B     LATATCLC
**
**
*********************************************************************
         MEND
MVAL     01058700:01060200 04/30/75 04/30/75

         MACRO
&LAB     MVAL  &ADDR,&ZONE,&DIR
*********************************************************************
**
**
.*IF &DIR IS NULL CONTENTS OF ZONES ADDRESSED BY &ADDR ARE MOVED TO
.*&ZONE,OTHERWISE REVERSE HAPPENS.
         AIF   ('&DIR' EQ 'REV').L1
&LAB     MVC   &ZONE,0(&ADDR)      STORE VALUE
         MEXIT
.L1      ANOP
&LAB     MVC   0(4,&ADDR),&ZONE    STORE VALUE
**
**
*********************************************************************
         MEND

REPVAL   01079200:01080900 04/30/75 04/30/75

         MACRO
&LAB     REPVAL &CELL,&VALUE,&EXIT
*********************************************************************
**
**
.*REPLACES CONTENTS OF CELL ADDRESSED BY &CELL WITH (&VALUE) THEN
.*CONTROL GOES TO &EXIT.
&LAB   TSTFLAG (OBJUN,0(&CELL),REP&SYSNDX),(BASIC,0(&CELL),REP&SYSNDX)
         DECR  &CELL               DECR REF COUNT OF VALUE TO BE
*                                  REPLACED
REP&SYSNDX  MVAL  &CELL,&VALUE,REV   REPLACE VALUE
         TSTFLAG (OBJUN,&VALUE,&EXIT),(BASIC,&VALUE,&EXIT)
         INCR  &VALUE              INCR NEW VALUE REF COUNT
         B     &EXIT               EXIT
**
**
*********************************************************************
         MEND
```

```
RETVC     01081000:01082900 04/30/75 04/30/75

          MACRO
&LAB      RETVC &ZONE1,&DISP,&REG
**********************************************************************
**
**
.*CALLS RETVALST ROUTINE TO REMOVE VALUE LIST POINTED TO BY CELL AT
.*&DISP (DISPLACEMENT) FROM ADDRESS IN &REG.&ZONE1 IS 2-WORD SAVE AREA
&LAB      ST    ROOTAD,&ZONE1          SAVE START ADDRESS
          ST    LINK,&ZONE1+4          SAVE CURRENT RETURN ADDRESS
          ST    10,NOTSTEND            & CURRENT POSITION IN STACK
          L     ROOTAD,&DISP.(&REG)    PARAM FOR RETVALST ROUTINE
*SW ALREADY POSITIONED
          BAL   LINK,RETVALST
          L     LINK,&ZONE1+4          RESTORE REGS
          L     ROOTAD,&ZONE1
          L     10,NOTSTEND
**
**
**********************************************************************
          MEND


NAMELK    01067400:01070100 04/30/75 04/30/75

          MACRO
&LAB      NAMELK &ROOT,&EXIT,&KEYARG,&FILE,&SAVE
**********************************************************************
**
**
.*(&ROOT)= ADDRESS OF A NOTION.IF THE NOTION HAS NO NAME BLOCK GOTO
.*&EXIT.OTHERWISE NAME IS REMOVED FROM THE DICTIONARY (&FILE,&KEYARG).
.*&SAVE IS A 2-WORD SAVE AREA.
&LAB      TM    0(&ROOT),NAMEL         NOTION NAMED?
          BZ    &EXIT                  NO->
          L     1,=V(#NOFIND)          SET
          MVI   0(1),X'02'             ERROR CODE
          LA    WKREG1,40(&ROOT)       REMOVE ENTRY
          L     1,=V(&KEYARG)
          MVC   0(8,1),0(WKREG1)       FROM
          L     1,=V(&FILE)
          ELIM  (1),KEY                DICTIONARY
          MVC   16(4,WKREG1),LSTPT     SET END OF BLOCK FOR RLTBLK ROUTINE
          ST    LINK,&SAVE             SAVE CURRENT RETURN ADDR
          ST    ROOTAD,&SAVE+4         SAVE NOTION ADDRESS
          LR    ROOTAD,WKREG1          PARAM,RETBLK ROUTINE
          BAL   LINK,RETBLK            RETURN NAME TO FREE LIST
          L     LINK,&SAVE             RESTORE RETURN ADDR
          L     ROOTAD,&SAVE+4         RESTORE NOTION ADDRESS
**
**
**********************************************************************
          MEND
```

```
SPATTR   01083000:01085100 04/30/75 04/30/75

         MACRO
&LAB     SPATTR &NAT,&ADDR,&VAL,&EXIT1,&EXIT2
*********************************************************************
**
**
.*A NEW ATTRIBUTE WITH NATURE AND VALUE SPECIFIED BY &NAT AND &VAL RESP
.*IS TO BE ADDED TO A NOTION AT &ADDR.IF THE NATURE IS NOT SPECIAL THEN
.*&EXIT1 ELSE STORE (&VAL) AT &ADDR,INCREMENT NATURE AND VALUE REF
.*COUNTS AND GOTO &EXIT2.
&LAB     NC    &NAT,MASK          MASK FLAGS
         CLC   &NAT,=A(ARG4)      NATURE SPECIAL?
         BH    &EXIT1             NO
         L     ATADD,&ADDR        ADDR OF SPECIAL ATTRIBUTE
         MVC   0(4,ATADD),&VAL    STORE VALUE
         INCR  &NAT               INCR NATURE REF COUNT
         TSTFLAG (OBJUN,&VAL,&EXIT2),(BASIC,&VAL,&EXIT2)
         INCR  &VAL               INCR VALUE REF COUNT
         B     &EXIT2             EXIT
**
**
*********************************************************************
         MEND

TSLIST   01090800:01092300 04/30/75 04/30/75

         MACRO
&LAB     TSLIST &ADDR,&EXIT1,&EXIT2
*********************************************************************
**
**
.*IF (&ADDR) POINTS TO LIST OF VALUES TRANSFER CONTROL WHEN &EXIT1
.*SPECIFIED.IF NOT POINTER TRANSFER CONTROL WHEN &EXIT2 IS SPECIFIED.
&LAB     TM    0(&ADDR),LIST      LIST OF VALUES?
         AIF   ('&EXIT1' EQ '').L1
         BO    &EXIT1             YES:TRANSFER CONTROL
.L1      AIF   ('&EXIT2' EQ '').END
         BZ    &EXIT2             NO,TRANSFER CONTROL
**
**
*********************************************************************
.END     MEND

WRERR    01097500:01098700 04/30/75 04/30/75

         MACRO
&LAB     WRERR &MESS
*********************************************************************
**
**
.*WROUTRET OUTPUTS MESSAGE WITH ADDRESS &MESS.
&LAB     LA    1,WRPARAM          PARAM LIST
         MVC   1(3,1),=AL3(&MESS) MESSAGE ADDRESS
         B     WROUTRET           TO MESSAGE OUTPUT
**
**
*********************************************************************
         MEND
```

```
SUPATTR  01085200:01088400 04/30/75 04/30/75

         MACRO
&LAB     SUPATTR &ATADDR,&NAT,&ROOT
*********************************************************************
**
**
.*SUPPRESSES ATTRIBUTE OF NOTION-(&ROOT)= ADDRESS OF NOTION-WITH NATURE
.*SPEC BY &NAT,THAT IS AT &ATADDR.DECREMENTS REF COUNTS,THEN TESTS IF
.*BLOCK CONTAINING ATTRIBUTE IS EMPTY AND CAN BE REMOVED FROM NOTION.
&LAB     TSTFLAG (OBJUN,0(&ATADDR),L3&SYSNDX),(BASIC,0(&ATADDR),L3&SYSNC
               DX)
         DECR  &ATADDR
*STATISTICS CELL OF NOTION (VALUE OF ATTRIBUTE) UPDATED
L3&SYSNDX MVC   0(4,&ATADDR),EMPTY  SUPPRESS VALUE OF ATTRIBUTE
         CLC   &NAT+1(3),=AL3(ARG4)        NATURE OF ATTRIBUTE SPECIAL?
         BNH   L1&SYSNDX             YES->L1&SYSNDX
         S     &ATADDR,=F'4'         POSITION ON NATURE CELL
         DECR  &ATADDR
*STATISTICS CELL OF NOTION (NATURE OF ATTRIBUTE) UPDATED
         MVC   0(4,&ATADDR),EMPTY    SUPPRESS NATURE OF ATTRIBUTE
         B     L2&SYSNDX
L1&SYSNDX L    WKREG1,&NAT           URDATE
         L     WKREG2,0(WKREG1)      STATISTICS CELL;
         BCTR  WKREG2,0              NATURE OF ATTRIBUTE
         ST    WKREG2,0(WKREG1)      IS SPECIAL
L2&SYSNDX LR    CELL,&ATADDR          ADDR OF CELL EMPTIED,TSTBLK PARAM
         L     STADDR,&ROOT          START ADDR OF NOTION:TSTBLK PARAM
         MVI   INDIC,'N'             INDICATE NOTION CHAIN,TSTBLK PARAM
         BAL   LINK,TSTBLK           CALL ROUTINE TO DECIDE IF A BLOCK
*CAN BE RETURNED TO FREE LIST,IF SO THIS IS CARRIED OUT
**
**
*********************************************************************
         MEND
SUPVAL   01088500:01090700 04/30/75 04/30/75

         MACRO
&LAB     SUPVAL &VALADD,&ATADD
*********************************************************************
**
**
.*(&VALADD)= ADDR OF VALUE LIST CELL CONTAINING VALUE TO BE SUPPRESSED.
.*(&ATADD)= ADDR OF VALUE CELL OF ATTRIBUTE,POINTS TO START OF VALUE
.*LIST.VALUE IS REMOVED,ITS REF COUNT DECREASED AND BLOCK IS CHECKED
.*TO SEE IF IT IS EMPTY AND CAN BE REMOVED.
&LAB     TSTFLAG (OBJUN,0(&VALADD),L1&SYSNDX),(BASIC,0(&VALADD),L1&SYSNC
               DX)
         DECR  &VALADD
*STATISTICS CELL OF NOTION (VALUE OF ATTRIBUTE) UPDATED
L1&SYSNDX MVC   0(4,&VALADD),EMPTY  SUPPRESS VALUE
         L     STADDR,0(&ATADD)      START ADDRESS OF LIST,TSTBLK PARAM
         LR    CELL,&VALADD          ADDR OF CELL EMPTIED,TSTBLK PARAM
         MVI   INDIC,'V'             INDICATE VALUE LIST,TSTBLK PARAM
         BAL   LINK,TSTBLK           CALL ROUTINE TO DECIDE IF A BLOCK IS
*TO BE RETURNED TO FREE LIST,IF SO THIS IS DONE
**
**
*********************************************************************
         MEND
```

```
TSTARG   01092400:01095200 04/30/75 04/30/75

         MACRO
&LAB     TSTARG &ACT,&DISP,&ZONE,&LXIT,&LINK,&SAVE
**********************************************************************
**
**
.*CHOOSE A VALUE OF THE ATTRIBUTE,DISPLACEMENT &DISP IN NOTION &ACT
.*AND STORE IN &ZONE.IF &EXIT IS SPECIFIED IT IS TAKEN IF THE VALUE
.*CHOSEN IS 'NIL'.IF &SAVE IS SPECIFIED,&LINK IS SAVED.
         AIF   ('&SAVE' EQ '').L1
&LAB     ST    &LINK,&SAVE          SAVE RETURN ADDRESS TO CALLING ROUT.
         TM    &DISP.(&ACT),LIST    LIST OF VALUES FOR ARG?
.L2      BZ    L1&SYSNDX            NO->L1&SYSNDX
         LA    ATADD,&DISP.(&ACT)   ADDR OF VALUE  CELL,PARAM FOR RANSUB
         BAL   &LINK,RANSUB         CALL ROUTINE TO CHOOSE A VALUE AT
*RANDOM FROM THE LIST,ADDR OF CELL CHOSEN IS RETURNED IN VALADD
         MVC   &ZONE,0(VALADD)      STORE CHOSEN VALUE
         B     *+10                 SKIP NEXT INSTRUCTION
L1&SYSNDX MVC  &ZONE,&DISP.(&ACT)   STORE ONLY VALUE
         AIF   ('&EXIT' EQ '').END
         CLC   &ZONE,EMPTY          VALUE OF ARGUMENT IS NIL?
         BE    &EXIT                YES->EXIT
.END     MEXIT
.L1      ANOP
&LAB     TM    &DISP.(&ACT),LIST    LIST OF VALUES FOR ARG?
         AGO   .L2
**
**
**********************************************************************
         MEND


TSTFLAG  01095300:01097400 04/30/75 04/30/75

         MACRO
&LAB     TSTFLAG &P1
         LCLA  &N
.*CALLS TO THIS MACRO HAVE A LIST OF PARAMETERS,EACH OF WHICH IS IN
.*TURN A SUBLIST.THE 1ST ELEMENT OF EACH SUBLIST IS THE FLAG CONFIGURAT-
.*ION,THE 2ND IS THE CELL CONCERNED,THE 3RD IS THE EXIT TAKEN WHEN THE
.*TEST RESULT IS POSITIVE.
&N       SETA  1
**********************************************************************
**
**
         AIF   ('&LAB' EQ '').L1
&LAB     EQU   *
.L1      AIF   ('&SYSLIST(&N)' EQ '').END
         TM    &SYSLIST(&N,2),&SYSLIST(&N,1)
         BO    &SYSLIST(&N,3)
&N       SETA  &N+1
         AGO   .L1
**
**
**********************************************************************
.END     MEND
```

```
FNDNAME  01028200:01031300 04/30/75 04/30/75

         MACRO
&LAB     FNDNAME &INPUT,&ZONE,&TAB
         GBLB  &FLAG
*********************************************************
**
**
.*(&INPUT)=START ADDRESS OF A ZONE CONTAINING A NOTION NAME,THE NAME IS
.*ISOLATED USING &TAB AND THEN TRANSFERRED TO &ZONE.
&LAB     MVC   &ZONE.(8),#BLANKS     INITIALISE NAME ZONE
         LA    WKREG1,&INPUT         START ADDRESS
         TRT   0(9,WKREG1),&TAB      FIND 1ST SPACE WITH MAX LNGTH NAME=8
         BC    8,#TRTEXIT            ERROR
         SR    1,WKREG1              LENGTH
         BZ    #TRTEXIT              ERROR
         BCTR  1,0                   LENGTH-1
         STC   1,#TRM&SYSNDX+1       INTO MVC
#TRM&SYSNDX MVC &ZONE,0(WKREG1)      MOVE NAME
         AIF   (&FLAG).L1
.*TREATMENT OF ERRORS IS GENERATED ONCE ONLY
.*ON 1ST CALL TO THIS MACRO.
         B     #TREXIT
*ERROR
#TRTEXIT WRERR #NAMERR               ERROR MESSAGE
#NAMERR  DC    X'0010D4040000'
         DC    'NOTION NAME IN ERROR'
#BLANKS  DC    CL8' '
#TREXIT  DS    0H
&FLAG    SETB  1
**
**
*********************************************************
.L1      MEND


CRNAM    01015000:01016900 04/30/75 04/30/75

         MACRO
&LAB     CRNAM &ADD,&NAMZON,&EXIT
*********************************************************
**
**
.*&ADD CONTAINS A NOTION ADDRESS FROM WHICH A NAME IS CREATED FOR THE
.*ADDRESS EG ADDRESS IS X'1FC4',NAME IS '&1FC4'.THEN CONTROL GOES TO
.*&EXIT.
&LAB     ST    &ADD,#DWORD           CONVERT ADDRESS
         UNPK  &NAMZON+1(5),#DWORD+2(3) TO READABLE FORM
         MVI   &NAMZON,'&'           CREATE
         MVC   &NAMZON+5(3),='   '   NAME
         TR    &NAMZON+1(4),#TAB2-240  TRANS UNPRINTABLE CHARS
*#TAB2-240 GIVES DISPLACEMENTS SO THAT 256 BYTES ARE NOT EXPLICITLY RESE
*-RVED
         B     &EXIT                 EXIT
**
**
*********************************************************
         MEND
```

```
GETADD    01036700:01040600 04/30/75 04/30/75

          MACRO
&LAB      GETADD &NAMZON,&ADD,&INDIC
**********************************************
**
**
.*(&NAMZON)=NAME OF A NOTION WHOSE ADDRESS IS OBTAINED AND RETURNED IN
.*&ADD.&INDIC WILL CONTAIN AN ERROR CODE WHEN THE DICTIONARY IS
.*CONSULTED (SEE ROOT PHASE).
          AIF   ('&NAMZON' NE '#ROOT').L1
&LAB      CLI   &NAMZON,'?'          CURRENT ACTION?
          BNE   #GT&SYSNDX           NO->
          ST    CURRADD,&ADD         ADDRESS OF CURRENT ACTION
          CRNAM CURRADD,&NAMZON,#GT1&SYSNDX NAME FOR CURRENT ACTION
#GT&SYSNDX CLI  &NAMZON,'@'          CREATED NAME?
          AGO   .L2
.L1       ANOP
&LAB      CLI   &NAMZON,'@'          CREATED NAME?
.L2       BNE   #GET&SYSNDX          NO->
          TR    &NAMZON+1(4),#TAB1-193 HEX LETTERS TO HEX NUMBERS BEFORE
*PACKING,#TAB1-193 GIVES DISPLACEMENT SO THAT 256 BYTES ARE NOT
*EXPLICITLY RESERVED
          PACK  #DWORD(3),&NAMZON+1(5) CONVERT
          MVC   &ADD+2(2),#DWORD       TO  INTERNAL FORM
          TR    &NAMZON+1(4),#TAB2-240 RESTORE NAME,#TAB2-240 GIVES
*DISPLACEMENTS OF 256 BYTES WITHOUT RESERVING THE TOTAL EXPLICITLY
          B     #GT1&SYSNDX
#GET&SYSNDX L   1,=V(KEY)
          MVC   0(8,1),&NAMZON       NAME TO KEY
          L     1,=V(&INDIC)
          MVI   0(1),X'01'           SET ERROR CODE
          L     1,=V(DICT)
          L     0,=V(INAREA)
          GETKY (1),(0)              GET RECORD
          LR    1,0                  ADDR OF IOAREA
          MVC   &ADD,R(1)            STORE ADDRESS
#GT1&SYSNDX EQU *
**
**
**********************************************
          MEND
```

```
GETNAM   01043700:01047800 04/30/75 04/30/75

         MACRO
&LAB     GETNAM &ADD,&NAMZON,&NIL
*********************************************************
**
**
.*&ADD ADDRESSES A CELL OF A NOTION,THE MACRO GETS,OR IF NECESSARY
.*CREATES,THE NAME CORRESPONDING TO THE CONTENTS OF THE CELL AND STORES
.*IT IN &NAMZON.&NIL INDICATES THAT THE CELL COULD CONTAIN 'NIL' AND
.*CONTROLS GENERATION OF INSTRUCTIONS TO DEAL WITH THIS CASE.
         AIF   ('&NIL' EQ '').L1
&LAB     CLC   0(4,&ADD),EMPTY      CELL EMPTY
         BNE   #GET&SYSNDX          NO->
         MVC   &NAMZON,=C'NIL       ' SET 'NIL'
         B     #END&SYSNDX          SKIP
#GET&SYSNDX TSTFLAG (BASIC,0(&ADD),#BAS&SYSNDX)
         AGO   .L2
.L1       ANOP
&LAB     TSTFLAG (BASIC,0(&ADD),#BAS&SYSNDX)
.L2      ANOP
*IF BASIC ACTION OCC OR OBJ OF UNIV->BRANCH
         L     WKREG1,0(&ADD)       ADDRESS OF NOTION
         TSTFLAG (NAMFL,0(WKREG1),#NOT&SYSNDX)
*IF NAMED NOTION ->BRANCH,ELSE CREATE NAME
         CRNAM WKREG1,&NAMZON,#END&SYSNDX
*BASIC ACTION OCCURRENCE
#BAS&SYSNDX LA WKREG1,#BASACT       TABLE OF BASIC ACTIONS
         AH    WKREG1,2(&ADD)       +DISPLACMENT
         MVC   &NAMZON,0(WKREG1)    NAME OF BASIC ACTION
         MVC   &NAMZON+4,='    '    + PADDING
         B     #END&SYSNDX
*NAMED NOTION
#NOT&SYSNDX MVC   &NAMZON,40(WKREG1)   NAME TO OUTPUT ZONE
#END&SYSNDX EQU *
**
**
*********************************************************
         MEND
```

```
GETATTR  01040700:01043600 04/30/75 04/30/75

         MACRO
&LAB     GETATTR &ROOT,&NAT,&ADDR
**********************************************************************
**
**
.*CALLS AND LINKS TO THE ENDATSUB ROUTINE IN THE SIMULATOR FROM THE I/O
.*ROUTINES,&ROOT SPECIFIES A NOTION TO BE SEARCHED FOR AN ATTRIBUTE
.*WHOSE NATURE IS SPECIFIED BY &NAT.THE RESULT OF THE SEARCH IS RETURNED
.*IN &ADDR.
*SAVE REGISTERS
&LAB     STM   BR1,BR2,#GETSAVE          BR1,BR2 SAVED
*PARAMETERS
         L     ROOTAD,&ROOT              ROOT & NATURE
         L     NATADD,&NAT               OF ATTRIBUTE
*LINKAGE AND CALL
         L     15,=A(ENDATSUB)           ENTRY  POINT
         L     BR2,=A(BASE+4096) BASE
         L     BR1,=A(BASE)    REGISTERS
*CALL ROUTINE TO FIND ATTRIBUTE SPECIFIED BY ROOT (ROOTAD) AND NATURE
*(NATADD) RESP. RESULT OF SEARCH RETURNED IN ATADD
         BALR  LINK,15
*RETURN
         USING *,LINK
         LM    BR1,BR2,#GETSAVE          RESTORE REGS
         USING #RDATA,BR1
         DROP  LINK
**
**
**********************************************************************
         MEND
```

```
OUTATT   01070200:01073200 04/30/75 04/30/75

         MACRO
&LAB     OUTATT &ADD,&EXIT,&LIST,&TYPE,&REG
*********************************************
**
**
.*OUTPUT ATTRIBUTE ADDRESSED BY &ADD.IF NO ATTRIBUTE THEN &EXIT,IF
.*ATTRIBUTE HAS LIST OF VALUES THEN &LIST.&TYPE INDICATES SPECIAL
.*ATTRIBUTE OR NOT.IF &TYPE= 'SPEC' &REG POINTS INTO A TABLE OF SPECIAL
.*ATTRIBUTE NAMES.
&LAB     CLC   0(4,&ADD),EMPTY     NO ATTR?
         BE    &EXIT               ->EXIT
         AIF   ('&TYPE' NE 'SPEC').L1
         L     WKREG2,0(&REG)      SPATTAB ENTRY
         MVC   #NATURE(8),40(WKREG2) SPECIAL ATTRIBUTE NAME
         AGO   .L2
.L1      GETNAM &ADD,#NATURE       GET NAME OF NATURE
.L2      LA    LENG,8              SET LENGTH
         PUTNAM PTR,#NATURE,LENG
*PUT NATURE OF ATTRIBUTE IN OUTPUT LINE
         AIF   ('&TYPE' NE 'NORM').L3
         LA    &ADD,4(&ADD)        VALUE CELL
.L3      TSLIST &ADD,&LIST         IF LIST OF VALUES->BRANCH
         GETNAM &ADD,#NOTION,NIL   GET NAME OF VALUE
         PUTNAM PTR,#NOTION,LENG
*PUT VALUE OF ATTRIBUTE IN OUTPUT LINE
         PUTLINE #OUTLINE,PTR,#TERMD
*OUTPUT LINE TO SYSOUT FILE
**
**
*********************************************
         MEND


ERRCALL  01025000:01026300 04/30/75 04/30/75

         MACRO
&LAB     ERRCALL &MESS
*********************************************
**
**
.*WROUT PRINTS THE MESSAGE WHOSE ADDRESS IS &MESS THEN TERMINATES THE
.*PROGRAM.
&LAB     LA    1,WRPARAM           PARAM LIST
         MVC   1(3,1),=AL3(&MESS)  MESSAGE ADDRESS
         P     WROUT               TO ERROR MESSAGE OUTPUT
**
**
*********************************************
         MEND
```

```
PUTLINE  01104900:01106600 05/02/75 05/02/75

         MACRO
&LAB     PUTLINE &LINE,&POS,&ERR
**********************************************************************
**
**
.*OUTPUTS LINE STARTING FROM &LINE UP TO POSITION IN &POS.&ERR IS EXIT
.*IN CASE OF WROUT ERROR.
&LAB     LA    WKREG1,&LINE          LENGTH TO
         SR    &POS,WKREG1           BE OUTPUT
         STH   &POS,&LINE
         WROUT &LINE,&ERR            OUTPUT LINE
         MVI   &LINE+5,' '           REINITIALISE
         MVC   &LINE+6(114),&LINE+5  LINE
         LA    &POS,&LINE+5          POINT TO START OF NEW LINE
**
**
**********************************************************************
         MEND

PUTNAM   01076400:01077900 04/30/75 04/30/75

         MACRO
&LAB     PUTNAM &POS,&NAMZON,&LENG
**********************************************************************
**
**
.*THE NOTION NAME IN &NAMZON WITH LENGTH (&LENG) IS STORED IN THE OUTPUT
.*LINE AT &POS.
&LAB     BCTR  &LENG,0               LENGTH-1
         STC   &LENG,#PN&SYSNDX+1    INTO MOVE
#PN&SYSNDX MVC   0(&,&POS),&NAMZON   NAME TO LINE
         AP    &POS,&LENG            POINT TO NEXT FREE SPACE
         LA    &POS,2(&POS)          IN OUTPUT LINE
**
**
**********************************************************************
         MEND
```

```
LENGTH   01101900:01104800 05/02/75 05/02/75

         MACRO
&LAB     LENGTH &NAMZON,&LENG,&NROOM,&TAB,&POS,&LINE
*********************************************************
**
**
.*USING &TAB THE NOTION IN &NAMZON IS ISOLATED.ITS LENGTH IS CALCULATED
.*AND RETURNED IN &LENG.&POS POINTS TO THE CURRENT POSITION IN &LINE
.*(OUTPUT ZONE).IF &NROOM IS SPECIFIED IT IS SET TO INDICATE WHETHER
.*THERE IS ROOM FOR THE NAME IN &LINE.
&LAB     SLR   1,1                 CLEAR REG 1
         TRT &NAMZON,&TAB          FIND END OF NAME
         BC    8,#LEN&SYSNDX       NOT FOUND->
         LA    WKREG1,&NAMZON      FINISH
         SR    1,WKREG1            - START
#STL&SYSNDX LR &LENG,1             =LENGTH
         AIF   ('&NROOM' EQ '').L1
         LR    WKREG2,&POS         ROOM IN
         AR    WKREG2,1            LINE FOR
         C     WKREG2,=A(&LINE+114) NAME?
         BHR   #END&SYSNDX         YES->
         MVI   &NROOM,X'01'        NO->SET FLAG
.L1      ANOP
         B     #END&SYSNDX
#LEN&SYSNDX LA  1,8                 LENGTH=8
         B     #STL&SYSNDX
#END&SYSNDX EQU *
**
**
*********************************************************
         MEND


RDACHK   01078000:01079100 04/30/75 04/30/75

         MACRO
&LAB     RDACHK &ZONE,&LEN,&TEST,&EXIT
*********************************************************
**
**
.*CHECKS LENGTH OF MESSAGE IN &ZONE IS OK.
&LAB     CLC   &ZONE.(2),=XL2'&LEN'  LEN OK?
         B&TEST &EXIT
**
**
*********************************************************
         MEND
```

N SOURCE

```
          TITLE '3-GRAPH SIMULATOR,ROOT PHASE'
*THE 3-GRAPH SIMULATOR IS DESIGNED WITH AN OVERLAY STRUCTURE TO LEAVE
*THE MAXIMUM AMOUNT OF SPACE FOR THE 3-GRAPH ITSELF.THE ROOT PHASE
*CONSISTS OF THE FOLLOWING CODE AND THE INITIAL 3-GRAPH.IT LOADS
*FIRST THE INITIALISER PHASE DIRECTLY BEHIND THE GRAPH AND THEN THE
*SIMULATOR ON TOP OF THE INITIALISER.
*THE ROOT PHASE ALSO CONTAINS THE DICTIONARY FILE DEFINITION AND ERROR
*ROUTINES
ROOTPH    START
          PRINT NOGEN
          EXTRN INITPP                    INITIALISER ENTRY POINT
          EXTRN SIMULATE                  SIMULATOR ENTRY POINT
          EXTRN #RDATA                    RETURN IN I/O ROUTINES
          ENTRY PAGETAB                   TABLE OF PAGES ALLOCATED IN 3-GRAPH
          ENTRY LDSIMUL,DICT,FILERR,KEY,#NOFIND
BR2       EQU   4
          BALR  BR2,0                     LOAD BASE REGISTER
          USING *,BR2
          LPOV  INIT,INITPR               LOAD INITIALISER PHASE
*RETURN FROM INITIALISER PHASE TO HERE
LDSIMUL   LPOV  SIMUL,SIMULATE            LOAD SIMULATOR PROPER
          TERM
*DICTIONARY FILE DEFINITION
DICT      FCB   LINK=SYNDICT,FCBTYPE=ISAM,RECFORM=F,RECSIZE=12,BLKSIZE=
                TD,EXIT=FILERR,KEYLEN=8,OPEN=OUTIN,KEYARG=KEY,DUPEKY=YE
*ERROR EXIT LIST AND ROUTINES
FILERR    EXLST NOFIND=NOFIND,DUPEKY=DUPEKY,OPENER=OPENERR,USERERR=USER
                ,COMMON=TERMD,NOSPACE=NOSPACE,ISPERR=NOSPACE
*
NOFIND    BALR  2,0                       NEW BASE REGISTER(TEMPORARY)
          USING *,2
          DROP  4
          CLI   #NOFIND,X'01'             ERROR TYPE 1,I/O ROUTINES

          BE    IONOFIND
          CLI   #NOFIND,X'02'             ERROR TYPE 2,NAMBLK(SIMULATOR)
          BE    ELIMNOFI
          CLI   #NOFIND,X'03'             ERROR TYPE 3,SIMULATOR MAIN
          BNE   TERMD
MAINNOFI  WROUT NOFMESS1,TERMD
          L     1,=A(SIMULATE)
          BR    1                         RETURN
ELIMNOFI  ERRCALL NOFMESS2                NO RETURN
IONOFIND  WROUT NOFMESS3,TERMD
          L     1,=A(#RDATA)
          BR    1                         RETURN
NOSPACE   ERRCALL NOSPMESS                NO RETURN
OPENERR   ERRCALL OPENMESS                NO RETURN
USERR     ERRCALL USERMESS                NO RETURN
DUPEKY    ERRCALL DUPKMESS                NO RETURN
*
TERMD     TERMD
```

```
M SOURCE

   WROUT    WROUT (1)                    WRITE ERR MESS
   ERR      TERMD
   *

   WRPARAM  DS    A                      PARAM LIST FOR WROUT
            DS    C
            DC    AL3(ERR)

   PAGETAB  DS    256C                   PAGE TABLE
   KEY      DC    CL8' '                 KEY FOR DICTIONARY
   #NOFIND  DC    X'00'                  ERROR CODE FOR DICTIONARY
   *ERROR MESSAGES
   NOSPMESS DC    X'001E404041'
            DC    'NO ROOM TO EXTEND FILE'
   OPENMESS DC    X'001D404041'
            DC    'OPEN ERROR ON DICT,TERMD'

   USERMESS DC    X'00224U4041'
            DC    'IMPROPER ACTION ON DICT,TERMD'
   DUPKMESS DC    X'001F404000'

            DC    'DUPLICATE KEY FOUND,TERMD'
   NOFMESS1 DC    X'001F404000'
            DC    'NOTION SPECIFIED NOT FOUND'

   NOFMESS2 DC    X'002D404000'
            DC    'NOTION TO BE REMOVED DOESN''T EXIST,TERMD'
   NOFMESS3 DC    X'001A404000'

            DC    'NOTION NAME NOT FOUND'



            END   ROOTPH


00, 000 MNOTES

2000 ASSEMBLER VERS. 0021
```

```
M SOURCE

          TITLE '3-GRAPH SIMULATOR,OVERLAY 1'
*DESCRIPTION:INITIALISER PHASE
*THIS IS THE FIRST PART OF THE SIMULATION OF A 3-GRAPH.THE REFERENCE
*COUNTS OF ALL NOTIONS ARE INITIALISED TO THE NUMBER OF TIMES EACH ONE
*IS EITHER NATURE OR VALUE OF A NOTION.PAGETAB-A TABLE INDICATING THE
*SECTION TO WHICH EACH 3-GRAPH PAGE BELONGS - IS ALSO INITIALISED.
*IF REQUESTED BY THE USER A DICTIONARY OF NOTION NAMES AND ADDRESSES I
*CREATED
INITPR  CSECT
          PRINT NOGEN
          EXTRN TRGR1                    START OF 3-GRAPH
          EXTRN LDSIMUL                  RETURN TO ROOT
          EXTRN PAGETAB                  TABLE OF 3-GRAPH PAGES IN USE
*REGISTER ALLOCATIONS
CURRADD   EQU   12                       CURRENT ADDRESS IN PAGE
CURRPAGE  EQU   6                        CURRENT PAGE IN 3-GRAPH
LINK      EQU   13                       LINKS BETWEEN ROUTINES
WKREG1    EQU   14                       WORK
WKREG2    EQU   15                       REGISTERS
BR1       EQU   3                        BASE REGISTER
SAVEREG   EQU   7                        SAVE ZONE
*SYMBOLIC IMMEDIATE OPERANDS
LIST      EQU   X'40'                    FLAGS FOR LIST OF VALUES
NAMFL     EQU   X'80'                    FLAGS FOR NAMED NOTION
**
BASE      BALR  BR1,0                    LOAD
          USING *,BR1                   BASE REG
**
*USER IS ASKED IF A DICTIONARY IS TO BE CREATED
QUERY     WROUT QUERYDIC,ERROR
          RDATA REPLYDIC,ERROR
          RDACHK REPLYDIC,7,11,QUERY CHECK REPLY LENGTH OK
          CLC   REPLYDIC+4(2),='NO' NO->

          BE    DICTEX                   DICTEX
          CLC   REPLYDIC+4(3),='YES'  REPLY NOT 'YES' OR 'NO'
          BNE   QUERY                    ->LOOP
          L     1,=V(DICT)
          OPEN  (1),OUTIN               OPEN FILE TO CREATE
          MVI   DICTSW+1,X'F0'           SET SWITCH,BRANCH TO CREATE DICT
          B     INIT
DICTEX    L     1,=V(DICT)
          OPEN  (1),INOUT               OPEN EXISTING DICT
```

M SOURCE

```
**
*TO PREVENT EXCESSIVE PAGING THE 3-GRAPH IS DIVIDED INTO GROUPS OF
*CONTIGUOUS PAGES AND THE USER IS ASKED TO DETERMINE THE SIZE OF A GROU
*THIS IS THE PARAMETER K WHERE.2.LE.K.LE TOTAL NO OF PAGES (DETERMINED
*AT LINKAGE).THE REPLY IS 4 DIGITS INCLUDING LEADING ZEROES
INIT     WROUT REQ1,ERROP          ASK FOR K,NO OF PAGES IN A GROUP
         RDATA INFO1,ERROR
         RDACHK INFO1,8,NE,INIT    CHECK REPLY LENGTH OK
**
*INITIALISATION OF PARAMETERS-FIRST PAGE,LAST PAGE,SIZE OF GROUP,NO OF
*GROUPS,START ADDRESS
         PACK DWORD+5(3),INFO1+4(4) CONVERT TO
         CVB   11,DWORD             USEFUL FORM
         BCTR  11,0                 K-1
         ST    11,KM1               STORE (K-1) FOR LATER USE
         L     11,=A(INITPR)        START ADDR OF SIMULATOR
         SRL   11,12                ISOLATE PAGE NO
         BCTR  11,0

         STC   11,LSTPAGE           NO OF LAST PAGE OF 3-GRAPH
         L     11,=A(TRGR1)         ADDRESS BEGINNING OF 3-GRAPH
         SRL   11,12
         STC   11,FSTPAGE           NO OF 1ST PAGE OF 3-GRAPH
         SLL   11,12
         LR    CURRADD,11           ADDRESS OF 1ST CELL,1ST PAGE
         XR    11,11                PREPARE TO
         IC    11,FSTPAGE           CALCULATE
         XR    10,10                NO OF PAGES TO
         IC    10,LSTPAGE           BE INITIALISED
         SR    10,11                LSTPAGE-FSTPAGE
         C     10,=F'0'             1 PAGE?
         BNE   INITLOOP             NO->INITLOOP
         MVI   ONEPAGSW+1,X'F0'     SET SWITCH TO SKIP UNNECESSARY
*                                   ROUTINES
INITLOOP SRDL  10,32                PREPARE FOR DIVISION
         L     9,KM1                K-1

         DR    10,9                 (LSTPAGE-FSTPAGE)/(K-1)
         C     10,=F'0'             REMAINDER=0?
         BE    INITCONT             YES
         LA    11,1(11)             QUOTIENT+1
INITCONT ST    11,LOOPCONT          NO OF PAGE GROUPS,(K-1) PAGES, TO
*                                   COVER 3-GRAPH
```

M SOURCE

```
**
DICTSW   NOP   CREDICT              WHEN RESET ROUTINE TO CREATE DICT
*                                   IS INVOKED
INITSCAN BAL   LINK,PAGTABSB        CALL ROUTINE TO INITIALISE PAGETAB
*                                   ENTRY FOR CURRENT PAGE
         BAL   LINK,SCANPAGE        CALL ROUTINE TO SCAN PAGE,UPDATE
*INTERNAL REFERENCES AND FILL IN REFTAB,OUTPUT PARAMETER:CURRPAGE
**
*WHEN A PAGE HAS BEEN TREATED BY SCANPAGE THE REMAINDER OF THE 3-GRAPH
*IS SCANNED IN GROUPS OF K-1 PAGES UPDATING REFERENCE COUNTS OF NOTIONS
*REFERRED TO IN PAGE JUST DEALT WITH.
**
ONEPAGSW NOP FINISH                 SWITCH,IF ONLY 1 PAGE THEN REST OF
*                                   PROGRAM UNNECESSARY
         L     11,LOOPCONT          LOOP CONTROL
         LA    CURRPAGE,1(CURRPAGE) PAGE NO OF 1ST
CALCT1   STC   CURRPAGE,PAGET1       PAGE OF GROUP,TRTINIT PARAM
         BAL   LINK,TRTINIT         CALL ROUTINE TO INITIALISE TRT TABLE
*                                   FOR CURRENT GROUP OF PAGES
         BAL   LINK,TRTROUT         CALL ROUTINE TO UPDATE REFERENCE
*                                   COUNTS INDICATED BY REFTAB

         IC    CURRPAGE,PAGET1      LAST PAGE OF GROUP +1
         STC   CURRPAGE,WORK
         CLC   WORK,LSTPAGE          WRAPAROUND NEEDED?
         BNH   CALCCONT              NO
         IC    CURRPAGE,FSTPAGE      YES
CALCCONT BCT   11,CALCT1
         SRL   CURRADD,12             ADDRESS OF
         LA    CURRADD,1(CURRADD)    NEXT PAGE
         STC   CURRADD,WORK          LAST PAGE
         CLC   WORK,LSTPAGE          HAS BEEN INITIALISED?
         BH    FINISH                YES->EXIT
         SLL   CURRADD,12            NEXT PAGE
         B     INITSCAN              LOOP

FINISH   L     LINK,ALDSIMUL         RETURN ADDRESS IN ROOT PHASE
         BR    LINK                  RETURN TO ROOT PHASE
```

M SOURCE

```
*INPUT PARAMETERS
*CURRADD CONTAINS START ADDRESS OF CURRENT PAGE
*SECTNO CONTAINS CURRENT SECTION NO
*OUTPUT PARAMETERS
*SECTNO CONTAINS SECTION NO FOR NEXT 3-GRAPH PAGE
*DESCRIPTION:PAGTABSB ROUTINE
*THIS ROUTINE INITIALISES PAGETAB FOR THE 3-GRAPH.PAGETAB CONTAINS A
*1-BYTE ENTRY FOR EACH PAGE OF THE 3-GRAPH.EACH CONTAINS THE NO OF THE
*SECTION TO WHICH THE PAGE BELONGS AND AN INDICATION OF WHETHER THE
*PAGE IS FULL OR NOT.THE ENTRY IS (N*2)+1 IF A PAGE IS NOT FULL,(N*2)+0
*IF THE PAGE IS FULL (N IS THE SECTION NO : THE PAGE BELONGS TO THE
*NTH SECTION INCLUDED IN THIS 3-GRAPH).WHEN A PAGE IS NOT FULL IT IS
*ASSUMED,BECAUSE OF THE WAY THE 3-GRAPH IS CREATED,THAT THE END OF A
*SECTION HAS BEEN REACHED,SO THE SECTION NO IS INCREMENTED FOR THE NEXT
*PAGE.IF THE FREE LIST FLAG BITS ARE X'FF' THEN THE END OF A SECTION
*COINCIDES WITH A FULL PAGE.
*
*NOTE:THE SECTION NO. N IS NOT THE SECTION NO. GIVEN BY THE USER TO HIS
*3-GRAPH DESCRIPTIONS.
*
*CALLS:0
*CALLED BY:INITIALISER MAIN ROUTINE
PAGTABSB STM   9,11,PAGTSAVE       SAVE WORK REG CONTENTS
         LR    11,CURRADD          CALCULATE ADDRESS OF
         SRL   11,12               PAGE ENTRY
         A     11,APAGETAB         IN PAGETAB
         LR    10,CURRADD          ADDRESS OF FREE
         C     10,FRLSTAD          LIST HEADER
         CLC   0(4,10),EMPTY       PAGE FULL?
         BE    PAGTFULL            YES
         XR    9,9
         IC    9,SECTNO            CURRENT SECTION NO
         SLL   9,1                 * BY 2
         TM    0(10),X'FF'         END OF SECTION COINCIDES WITH FULL
*                                  PAGE
         BO    PAGTST              YES
         LA    9,1(9)              + 1 INDICATES PAGE NOT FULL
PAGTST   STC   9,0(11)             STORE ENTRY IN PAGETAB
         SRL   9,1                 DIVIDE BY 2
         LA    9,1(9)              SECTION NO INCREASED BY 1
         STC   9,SECTNO
         NI    0(10),X'0F'         RESET FLAG BITS IF NECESSARY
PAGTEXIT LM    9,11,PAGTSAVE       RESTORE REGS
         BR    LINK                EXIT
PAGTFULL XR    9,9
         IC    9,SECTNO            CURRENT SECTION NO
         SLL   9,1                 * BY 2
*                                  + 0 INDICATES PAGE FULL
         STC   9,0(11)             STORE ENTRY IN PAGETAB
         B     PAGTEXIT            EXIT
```

M SOURCE

```
*INPUT PARAMETERS
*CURRADD CONTAINS START ADDRESS OF PAGE TO BE SCANNED
*OUTPUT PARAMETERS
*CURRADD IS UNCHANGED
*CURRPAGE CONTAINS THE NO OF THE PAGE JUST SCANNED
*REFTAB HAS BEEN INITIALISED FOR THE CURRENT PAGE
*DESCRIPTION:SCANPAGE ROUTINE
*AFTER CURRPAGE HAS BEEN INITIALISED THE CURRENT PAGE IS SCANNED.CELLS
*NOT CONTAINING REFERENCES TO 3-GRAPH NOTIONS ARE SKIPPED,CELLS THAT
*CONTAIN A REFERENCE CAUSE ONE OF TWO ACTIONS-IF THE REFERENCE IS TO A
*NOTION IN THE CURRENT PAGE THIS NOTION'S REFERENCE COUNT IS INCREMENT-
*-ED.IF THE NOTION REFERENCED IS IN ANOTHER PAGE AN ENTRY IS MADE IN
*REFTAB.THE CURRENT CELL ADDRESS IS TURNED INTO AN OFFSET IN REFTAB,
*THE PAGE NO OF THE NOTION REFERENCED IS ENTERED INTO REFTAB,AT THAT
*POINT.
*CALLS:0
*CALLED BY:INITIALISER MAIN ROUTINE
SCANPAGE STM   7,11,SCANSAVE        WORK REGS

         LR    CURRPAGE,CURRADD     INITIALISE
         SLL   CURRPAGE,12          CURRPAGE
         SRL   CURRPAGE,24          (CURRENT PAGE NO)
*LIMIT SCAN (EXCLUDE FREELIST)
         LR    11,CURRADD           ADDRESS BEGINNING OF PAGE
         O     11,FRLSTAD           FREE LIST HEADER ADDRESS
         CLC   0(4,11),EMPTY        PAGE FULL?
         BE    SCANPCEU             YES
         L     10,0(11)
         SH    10,=H'4'             ADDRESS OF LAST CELL
         ST    10,ENDADDR           TO BE SCANNED
*SCAN PAGE
         LR    11,CURRADD           ADDRESS OF 1ST CELL IN PAGE
SCANTEST LA    10,4                 4 CELLS/BLOCK:LOOP CONTROL
         TSTFLAG (NAMFL,0(11),SETSW) IF NAMED NOTION->SETSW
SCANLOOP CLI   0(11),X'0F'          IF NOT A NOTION OR

         BNE   SCANNEXT
         CLC   0(4,11),EMPTY        IF EMPTY CELL ->SKIP
         BE    SCANNEXT
*INCREMENT REF COUNT OR PLACE ENTRY IN REFTAB
         L     9,0(11)              NOTION ADDRESSED
         SLL   9,12                 ISOLATE
         SRL   9,24                 PAGE NO
         CR    9,CURRPAGE           REFERENCE IN CURRENT PAGE?
         BNE   SCANTAB              NO->
         INCR  0(11)                INCREMENT REFERENCE COUNT
         B     SCANNEXT
SCANTAB  STC   9,REFNO              SAVE PAGE NO OF REF
         LR    8,11                 INITIALISE
         SR    8,CURRADD            REL ADDR IN PAGE
         LR    WKREG1,8
         SRDL  8,32                 FOR DIVISION
```

M SOURCE

```
          D      8,=F'4'                 DIVIDE BY CELL LENGTH
          SRDL   WKREG1,32               INITIALISE FOR DIVISION
          D      WKREG1,=F'20'           DIVIDE BY BLOCK LENGTH
          SR     9,WKREG2                RESULT IS CELL NO
*CELL NO=RELATIVE NO OF CELL-NO OF POINTER CELLS PRECEDING IT
*RELATIVE NO OF CELL=REL ADDR OF CELL/CELL LENGTH
*NO OF POINTER CELLS=NO OF BLOCKS PRECEDING CELL
*NO OF BLOCKS=REL ADDR OF CELL/BLOCK LENGTH
          A      9,=A(REFTAB)            RESULT IS ADDR OF ENTRY FOR CELL
          MVC    0(1,9),REFNO            STORE PAGE NO FOR EXT REF IN
*                                        ENTRY CORR TO CELL
*CONTROL SCAN

SCANNEXT  LA     11,4(11)                NEXT CELL
          BCT    10,SCANLOOP
          BCTR   7,0                     END OF BLOCK:
          LTR    7,7                     IF (7) = 0 ->SKIPBL
          BZ     SKIPBL
SCANEND   C      11,ENDADDR              END OF PAGE?
          BNL    SCANTERM                YES
          LA     11,4(11)                ADDR 1ST CELL NEXT BLOCK
          B      SCANTEST
SCANPGFU  SH     11,=H'4'                SCAN LIMIT IS
          ST     11,ENDADDR              END OF PAGE
          LR     11,CURRADD              START ADDRESS
          B      SCANTEST
**
SKIPBL    LA     11,20(11)               SKIP NAME BLOCK
          B      SCANEND
**

SETSW     LA     7,2                     AFTER 2 BLOCKS,NAME BLOCK TO SKIP
          B      SCANNEXT
**
SCANTERM  LM     7,11,SCANSAVE           RESTORE WORK REGS
          BR     LINK                    EXIT
```

```
M SOURCE

*INPUT PARAMETERS
*REFTAB CONTAINS ENTRIES FOR EXTERNAL REFS IN CURRENT PAGE(SCANPAGE)
*TRTTAB HAS BEEN INITIALISED FOR A GROUP OF PAGES IN THE 3-GRAPH-TRTINIT
*OUTPUT PARAMETERS
*REFTAB & TRTTAB ARE CLEARED
*DESCRIPTION:TRTROUT ROUTINE
*USING A TRT INSTRUCTION REFTAB IS SCANNED FOR REFERENCES TO NOTIONS IN
*A GROUP OF PAGES SPECIFIED BY TRTTAB.WHEN A REFERENCE IS FOUND THE
*ADDRESS OF THE ENTRY IN REFTAB IS USED TO GET THE ADDRESS OF THE CELL
*CONTAINING THE REFERENCE (THE REVERSE OF WHAT HAPPENS IN SCANPAGE)
*THEN THE NOTION REFERENCED IS UPDATED (REF COUNT INCREMENTED).
*BECAUSE REFTAB IS 816 BYTES LONG AND THE MAX LENGTH FOR A TRT IS 256
*THE SCAN IS EFFECTIVELY DIVIDED INTO 3 SCANS OF 256 BYTES AND ONE OF
*48 BYTES
*CALLS:0
*CALLED BY:INITIALISER MAIN ROUTINE
TRTROUT  STM   9,11,TRTSAVE      WORK REGS
         MVI   TRT+1,X'FF'       INITIALISE TRT LENGTH
         LA    1,REFTAB          INITIALISE TRT ADDRESS
         LA    11,1              NO OF TIMES TRT EXECUTED OVER 256B
         LA    9,REFTAB          START ADDRESS OF CURRENT TRT
TRT      TRT   0(256,1),TRTTAB
         BC    8,TRTREIN         NO REFERENCES->TRTREIN
         LR    10,1              RECOVER
         S     10,=A(REFTAB)     CELL NO
         LR    WKREG1,10         INITIALISE
         SRDL  WKREG1,32         FOR DIVISION

         D     WKREG1,=F'4'      DIVIDE BY NO OF CELLS/BLOCK
         AR    10,WKREG2         ADD BLOCK NO
         MH    10,=H'4'          * BY CELL LENGTH
         AR    10,CURRADD        +PAGE ADDRESS GIVES CELL ADDRESS
         INCR  0(10)             INCREMENT REFERENCE COUNT
         MVI   0(1),X'FF'        REINITIALISE REFTAB ENTRY
         LR    WKREG1,1
         SR    WKREG1,9          LENGTH OF TRT SO FAR

         CH    WKREG1,=H'255'    MAX LENGTH?
         BE    TRTREIN
         LH    WKREG2,=H'254'    UPDATE
         SR    WKREG2,WKREG1     LENGTH
         STC   WKREG2,TRT+1      OF TRT
         LA    1,1(1)            UPDATE START ADDRESS
         B     TRT
TRTREIN  CH    11,=H'3'          REFTAB EXHAUSTED?
         BH    TRTEXIT           YES
         LA    1,REFTAB          NEW START ADDRESS
         LH    WKREG2,=H'256'    =A(REFTAB)
         MR    WKREG1,11         + N*256
         AR    1,WKREG2          N LE 3
         CH    11,=H'3'          LAST TRT?
         BE    TRTMVI            YES
```

M SOURCE

```
            MVI   TRT+1,X'FF'          REINITIALISE LENGTH
TRTRINCN    LA    11,1(11)             INCREASE N
            LR    9,1                  NEW START ADDRESS OF CURRENT TRT
            B     TRT
TRTMVI      MVI   TRT+1,X'2F'          LENGTH=48
            B     TRTRINCN
TRTEXIT     MVI   TRTTAB,X'00'         REINITIALISE
            MVC   TRTTAB+1(255),TRTTAB TRTTAB
            LM    9,11,TRTSAVE         WORK REGS
            BR    LINK                 EXIT



*INPUT PARAMETERS
*PAGET1 CONTAINS THE NO OF THE FIRST PAGE IN THE GROUP
*KM1 CONTAINS THE NO OF PAGES IN THE GROUP
*OUTPUT PARAMETERS
*PAGET1 CONTAINS THE NO OF THE LAST PAGE IN THE GROUP +1
*DESCRIPTION:TRTINIT ROUTINE
*TRTROUT HAS TO SCAN REFTAB FOR REFERENCES TO NOTIONS IN PAGES OF THE
*3-GRAPH OTHER THAN THE CURRENT ONE.IN ORDER TO AVOID EXCESSIVE PAGING
*THE SCAN IS LIMITED TO A CONTIGUOUS GROUP OF PAGES.THE SIZE OF THE
*GROUP IS GIVEN BY KM1 AND THE FIRST PAGE BY PAGET1.TRTTAB IS INITALISE
*WITH AN ENTRY OF X'FF' FOR EACH PAGE IN THE GROUP
*CALLS:0
*CALLED BY:INITIALISER MAIN ROUTINE
TRTINIT     STM   9,11,TRTISAVE        WORK REGS
            XR    11,11
            IC    11,PAGET1            FIRST PAGE IN GROUP
            L     10,KM1               K-1,NO OF PAGES IN GROUP,LOOPCONTROL
TRTICHK     STC   11,WORK
            CLC   WORK,LSTPAGE         NEXT PAGE NO>LAST PAGE NO?
            BNH   TRTILOOP             NO
            IC    11,FSTPAGE           WRAPAROUND ON PAGE NOS
TRTILOOP    LA    9,TRTTAB             ADDRESS OF ENTRY
            AR    9,11                 IN TRTTAB
            MVI   0(9),X'FF'           INITIALISE ENTRY
            LA    11,1(11)             NEXT PAGE NO
            BCT   10,TRTICHK

            STC   11,PAGET1            RETURN NEXT PAGE NO
            LM    9,11,TRTISAVE        WORK REGS RESTORED
            BR    LINK                 EXIT
```

M SOURCE

```
*INPUT PARAMETERS
*CURRADD CONTAINS THE ADDRESS OF THE START OF THE 3-GRAPH.
*DICT FILE HAS BEEN OPENED OUTIN.
*OUTPUT PARAMETERS
*CURRADD IS UNCHANGED
*GENERAL DESCRIPTION:CREDICT ROUTINE
*THE GRAPH IS SCANNED TO LOCATE NOTIONS WITH NAMES.WHEN ONE IS LOCATED
*ITS NAME BLOCK (NAME AND ADDRESS) IS TRANSFERRED TO THE DICT FILE
*CALLS:0
*CALLED BY:INITIALISER MAIN ROUTINE
CREDICT  LR    SAVEREG,CURRADD      SAVE CONTENTS CURRADD
*INITIALISE END-OF-PAGE LIMIT
LIMIT    LR    WKREG1,CURRADD
         O     CURRADD,FRLSTAD      FREE LIST HEADER
         CLC   0(4,CURRADD),EMPTY   PAGE FULL?
         BE    FULLPAGE             YES->

         CLI   0(CURRADD),X'FF'     END OF SECTION & FULL PAGE?
         BE    FULLPAGE             YES->
         MVC   ENDADDR,0(CURRADD)   LIMIT SCAN
         B     LOCENTRY
FULLPAGE ST    CURRADD,ENDADDR      SCAN WHOLE PAGE
*LOCATE A DICTIONARY ENTRY
LOCENTRY LR    CURRADD,WKREG1
TSTFLAGS CLI   0(CURRADD),NAMFL     NOTION NAMED?
         BNE   NONAME               NO->NONAME
         LA    CURRADD,40(CURRADD)  GET
         MVC   ZONOUT,0(CURRADD)    ENTRY
*WRITE DICTIONARY ENTRY
         L     1,=V(DICT)
         INSRT (1),ZONOUT
NONAME   LA    CURRADD,20(CURRADD)  NEXT BLOCK
         C     CURRADD,ENDADDR      END OF PAGE?
         BNE   TSTFLAGS             NO->TSTFLAGS
*END OF A PAGE
         PAGNO CURRADD,WORK         CURRENT PAGE NO
         CLC   LSTPAGE,WORK         =LAST PAGE NO?
         BE    ENDOFGR              YES->END OF GRAPH
         N     CURRADD,SETZEROS     START ADDRESS
         AH    CURRADD,PGPLUS1      OF NEXT PAGE
         B     LIMIT
ENDOFGR  LR    CURRADD,SAVEREG      RESTORE CURRADD
         B     INITSCAN             RETURN TO INITIALISER
```

```
* M SOURCE

    *ERROR EXITS
    ERROR    TERMD
    *IOAREA

    ZONOUT   DS     0CL12
    KEY      DS     CL8
    DATA     DS     CL4
    *WORK ZONES
    ENDADDR  DS     F
    LOOPCONT DS     F
    DWORD    DS     0D
             DC     PL8'0'
    TRTTAB   DC     256X'00'



    REFNO    DS     C
    WORK     DS     C
    SAVE     DC     X'00'

             DS     0F
    SETZEROS DC     X'FFFFF000'
    PGPLUS1  DC     X'1000'
    *REGISTER SAVE AREAS

    SCANSAVE DS     5F
    PAGTSAVE EQU    SCANSAVE
    TRT1SAVE EQU    SCANSAVE
    TRTRSAVE EQU    SCANSAVE
    *SPECIAL ADDRESSES IN 3-GRAPH
    FRLSTAD  DS     0F
             DC     XL4'FFD'
    EMPTY    DC     X'FFFFFFE'
    *EXTERNAL ADDRESSES
    APAGETAB DC     A(PAGETAB)
    ALDSIMUL DC     A(LDSIMUL)
    *TABLE OF REFERENCES EXTERNAL TO A GIVEN PAGE
    REFTAB   DC     816X'FF'
    *PARAMETERS GOVERNING EXECUTION OF INITIALISER

    FSTPAGE  DS     C
    LSTPAGE  DS     C
    PAGET1   DS     C
    SECTNO   DC     X'01'

    KM1      DS     F
    *MESSAGES TO USER AND REPLIES
    INFO1    DS     CL8
    REQ1     DC     X'002A004040'
             DC     'HOW MANY PAGES IN A PAGE GROUP(NNNN)'
    QUERYDIC DC     X'002E404041'
             DC     'DO YOU WANT A DICTIONARY CREATED (YES/NO)'

    REPLYDIC EQU    INFO1

             END    INITPP
```

M SOURCE

```
          TITLE '3-GRAPH SIMULATOR PROGRAM,OVERLAY 2'
SIMULATE  CSECT
          PRINT NOGEN
          EXTRN OCCDE,SUC1,SUC2       THE SPECIAL NOTIONS OF THE 3-GRAPH
          EXTRN ARG1,ARG2,ARG3,ARG4 WHOSE ADDRESSES WILL BE NEEDED
          EXTRN PAGETAB               TABLE OF 3-GRAPH PAGES IN USE
          EXTRN #ATHALT,#USERINT      LINK TO I/O ROUTINES
          ENTRY NEXTACT,FNDATSUB,INAREA,SPATTAB,BASE FOR I/O ROUTINES
*REGISTER ALLOCATIONS
BR1       EQU   3                     BASE REGISTERS FOR THE
BR2       EQU   4                     SIMULATION ROUTINES
ATADD     EQU   5                     TO CONTAIN ATTRIBUTE ADDRESSES
STADDR    EQU   5                     START ADDRESSES OF CHAINS OF BLOCKS
VALADD    EQU   6                     VALUE CELL ADDRESSES
NATADD    EQU   6                     ATTRIBUTE NATURES,ADDRESS OF NOTION
CELL      EQU   6                     CELL ADDRESSES
NEWADD    EQU   8                     ADDRESS OF NEWLY ALLOCATED BLOCKS
ROOTAD    EQU   7                     ADDRESSES OF ROOT NOTIONS
CURRADD   EQU   12                    ADDRESS OF CURRENT ACTION OCCURRENCE
LINK      EQU   13                    LINKS BETWEEN ROUTINES
WKREG1    EQU   14                    WORK
WKREG2    EQU   15                    REGISTERS
*SYMBOLIC IMMEDIATE OPERANDS
BASIC     EQU   X'80'                 FLAGS FOR BASIC ACTION OCCURRENCE
OBJUN     EQU   X'20'                 FLAGS FOR OBJECT OF THE UNIVERSE
TRUE      EQU   X'01'                 BOOLEAN VALUE 'TRUE'
LIST      EQU   X'40'                 FLAGS FOR LIST OF VALUES
NAMEL     EQU   X'80'                 FLAGS FOR NAMED NOTION
*DISPLACEMENTS FOR SPECIAL ATTRIBUTES
DSUC1      EQU   8
DSUC2      EQU   12
DARG1      EQU   20
DARG2      EQU   24
DARG3      EQU   28
DARG4      EQU   32
          BALR  BR1,0                    LOAD
          USING *,BR1,BR2             BASE
          LA    BR2,4095(BR1)         REGISTERS
          LA    BR2,1(BR2)
          STXIT ,,,MINTERR             CONTINGENCY EXIT
          WROUT ASK,LPR               ASK FOR 1ST ACTION
          RDATA REPLY,ERR             GET REPLY
          SLR   WKREG1,WKREG1
          LH    WKREG1,REPLY          LENGTH
          SH    WKREG1,=H'5'          -1 OF NAME
          STC   WKREG1,MMOVEKEY+1      TO MVC INSTR
          L     1,=V(KEY)
MMOVEKEY  MVC   0(8,1),REPLY+4         NOTION NAME TO KEYARG
          L     1,=V(#NOFIND)
          MVI   0(1),X'03'            SET ERROR CODE
          L     1,=V(DICT)
          GETKY (1),INAREA            GET ADDRESS
```

```
M SOURCE

          L     CURRADD,DATA          ADDRESS OF FIRST ACTION
          BALR  BR1,0                 REDEFINE
          USING *,BR1,BR2             BASE
BASE      LA    BR2,4095(BR1)         REGISTERS
          LA    BR2,1(BR2)
NEXTACT   L     WKREG2,4(CURRADD)         CONTENTS OCCDF CELL
          N     WKREG2,MASK           REMOVE ANY FLAG BITS
          A     WKREG2,=A(SIMTAB)         ADD BASE ADDRESS
          L     WKREG2,0(WKREG2)      ADDR OF SIMULATION ROUTINE
          BALR  LINK,WKREG2               BRANCH TO SIMULATION ROUTINE
          BAL   LINK,SIMSUCC              GET NEXT ACTION

MOUTSW    NOP   NEXTACT               NEXT ACTION WHEN SW IS 'B'
*OTHERWISE CONTROL GOES TO USER OUTPUT ROUTINES
          L     15,=A(#USERINT)
          BALR  14,15                 CALL ROUTINES
          MVI   MOUTSW+1,X'F0'        FLIP SW TO 'B'
          B     NEXTACT
*USER INTERRUPT ROUTINE,SETS SWITCH TO USER OUTPUT ROUTINES
MINTERR   MVI    MOUTSW+1,X'00'        FLIP SW TO NOP
          EXIT  CR

          LTORG

*GET SUCCESSOR
*THE FOLLOWING ROUTINE ASSUMES THAT CURRADD CONTAINS THE ADDRESS OF THE
*SUC1(2) CELL OF THE CURRENT ACTION OCCURRENCE.AT THE END OF THIS
*ROUTINE CURRADD WILL CONTAIN THE ADDRESS OF THE NEXT ACTION OCCURRENCE
*TO BE SIMULATED.
*IF THERE IS NO SUCCESSOR TO THE CURRENT ACTION OCCURRENCE THE
*SIMULATION IS TERMINATED.MULTIPLE SUCCESSORS ARE DEALT WITH IN THE
*PARLSUB ROUTINE
*CALLS:PARLSUB
SIMSUCC   TM    0(CURRADD),LIST       LIST OF SUCCESSORS?
          BZ    SUCCONE               NO
SUCCPARL  ST    LINK,SUCCSAVE         SAVE RETURN ADDRESS
          BAL   LINK,PARLSUB          CALL PARALLELISM ROUTINE
          L     LINK,SUCCSAVE         RESTORE RETURN ADDRESS
          B     SUCCMASK
SUCCONE   CLC   0(4,CURRADD),EMPTY    NO SUCCESSORS?
          BE    SIMEND                NONE

          L     CURRADD,0(CURRADD)    LOAD ADDR OF NEXT ACTION
SUCCMASK  N     CURRADD,MASK          MASK FLAG BITS
          BR    LINK                  EXIT
SIMEND    CLC   STACK(4),EMPTY        STACK OF SUCCESSORS EMPTY?
*THE ABOVE IS DEPENDENT ON THE WAY PARALLELISM IS HANDLED
*SEE PARLSUB
          BE    SIMTERM               YES:SIMTERM
          L     CURRADD,0(CURRADD)    PARAMETER FOR PARLSUB (='EMPTY')
          B     SUCCPARL
SIMTERM   TERM                        END SIMULATION
*NOTE:ANY ADDITIONS MADE TO THE 3-GRAPH DURING A SIMULATION ARE LOST BY
*THIS METHOD.ONE METHOD OF SAVING THE CURRENT CONTENTS IS WITH THE
*CHKPT MACRO,ANOTHER IS THE USE OF A PAM FILE
          TERM  SAFETY MEASURE
```

```
M SOURCE

*INPUT PARAMETERS
*CURRADD CONTAINS THE ADDRESS OF THE CREN ACTION OCCURRENCE TO BE
*SIMULATED
*OUTPUT PARAMETERS
*CURRADD CONTAINS THE ADDRESS OF THE SUC1 ATTRIBUTE
*DESCRIPTION:SIMCREN ROUTINE
*SIMCREN SIMULATES THE CREATION OF A NOTION WHICH BECOMES A VALUE OF
*AN ATTRIBUTE OF AN EXISTING NOTION
*IF ANY ARGUMENT TAKING PART IN THIS ACTION IS VALUELESS THE EFFECT IS
*A NO-OPERATION CONTROL PASSES DIRECTLY TO CRENSUC1
*CALLS:RANSUB,GETBLK,FNDATSUB,FNDVALSU
*************************************************
SIMCREN  TSTARG CURRADD,DARG1,FSTARG,CRENSUC1,LINK,CRENSAVE
*THE FIRST ARGUMENT IS CHOSEN AND STORED IN FSTARG.
         TSTARG CURRADD,DARG2,SNDARG,CRENSUC1,LINK
*THE SECOND ARGUMENT IS CHOSEN AND STORED IN SNDARG
         CRENOT THDARG,FSTARG
*SPACE IS ALLOCATED AND INITIALISED FOR THE NEW NOTION WHOSE ADDRESS IS
*STORED IN THDARG
         FNDATTR FSTARG,SNDARG,ATADD
*THE FIRST ARGUMENT NOTION IS SCANNED TO SEE IF AN ATTRIBUTE WITH NATURE
*THE SECOND ARGUMENT NOTION EXISTS,IF IT DOES ITS ADDRESS IS RETURNED
*IN ATADD
         C     ATADD,LSTPT
         BE    CRENEMPT
*THE ABSENCE OF THE ATTRIBUTE IS INDICATED BY A SPECIAL VALUE IN ATADD,
*IF SO CONTROL GOES TO CRENEMPT.
         ADDVAL ATADD,THDARG
CRENSUC1 MVARG CURRADD,DSUC1,LINK,CRENSAVE
*CURRADD IS SET TO THE SUC1 ATTRIBUTE.
         BR    LINK                    EXIT
CRENEMPT SPATTR SNDARG,ADDSAVE,THDARG,CRENATTR,CRENSUC1
*IF THE ATTR IS A SPECIAL ONE THE NEW NOTION IS STORED OTHERWISE CONTROL
*GOES TO CRENATTR
CRENATTR FNDATTR FSTARG,EMPTY,ATADD
*THE FIRST ARGUMENT NOTION IS SCANNED TO FIND ROOM FOR A NEW ATTRIBUTE,
*THE POSITION IS STORED IN ATADD
         CRATTR ATADD,SNDARG,THDARG
*A NEW ATTRIBUTE IS ADDED WITH NATURE THE 2ND ARGUMENT NOTION AND VALUE
*THE NEW NOTION
         B     CRENSUC1
```

```
M SOURCE

   *INPUT PARAMETERS
   *CURRADD CONTAINS THE ADDRESS OF THE CRER ACTION OCCURRENCE TO BE
   *SIMULATED.
   *OUTPUT PARAMETERS
   *CURRADD CONTAINS THE ADDRESS OF THE SUC1 ATTRIBUTE
   *DESCRIPTION:SIMCREP ROUTINE
   *SIMULATES THE CREATION OF A RELATION BETWEEN 3 EXISTING NOTIONS:ROOT,
   *NATURE AND VALUE OF THE NEW RELATION RESP.IF A NOTION IS NOT SPECIFIED
   *OR IS 'NIL' OR THE RELATION ALREADY EXISTS THE EFFECT IS A NO-OP
   *CALLS:RANSUB,FNDATSUB,FNDVALSU,GETBLK
   ***********************************************************
   SIMCRER  TSTARG CURRADD,DARG1,FSTARG,CRERSUC1,LINK,CRERSAVE
   *THE FIRST ARGUMENT IS CHOSEN AND STORED IN FSTARG.
            TSTARG CURRADD,DARG2,SNDARG,CRERSUC1,LINK
   *THE SECOND ARGUMENT IS OBTAINED AND STORED IN SNDARG.
            TSTARG CURRADD,DARG3,THDARG,CRERSUC1,LINK
   *THE THIRD ARGUMENT IS OBTAINED AND STORED IN THDARG.
            FNDATTR FSTARG,SNDARG,ATADD
   *THE FIRST ARGUMENT  NOTION IS SCANNED FOR AN ATTRIBUTE WHOSE NATURE IS
   *THE SECOND ARGUMENT NOTION,THE ADDRESS OF THE ATTRIBUTE IS STORED IN
   *ATADD
            C       ATADD,LSTPT             ATTRIBUTE EXISTS?

            BE      CRERENPT                NO:CRERENPT
            FNDVAL ATADD,THDARG,VALFND
   *THE ATTRIBUTE IS SCANNED FOR A VALUE WHICH IS THE THIRD ARGUMENT NOTION
   *IF THE RELATION TO BE CREATED ALREADY EXISTS.EXISTENCE OR NON-EXISTENCE
   *IS INDICATED BY VALFND

            CLI     VALFND,TRUE             VALUE FOUND?
            BNE     CRERADVA                NO:CRERADVA
   CRERSUC1 MVARG CURRADD,DSUC1,LINK,CRERSAVE
   *CURRADD IS SET TO THE SUC1 ATTRIBUTE.
            BR      LINK                    EXIT
   CRERADVA ADDVAL ATADD,THDARG
   *THE THDARG NOTION IS ADDED TO THE VALUES OF THE ATTRIBUTE,CREATING THE
   *RELATION REQUIRED
            B       CRERSUC1
   CRERLMPT SPATTR SNDARG,ADDSAVE,THDARG,CRERATTR,CRERSUC1
   *IF THE NATURE OF THE ATTR IS 'SPECIAL' AND NO SUCH ATTR EXISTS IT IS
   *CREATED.OTHERWISE CONTROL  GOES TO CRERATTR.
   CRERATTR FNDATTR FSTARG,EMPTY,ATADD
   *ROOM FOR A NEW ATTRIBUTE OF THE FIRST ARGUMENT NOTION IS OPTAINED AND
   *ITS ADDRESS STORED IN ATADD
            CRATTR ATADD,SNDARG,THDARG
   *A NEW ATTRIBUTE IS ADDED CREATING THE SPECIFIED RELATION
            B       CRERSUC1
```

```
M SOURCE

   *INPUT PARAMETERS
   *CURRADD CONTAINS THE ADDRESS OF THE LAT ACTION OCCURRENCE TO BE
   *SIMULATED
   *OUTPUT PARAMETERS
   *CURRADD CONTAINS THE ADDRESS OF THE SUC1 ATTRIBUTE
   *DESCRIPTION:SIMLAT ROUTINE
   *THE ARG1 ATTRIBUTE OF THE LAT ACTION OCCURRENCE SPECIFIES A ROOT
   *NOTION.FROM THIS NOTION AN ATTRIBUTE IS CHOSEN AT RANDOM.THE NATURE OF
   *THE ATTRIBUTE BECOMES VALUE OF THE ARG2 ATTRIBUTE OF THE LAT OCCURRENCE
   *THE VALUE BECOMES A VALUE OF THE ARG3 ATTRIBUTE.IF NO ROOT NOTION IS
   *SPECIFIED THE EFFECT IS A NO-OPERATION.
   *CALLS:RANSUB,CHRATSUB,FNDVALSU,GETBLK
   **********************************************************************
   SIMLAT   TSTARG CURRADD,DARG1,FSTARG,LATSUC1,LINK,LATSAVE
   *THE FIRST ARGUMENT IS   CHOSEN AND STORED IN FSTARG
            CHATRAN FSTARG,ATADD,NATSAVE,LATSUC1
   *AN ATTRIBUTE OF THE FIRST ARGUMENT NOTION IS CHOSEN AT RANDOM,ITS
   *ADDRESS IS STORED IN ATADD AND ITS NATURE IN NATSAVE
            TSLIST ATADD,,LATNOLST
   *IF THE ATTRIBUTE HAS ONLY 1 VALUE GO TO LATNOLST
            CHVAL ATADD,VALADD,VALSAVE,LATSUC1
   *A VALUE OF THE ATTRIBUTE IS CHOSEN AT RANDOM AND STORED IN VALSAVE
   LATLOAD  LATNAT  CURRADD,24,NATSAVE,LATVALUE
   *THE NATURE OF THE ATTRIBUTE IS MADE A VALUE OF THE ARG2 ATTRIBUTE OF
   *THE LAT ACTION OCCURRENCE
   LATVALUE LATVAL CURRADD,28,VALSAVE,LATSUC1
   *THE VALUE OF THE ATTRIBUTE IS MADE A VALUE OF THE ARG3 ATTRIBUTE OF
   *THE LAT ACTION OCCURRENCE
   LATSUC1  MVARG CURRADD,DSUC1,LINK,LATSAVE
   *CURRADD IS SET TO SUC1 ATTRIBUTE
            BR    LINK                  EXIT
   LATNOLST MVAL  ATADD,VALSAVE
   *THE ATTRIBUTE'S ONLY VALUE IS STORED IN VALSAVE
            B     LATLOAD
```

```
M SOURCE

      *INPUT PARAMETERS
      *CURRADD CONTAINS THE ADDRESS OF THE REP ACTION OCCUPRENCE TO BE
      *SIMULATED.
      *OUTPUT PARAMETERS
      *CURRADD CONTAINS  THE ADDRESS OF THE SUC1 ATTRIBUTE.
      *DESCRIPTION:SIMREP ROUTINE
      *THE ACTION SIMULATED REPLACES A VALUE OF AN ATTRIBUTE SPECIFIED BY
      *ROOT AND NATURE (ARG3 & ARG4 ATTR) BY A VALUE OF AN ATTRIBUTE ALSO
      *SPECIFIED BY ROOT AND NATURE (ARG1 & ARG2 ATTR).EITHER OR BOTH VALUES
      *WILL BE CHOSEN AT RANDOM IF NECESSARY.ABSENCE OF A VALUE TO BE REPLACED
      *MEANS THAT THE ACTION REDUCES TO A TMV ACTION.ABSENCE OF A VALUE TO
      *REPLACE THE 1ST MEANS THAT THE ACTION IS A NO-OPERATION.
      *CALLS:RANSUB,FNDATSUB,RETNOTSB
      **********************************************************
      SIMREP   TSTARG CURRADD,DARG3,THDARG,REPSUC1,LINK,REPSAVE
      *THE THIRD ARGUMENT IS OBTAINED AND STORED IN THDARG
               TSTARG CURRADD,DARG4,FTHARG,REPSUC1,LINK
      *THE FOURTH ARGUMENT IS OBTAINED AND STORED IN FTHARG
               FNDATTR THDARG,FTHARG,ATADD
      *THE THIRD ARGUMENT NOTION IS SCANNED FOR AN ATTRIBUTE WHOSE NATURE IS
      *THE FOURTH ARGUMENT NOTION,THE ATTRIBUTE'S ADDRESS IS STORED IN ATADD
               C     ATADD,LSTPT          ATTRIBUTE EXISTS?
               BNE   REPCONT              YES:REPCONT
      *IF THE ATTRIBUTE DOES NOT EXIST THE ACTION REDUCES TO A TMV
      TMVCALL  MVC   TMVSAVE,REPSAVE      TRANSFER RETURN ADDRESS
               B     TMVREPEN             TMV ROUTINE
      REPCONT  TSLIST ATADD,,REPNOLS2
               CHVAL ATADD,VALADD,,TMVCALL
      *IF ATTRIBUTE HAS MORE THAN 1 VALUE A VALUE IS CHOSEN AT RANDOM,ITS
      *ADDRESS IS GIVEN IN VALADD.IF ONLY 1 VALUE CONTROL TRANSFERS TO
      *REPNOLS2.
               ST    VALADD,REPSAVE+4
      *VALADD IS SAVED FOR LATER USE.
      REPARG1  TSTARG CURRADD,DARG1,FSTARG,REPSUC1,LINK
      *THE FIRST ARGUMENT IS OBTAINED AND STORED IN FSTARG
               TSTARG CURRADD,DARG2,SNDARG,REPSUC1,LINK
      *THE SECOND ARGUMENT IS OBTAINED AND STORED IN SNDARG
               FNDATTR FSTARG,SNDARG,ATADD
      *THE FIRST ARGUMENT NOTION IS SCANNED FOR AN ATTRIBUTE WHOSE NATURE IS
      *THE SECOND ARGUMENT NOTION,IF THE ATTRIBUTE EXISTS ITS ADDRESS IS
      *STORED IN ATADD
               C     ATADD,LSTPT          ATTRIBUTE EXISTS?
               BNE   REPATEND             YES:REPATEND
      REPSUC1  MVARG CURRADD,DSUC1,LINK,REPSAVE
      *CURRADD IS SET TO THE SUC1 ATTRIBUTE
               BR    LINK                 EXIT
      REPATEND TSLIST ATADD,,REPNOLST
      *IF THE ATTRIBUTE HAS ONLY 1 VALUE GO TO REPNOLST
               CHVAL ATADD,VALADD,VALSAVE,REPSUC1
      *CHOOSE A VALUE AT RANDOM AND STORE IT IN VALSAVE
      REPVALUE L     VALADD,REPSAVE+4     GET ADDRESS OF VALUE TO BE REPLACED
               REPVAL VALADD,VALSAVE,REPSUC1
      *REPLACE VALUE ADDRESSED BY VALADD BY VALUE IN VALSAVE,THEN EXIT
      REPNOLST MVAL  ATADD,VALSAVE
      *THE ATTRIBUTE'S ONLY VALUE IS STORED IN VALSAVE
               B     REPVALUE
      REPNOLS2 ST    ATADD,REPSAVE+4      SAVE ADDRESS OF VALUE CELL
               B     REPARG1
```

M SOURCE

```
*INPUT PARAMETERS
*CURRADD CONTAINS THE ADDRESS OF THE SUP ACTION OCCURRENCE TO BE
*SIMULATED.
*OUTPUT PARAMETERS
*CURRADD CONTAINS THE ADDRESS OF THE SUC1 ATTTRIBUTE
*DESCRIPTION:SIMSUP ROUTINE
*SIMSUP SUPPRESSES AN ATTRIBUTE OF A NOTION.THE NOTION IS SPECIFIED BY
*THE VALUE OF ARG1 AND THE NATURE OF THE ATTRIBUTE BY THE VALUE OF ARG2
*OF THE SUP OCCURRENCE.IF ONE OF THESE IS NOT SPECIFIED OR THE ATTRIBUTE
*DOES NOT EXIST THE RESULT IS A NO-OP.IF THE ATTRIBUTE HAS MORE THAN 1
*VALUE THE VALUE TO BE SUPPRESSED IS CHOSEN AT RANDOM.IF A NOTION IS NO
*LONGER REFERENCED AFTER ITS OCCURRENCE AS NATURE OR VALUE OF THE
*ATTRIBUTE HAS BEEN SUPPRESSED IT IS REMOVED FROM THE GRAPH.
*CALLS:RANSUB,FNDATSUB,RETNOTSB,TSTBLK
**************************************************
SIMSUP   TSTARG CURRADD,DARG1,FSTARG,SUPSUC1,LINK,SUPSAVE
*THE FIRST ARGUMENT IS    OBTAINED AND STORED IN FSTARG
         TSTARG CURRADD,DARG2,SNDARG,SUPSUC1,LINK
*THE SECOND ARGUMENT IS OBTAINED AND STORED IN SNDARG
         FNDATTR FSTARG,SNDARG,ATADD
*THE FIRST ARGUMENT NOTION IS SCANNED FOR AN ATTRIBUTE WHOSE NATURE IS
*THE SECOND ARGUMENT NOTION IF IT EXISTS THE ATTRIBUTE'S ADDRESS IS
*STORED IN ATADD
         C     ATADD,LSTPT          ATTRIBUTE EXISTS?
         BNE   SUPATFND             YES:SUPATFND
SUPSUC1  MVARG CURRADD,DSUC1,LINK,SUPSAVE
*CURRADD IS SET TO THE SUC1 ATTRIBUTE;
         BR    LINK                 EXIT
SUPATFND TSLIST ATADD,SUPATLST
*IF ATTRIBUTE HAS MORE THAN 1 VALUE GO TO SUPATLST
         SUPATTR ATADD,SNDARG,FSTARG
*SUPPRESS ATTRIBUTE
         B     SUPSUC1
SUPATLST CHVAL ATADD,VALADD,,SUPSUC1
*A VALUE OF THE ATTRIBUTE IS CHOSEN AT RANDOM,ITS POSITION IS STORED IN
*VALADD
         SUPVAL VALADD,ATADD
*THE VALUE IS REMOVED
         B     SUPSUC1
```

```
M SOURCE

*INPUT PARAMETERS
*CURRADD CONTAINS THE ADDRESS OF THE SUPR ACTION OCCUPRENCE TO BE
*SIMULATED.
*SIMSUPR SUPPRESSES AN ATTRIBUTE WHOSE ROOT,NATURE AND VALUE APE
*SPECIFIED BY THE ARG1,ARG2,ARG3 ATTRIBUTES OF THE SUPR OCCURRENCE
*THE RESULT IS A NO-OP IF ANY ONE ARGUMENT IS NOT SPECIFIED OP IF THE
*GIVEN ATTRIBUTE DOES NOT EXIST.IF THE VALUE OF THE ATTRIBUTE IS ONE OF
*SEVERAL ONLY THE SPECIFIED ONE IS SUPPRESSED.IF THE LAST REFERENCE TO
*A NOTION IS SUPPRESSED BY THE SUPR OCCURRENCE THE NOTION IS REMOVED
*FROM THE 3-GRAPH
*CALLS:RANSUB,FNDATSUB,RETNOTSB,TSTELK,FNDVALSU
**********************************************************
SIMSUPR  TSTARG CURRADD,DARG1,FSTARG,SUPRSUC1,LINK,SUPRSAVE
*THE FIRST ARGUMENT IS OBTAINED AND STORED IN FSTARG
         TSTARG CURRADD,DARG2,SNDARG,SUPRSUC1,LINK
*THE SECOND ARGUMENT IS OBTAINED AND STORED IN SNDARG
         TSTARG CURRADD,DARG3,THDARG,SUPRSUC1,LINK
*THE THIRD ARGUMENT IS OBTAINED AND STORED IN THDARG
         FNDATTR FSTARG,SNDARG,ATADD
*THE FIRST ARGUMENT NOTION IS SCANNED FOR AN ATTRIBUTE WITH NATUPE THE
*SECCND ARGUMENT NOTION.ATADD CONTAINS THE ADDRESS OF THE ATTRIBUTE IF
*IT IS FOUND.
         C       ATADD,LSTPT          ATTRIBUTE EXISTS
         BNE     SUPRATFD             YES:SUPRATFD
SUPRSUC1 MVARG CURRADD,DSUC1,LINK,SUPRSAVE
*CURRADD IS SET TO THE SUC1 ATTRIBUTE
         BR      LINK                 EXIT
SUPRATFD TSLIST ATADD,SUPRATLS
*IF THE ATTRIBUTE HAS MORE THAN 1 VALUE CONTROL PASSES TO SUPPATLS
         CVAL    ATADD,THDARG,SUPRSUC1
*THE VALUE OF THE ATTRIBUTE IS INSPECTED TO SEE IF IT CORRESPONDS TO
*THE THIRD ARGUMENT IF NOT CONTROL PASSES TO SUPRSUC1 IE THE RELATION
*TO BE SUPPRESSED DOES NOT EXIST
         SUPATTR ATADD,SNDARG,FSTARG
*THE RELATION IS SUPPRESSED
         B       SUPRSUC1
SUPRATLS FNDVAL ATADD,THDARG,VALFND
*THE VALUES OF THE ATTRIBUTE ARE SCANNED FOR THE THIRD ARGUMENT NOTION
         CLI     VALFND,TRUE          VALUE EXISTS?
         BNE     SUPRSUC1             NO,NO RELATION TO BE SUPPRESSED
         SUPVAL VALADD,ATADD
*THE RELATION IS SUPRESSED BY REMOVING THE THIRD ARGUMENT NOTION FROM
*THE VALUE LIST (ADDRESS OF CELL CONCERNED IS STORED IN VALADD BY
*FNDVAL).
         B       SUPPSUC1
```

```
M SOURCE

*INPUT PARAMETERS
*CURRADD CONTAINS THE ADDRESS OF THE TMV ACTION OCCURRENCE TO BE
*SIMULATED.
*OUTPUT PARAMETERS
*CURRADD CONTAINS THE ADDRESS OF THE SUC1 ATTRIBUTE
*DESCRIPTION:SIMTMV ROUTINE
*SIMTMV SIMULATES THE TRANSFER OF A VALUE OF AN ATTR OF A GIVEN NOTION
*TO AN ATTRIBUTE OF ANOTHER NOTION.THE FIRST NOTION IS SPECIFIED BY ARG1
*OF THE TMV OCCURENCE,THE ATTRIBUTE'S NATURE BY ARG2,THE 2ND NOTION BY
*ARG3 AND ITS ATTRIBUTE (NATURE) BY ARG4.IF ANY OF THESE IS NOT SPECIF-
*-IED OR THE FIRST ATTRIBUTE DOES NOT EXIST OR THE VALUE CHOSEN IS
*ALREADY A VALUE OF THE 2ND ATTRIBUTE THE ACTION OCCURRENCE HAS NO
*EFFECT.
*CALLS:RANSUB,FNDATSUB,FNDVALSU,GETBLK
*THE THIRD AND FOURTH ARGUMENTS ARE CHOSEN FIRST BECAUSE OF THE ENTRY
*POINT FROM THE SIMREP ROUTINE
**********************************************************
SIMTMV   TSTARG CURRADD,DARG3,THDARG,TMVSUC1,LINK,TMVSAVE
*THE THIRD ARGUMENT NOTION IS OBTAINED AND STORED IN THDARG
         TSTARG CURRADD,DARG4,FTHARG,TMVSUC1,LINK
*THE FOURTH ARGUMENT NOTION IS OBTAINED AND STORED IN FTHARG
*ENTRY POINT FROM SIMREP ROUTINE WHEN REP ACTION REDUCES TO A TMV
TMVREPEN TSTARG CURRADD,DARG1,FSTARG,TMVSUC1,LINK
*THE FIRST ARGUMENT IS OBTAINED AND STORED IN FSTARG
         TSTARG CURRADD,DARG2,SNDARG,TMVSUC1,LINK
*THE SECOND ARGUMENT IS OBTAINED AND STORED IN SNDARG
         FNDATTR FSTARG,SNDARG,ATADD
*THE FIRST ARGUMENT NOTION IS SCANNED FOR AN ATTRIBUTE WHOSE NATURE IS
*THE SECOND ARGUMENT NOTION,IF IT EXISTS THE ATTRIBUTE'S ADDRESS IS
*STORED IN ATADD
         C     ATADD,LSTPT          ATTRIBUTE EXISTS
         BNE   TMVATFND             YES:TMVATFND
TMVSUC1  MVARG CURRADD,DSUC1,LINK,TMVSAVE
*CURRADD IS SET TO SUC1 ATTR
         BR    LINK                 EXIT
TMVATFND TSLIST ATADD,,TMVNOLST
*IF THE ATTRIBUTE HAS ONLY 1 VALUE GO TO TMVNOLST
         CHVAL ATADD,VALADD,VALSAVE,TMVSUC1
*A VALUE OF THE ATTR IS CHOSEN AT RANDOM AND STORED IN VALSAVE
TMVFNDAT FNDATTR THDARG,FTHARG,ATADD
*THE THIRD ARGUMENT NOTION IS SCANNED FOR AN ATTRIBUTE WITH NATURE THE
*FOURTH ARGUMENT NOTION
         C     ATADD,LSTPT          ATTRIBUTE EXISTS?
         BNE TMVAT2EX               YES:TMVAT2EX
         SPATTR FTHARG,ADDSAVE,VALSAVE,TMVNSPEC,TMVSUC1
*IF NATURE OF ATTRIBUTE SPECIAL,ATTRIBUTE IS ADDED AND
*CONTROL PASSES TO TMVSUC1
TMVNSPEC FNDATTR THDARG,EMPTY,ATADD
*SPACE IS FOUND FOR A NEW ATTRIBUTE OF THE THIRD ARGUMENT NOTION
         CRATTR ATADD,FTHARG,VALSAVE
*THE NEW ATTRIBUTE IS CREATED
         B     TMVSUC1
```

```
M SOURCE

   TMVNOLST MVAL  ATADD,VALSAVE
   *THE ATTRIBUTE'S ONLY VALUE IS STORED IN VALSAVE
            B     TMVFNDAT
   TMVAT2EX FNDVAL ATADD,VALSAVE,VALFND
   *THE ATTRIBUTE OF THE THIRD ARGUMENT NOTION IS SCANNED FOR THE VALUE TO
   *BE ADDED.
            CLI   VALFND,TRUE          VALUE ALREADY EXISTS?
            BE    TMVSUC1              YES:TMVSUC1
            ADDVAL ATADD,VALSAVE
   *THE VALUE IS TRANSMITTED TO THE ATTRIBUTE OF THE THIRD ARGUMENT NOTION
            B     TMVSUC1



   *INPUT PARAMETERS
   *CURRADD CONTAINS THE ADDRESS OF THE COMP ACTION OCCURRENCE TO BE
   *SIMULATED.
   *OUTPUT PARAMETERS
   *CURRADD CONTAINS THE ADDRESS OF THE SUC1 OR SUC2 ATTRIBUTE DEPENDING ON
   *THE RESULT OF THE SIMULATION.
   *DESCRIPTION:SIMCOMP ROUTINE
   *SIMCOMP SIMULATES THE COMPARISON OF TWO 3-GRAPH NOTIONS(CHOSEN AT
   *RANDOM IF NECESSARY) AND THE CHOICE OF SUCCESSOR ACTION DEPENDING ON
   *THE RESULT
   *CALLS:RANSUB
   ************************************************************
   SIMCOMP  TSTARG CURRADD,DARG1,FSTARG,,LINK,COMPSAVE
   *A VALUE OF ARG1 IS CHOSEN FOR FSTARG.
            TSTARG CURRADD,DARG2,SNDARG,,LINK
   *THE SECOND ARGUMENT IS CHOSEN AND STORED IN SNDARG
            CLC   FSTARG,SNDARG        COMPARE THE TWO NOTIONS

            BE    COMPSUC1             IF THE SAME,SUC1 IS NEXT ACTION
   COMPSUC2 MVARG CURRADD,DSUC2,LINK,COMPSAVE
   *THE NOTIONS ARE NOT THE SAME;SUC2 IS THE NEXT ACTION SO CURRADD IS SET
   *TO THE SUC2 ATTRIBUTE
            BR    LINK                 EXIT
   COMPSUC1 MVARG CURRADD,DSUC1,LINK,COMPSAVE
   *CURRADD IS SET TO THE SUC1 ATTRIBUTE
            BR    LINK                 EXIT
```

```
* M SOURCE

    *INPUT PARAMETERS
    *ATADD CONTAINS ADDRESS OF VALUE CELL POINTING TO LIST OF VALUES
    *REG 11 CONTAINS ADDRESS OF 1ST BLOCK OF LIST TO BE SEARCHED
    *OUTPUT PARAMETERS
    *REG 11 CONTAINS ADDRESS OF NON-EMPTY CELL OR 'NIL'
    *ATADD IS UNCHANGED
    *DESCRIPTION:FSTVALSB ROUTINE
    *THIS ROUTINE SCANS A VALUE LIST TO FIND A NON-EMPTY CELL.IT STARTS AT
    *THE ADDRESS IN REG 11,SKIPS THE 1ST CELL OF THE BLOCK (AVOIDS PROBLEMS
    *WITH THE COUNTER CELL) AND STOPS THE FIRST TIME A NON-EMPTY CELL IS
    *ENCOUNTERED.IF THE END OF THE LIST IS REACHED THE ROUTINE STARTS AGAIN
    *FROM THE BEGINNING OF THE LIST.IF IT REACHES THE END OF THE LIST AGAIN
    *IT RETURNS THE VALUE 'NIL'
    *CALLS:0
    *CALLED BY:RANSUB
FSTVALSB STM   9,10,FSTVSAVE        SAVE WORK REG CONTENTS
         LA    9,2                  CONTROL TO PREVENT INFINITE LOOPING
FSTVLA   LA    10,3                 LOOP CONTROL (3 CELLS IN 1ST BLOCK)

FSTVLOOP LA    11,4(11)             MOVE TO NEXT CELL
FSTVCLC  CLC   0(4,11),EMPTY        CELL EMPTY?
         BNE   FSTVFND              NO->FSTVFND
         BCT   10,FSTVLOOP
         LA    11,4(11)             MOVE TO POINTER CELL
         CLC   0(4,11),LSTPT        END OF LIST?
         BE    FSTVSTAG             YES->FSTVSTAG
         L     11,0(11)             ADDRESS OF NEXT BLOCK
         LA    10,4                 LOOP CONTROL
         B     FSTVCLC
FSTVFND  LM    9,10,FSTVSAVE        RESTORE WORK REGS
         BR    LINK                 EXIT
FSTVSTAG L     11,0(ATADD)          START OF LIST

         BCT   9,FSTVLA             IF WRAPAROUND HAS NOT YET OCCURRED
*                                   GO TO FSTVLA
         L     11,EMPTY             RETURN NIL SINCE LIST COMPLETELY
         B     FSTVFND              EMPTY
```

M SOURCE

```
*INPUT PARAMETERS
*ROOTAD CONTAINS ADDRESS OF NOTION TO BE SCANNED
*REG 11 CONTAINS ADDRESS OF FIRST BLOCK TO BE SCANNED
*OUTPUT PARAMETERS
*REG 11 CONTAINS ADDRESS OF VALUE CELL OF 1ST NON-EMPTY ATTRIBUTE
*NATADD CONTAINS ADDRESS OF NOTION WHICH IS NATURE OF ATTRIBUTE
*ROOTAD IS UNCHANGED
*DESCRIPTION:FSTATSUB ROUTINE
*THIS ROUTINE HAS TWO ENTRY POINTS.FSTATSUB IS THE NORMAL ENTRY POINT,
*FSTABEG IS USED WHEN THE 1ST TWO BLOCKS OF THE NOTION ARE TO BE
*SEARCHED FIRST.
*STARTING FROM THE ADDRESS IN REG 11 THE NOTION IS SEARCHED BLOCK BY
*BLOCK.THE SEARCH TERMINATES AS SOON AS A NON-EMPTY NATURE CELL IS
*ENCOUNTERED.IF THE SEARCH REACHES THE END OF THE NOTION IT STARTS FROM
*THE BEGINNING AGAIN (ROOTAD).IF THE NOTION IS EMPTY AN ERROR MESSAGE
*IS OUTPUT AND THE PROGRAM TERMINATES
*CALLS:0
*CALLED BY:CHRATSUB
FSTATSUB STM   8,10,FSTASAVE        SAVE WORK REG CONTENTS
         LA    9,2                  CONTROL TO PREVENT INFINITE LOOPING
FSTALOOP LA    10,2                 LOOP CONTROL(2 ATTR/BLOCK)
FSTACLC  CLC   0(4,11),EMPTY        NATURE OF ATTR EMPTY? (& ATTR)
         BNE   FSTAFND              NO->FSTAFND
         LA    11,8(11)             MOVE TO NEXT ATTRIBUTE
         BCT   10,FSTACLC
         L     11,0(11)             LOAD POINTER CELL CONTENTS
FSTAPTR  C     11,LSTPT             END OF NOTION?
         BNE   FSTALOOP             NO->FSTALOOP

         BCT   9,FSTABEG            WHILE WRAPAROUND HAS NOT OCCURRED GO
*                                   TO FSTABEG
         ERRCALL ERRMESS2
*SET ERROR MESSAGE PARAMS AND BRANCH
FSTAFND  L     NATADD,0(11)         NATURE OF ATTRIBUTE RETURNED
         LA    11,4(11)             ADDR OF (VALUE CELL OF) ATTR
         LM    8,10,FSTASAVE        RESTORE WORK REGS
         BR    LINK                 EXIT
*SCAN OF 1ST 2 BLOCKS
*SKIPPING THE STATISTICS CELL,THE BLOCKS ARE TESTED CELL BY CELL WHILE
*REG 8 POINTS INTO A TABLE CONTAINING THE ADDRESS OF THE CORRESPONDING
*NATURES.IF THE SEARCH REACHES THE END OF THIS TABLE THEN THE REST OF
*THE NOTION IS SEARCHED IN THE ORDINARY WAY.
FSTABEG  LA    11,4(ROOTAD)         ADDRESS OCCDE CELL OF NOTION
         LA    10,3                 LOOP CONTROL (3 ATTR IN 1ST BLOCK)
         LA    8,SPATTAB            TABLE OF SPECIAL NOTION ADDRESSES
FSTAL01  CLC   0(4,11),EMPTY        CELL EMPTY?
         BNE   FSTAFND2             NO->FSTAFND2

         LA    8,4(8)               MOVE POINTER IN SPATTAB
         LA    11,4(11)             MOVE TO NEXT CELL
         BCT   10,FSTAL01
         C     8,=A(SPATTAB+28)     END OF SPECIAL ATTRIBUTES?
```

```
*M SOURCE

         PE    FSTACONT            YES->FSTACONT
         LA    10,4                LOOP CONTROL (4 ATTR IN 2ND BLOCK)
         LA    11,4(11)            ADDRESS ARG1 CELL
         B     FSTALO1
FSTAFND2 L     NATADD,0(8)         NATURE OF ATTRIBUTE RETURNED
*                                  ADDR OF VALUE OF ATTR OK
         LM    8,10,FSTASAVE       RESTORE WORK REGS
         BR    LINK                EXIT
FSTACONT L     11,0(11)            CONTENTS OF POINTER CELL
         B     FSTAPTR


*INPUT PARAMETERS
*ROOTAD CONTAINS ADDRESS OF FIRST (OR ONLY) BLOCK TO BE RETURNED TO
*FREELIST
*OUTPUT PARAMETERS
*NONE,ROOTAD IS UNCHANGED
*DESCRIPTION:RETBLK ROUTINE
*THE ROUTINE SCANS THE POINTER CELL OF THE BLOCKS TO BE RETURNED TO THE
*FREELIST UNTIL IT FINDS THE END OF THE CHAIN THEN IT FORMS THE ADDRESS
*OF THE FREE LIST HEADER CELL AND CHECKS WHETHER THERE ARE ANY BLOCKS
*IN THE FREE LIST.IF THERE ARE,THEY ARE CHAINED TO THE END OF THE NEW
*CHAIN OF BLOCKS. THE ADDR OF THE 1ST BLOCK IN THE NEW OR UPDATED FREE
*LIST IS STORED IN THE FREE LIST HEADER AND,FINALLY,THE CORRESPONDING
*ENTRY IN THE PAGETAB IS RESET.
*CALLS:0
*CALLED BY:TSTBLK,RETVALST,RETNOTSB
RETBLK   STM   10,11,RETSAVE       STORE WORK REGISTER CONTENTS

         LA    11,16(ROOTAD)       ADDRESS OF POINTER CELL
RETBCLC  CLC   0(4,11),LSTPT       END OF CHAIN?
         BE    RETFRLSA            YES:RETFRLSA
         L     11,0(11)            NEXT BLOCK
         LA    11,16(11)           POINTER CELL
         B     RETBCLC
RETFRLSA LR    10,ROOTAD           FORM
         SRL   10,12               ADDRESS OF
         SLL   10,12               FREELIST
         O     10,FRLSTAD          HEADER
         CLC   0(4,10),EMPTY       FREE LIST EMPTY?
         BE    RETRBLKS+6          YES
RETRBLKS MVC   0(4,11),0(10)       CHAIN FREE LIST TO END OF NEW BLOCKS

         ST    ROOTAD,0(10)        START ADDR OF NEW BLOCKS IN FREELIST
*                                  HEADER
         N     10,MASK             MASK FLAG BITS
         SRL   10,12               REMOVE PAGE REL ADDR
         A     10,APAGETAB         ADDR CORRESPONDING ENTRY IN PAGETAB
         OI    0(10),X'01'         INDICATE NON-EMPTY FREELIST IN PAGE
         LM    10,11,RETSAVE       RESTORE WORK REGS
         BR    LINK                EXIT
```

```
M SOURCE

    *INPUT PARAMETERS
    *CURRADD CONTAINS EITHER THE ADDRESS OF THE SUC1(2) CELL OF THE CURRENT
    *ACTION OR 'NIL'(EMPTY)
    *OUTPUT PARAMETERS
    *CURRADD CONTAINS THE ADDRESS OF THE NEXT ACTION TO BE SIMULATED
    *DESCRIPTION:PARLSUB ROUTINE
    *THIS ROUTINE HAS TWO SEPARATE FUNCTIONS
    *A)WHEN CURRADD ADDRESSES A SUC1(2) CELL THE LIST OF (PARALLEL) SUCCESS-
    *ORS OF THE CURRENT ACTION ARE ADDED TO THE TOP OF A STACK OF SUCCESSORS
    *B)AFTER THIS OR IF CURRADD CONTAINS 'NIL' THE TOP ENTRY IN THE STACK
    *IS REMOVED AND RETURNED TO THE CALLING ROUTINE
    *CALLS:0
    *CALLED BY:SIMSUCC
PARLSUB  STM    10,11,PARLSAVE      SAVE WORK REG CONTENTS
         L      11,STCKPNTR         CURRENT TOP ENTRY IN STACK
         C      CURRADD,EMPTY       ADD OR REMOVE ENTRIES IN STACK?

         BE     PARLREM             REMOVE:PARLREM
         L      CURRADD,0(CURRADD)  ADDRESS OF START OF LIST OF VALUES
PARLADD  LA     10,3                LOOP CONTROL(BECAUSE OF COUNTER CELL
*                                   ONLY 3 VALUES IN 1ST BLOCK)
PARLOOP  LA     CURRADD,4(CURRADD)  MOVE TO NEXT CELL
PARLCLC  CLC    0(4,CURRADD),EMPTY  CELL EMPTY?
         BE     PARLBCT             YES->PARLBCT
         C      11,=A(STCKPNTR)     END OF STACK?
    *NOTE:DEPENDS ON STCKPNTR BEING DECLARED IMMEDIATELY BEHIND STACK
         BE     PARLERR             YES->PARLERR
         MVC    0(4,11),0(CURRADD)  ADD ENTRY TO STACK
         LA     11,4(11)            NEXT FREE ENTRY IN STACK
PARLBCT  BCT    10,PARLOOP
         LA     CURRADD,4(CURRADD)  MOVE TO POINTER CELL

         CLC    0(4,CURRADD),LSTPT  END OF LIST?
         BE     PARLREM             YES->PARLREM
         L      CURRADD,0(CURRADD)  ADDRESS OF NEXT BLOCK
         LA     10,4                LOOP CONTROL
         B      PARLCLC
    *SET ERROR PARAMS AND BRANCH
PARLERR  ERRCALL ERRMESS4            (NO RETURN)
PARLREM  S      11,=F'4'            UPDATE STCKPNTR
         L      CURRADD,0(11)       TAKE TOP ENTRY OFF STACK
         MVC    0(4,11),EMPTY       UPDATE STACK
         ST     11,STCKPNTR         UPDATE STCKPNTR

PARLEXIT LM     10,11,PARLSAVE      RESTORE WORK REGISTERS
         BR     LINK                EXIT
```

```
M SOURCE

*INPUT PARAMETERS
*ATADD CONTAINS ADDRESS OF VALUE CELL POINTING TO LIST OF VALUES
*OUTPUT PARAMETERS
*VALADD CONTAINS ADDRESS OF A NON-EMPTY VALUE CELL IN THE LIST
*ATADD IS UNCHANGED
*DESCRIPTION:RANSUB ROUTINE
*THIS ROUTINE GETS THE TASK TIME FOR THE CURRENT TASK AND DIVIDES IT BY
*THE NO OF CELLS ALLOCATED TO THE VALUE LIST ADDRESSED BY ATADD,THIS
*PRODUCES A 'RANDOM' NO WHICH IS USED TO PICK OUT A CELL FROM THE VALUE
*LIST.IF THIS CELL CONTAINS A VALUE,ITS ADDRESS IS RETURNED TO THE
*CALLING ROUTINE IN VALADD.IF THE CELL IS EMPTY THE ROUTINE GENERATES A
*NEW RANDOM NO,ETC.IF,AFTER 3 NOS HAVE BEEN GENERATED A NON-EMPTY CELL
*HAS NOT BEEN FOUND THE FSTVALSB ROUTINE IS CALLED TO SEARCH SYSTEMATIC-
*ALLY FOR A NON-EMPTY CELL
*CALLS:FSTVALSB
*CALLED BY:SIMCOMP,SIMCREN,SIMIFOU,SIMCRER,SIMSUP,SIMSUPR,SIMTMV
RANSUB   STM   8,11,RANSAVE          WORK REGS
*NOTE:SAVE AREA=5 WORDS,1 WORD NEEDED BY ROUTINE.
         XR    10,10                 INITIALISE TRY COUNT
RANSTART L     11,0(ATADD)           ADDRESS OF COUNTER CELL
         N     11,MASK               MASK FLAG BITS
         TMODE TASKINFO              GET TASK INFORMATION
         L     8,TASKINFO+24         CPU TIME

         SRDA  8,32                  INITIALISE REGISTERS(PROPAGATE SIGN)
         L     WKREG1,0(11)          COUNTER CELL CONTENTS
         N     WKREG1,MASK           MASK FLAG BITS
         DR    8,WKREG1              DIVIDE BY NO OF CELLS ALLOCATED
         LA    8,1(8)                8 CONTAINS REMAINDER R;Q LE R LE
*NO OF CELLS - 1.ADD 1 FOR COUNTER CELL
         SRDA  8,32                  REINITIALISE REGISTERS
         D     8,=F'4'               DIV BY NO OF CELLS/BLOCK
         LTR   9,9                   TEST NO OF BLOCK=0
         BZ    RANSKIP               YES->RANSKIP
RANLOOP  LA    11,16(11)             SKIP
         L     11,0(11)              TO
         BCT   9,RANLOOP             BLOCK CONTAINING CELL CHOSEN
*CELL NO BETWEEN 0 & 3
RANSKIP  MH    8,=H'4'               * BY CELL LENGTH
         AR    11,8                  ADDRESS OF CELL
         CLC   0(4,11),EMPTY         CELL EMPTY?

         BE    RANRETRY              YES->RANRETRY
RANEXIT  LR    VALADD,11             RETURN CELL ADDRESS
         LM    8,11,RANSAVE          RESTORE REGS
         BR    LINK
RANRETRY LA    10,1(10)              INCREMENT TRY COUNT
         CH    10,=H'3'              3 TRIES?
         BL    RANSTART              NO->RANSTART
         SR    11,8                  ADDRESS OF BLOCK OF LAST TRY
         ST    LINK,RANSAVE+16       SAVE CURRENT RETURN ADDRESS
         BAL   LINK,FSTVALSB         CALL ROUTINE TO SCAN LIST FOR 1ST
*NON-EMPTY CELL FOLLOWING LAST ATTEMPT,ADDRESS RETURNED IN REG11
         L     LINK,RANSAVE+16       RESTORE RETURN ADDRESS
         B     RANEXIT
```

```
M SOURCE

*INPUT PARAMETERS
*INDIC INDICATES THE TYPE OF BLOCK TO BE TESTED IF 'N'-AN ATTRIBUTE
*BLOCK OF A NOTION,V-A BLOCK OF A VALUE LIST
*STADDR CONTAINS THE START ADDRESS OF THE CHAIN OF BLOCKS CONCERNED
*CELL CONTAINS THE ADDRESS OF A CELL IN THE BLOCK TO BE TESTED
*OUTPUT PARAMETERS
*NONE.INDIC IS UNCHANGED
*DESCRIPTION:TSTBLK ROUTINE
*AFTER MASKING THE FLAG BITS OF THE ADDRESS IN CELL,INDIC IS TESTED TO
*SEE WHICH TYPE OF BLOCK IS TO BE TESTED.IF IT IS AN ATTRIBUTE BLOCK AN
*PROVES TO BE ONE OF THE 1ST TWO BLOCKS OF THE NOTION CONTROL RETURNS
*TO THE CALLING ROUTINE SINCE THESE BLOCKS REMAIN WHILE THE NOTION
*EXISTS,THE SAME APPLIES TO THE 1ST BLOCK OF A VALUE LIST WHICH REMAINS
*WHILE THE ATTRIBUTE EXISTS
*IF NEITHER OF THE ABOVE IS THE CASE THE CHAIN IS SCANNED UNTIL THE
*BLOCK TO BE TESTED IS FOUND.EACH CELL OF THE BLOCK IS THEN SCANNED.IF
*A NON-EMPTY CELL IS FOUND CONTROL RETURNS TO THE CALLING ROUTINE IF
*THE BLOCK IS EMPTY BIT 24 OF THE STATISTICS OR COUNTER CELL (NOTION OR
*VALUE LIST) IS TESTED TO SEE IF THE CHAIN ALREADY CONTAINS AN EMPTY
*BLOCK.IF IT DOES,THE RETBLK ROUTINE IS CALLED TO RETURN THE CURRENT
*BLOCK TO THE FREE LIST,OTHERWISE THE BIT IS SET (=1) TO INDICATE THAT
*AN EMPTY BLOCK NOW EXISTS.
*CALLS:RETBLK
*CALLED BY:SIMSUP,SIMSUPR
TSTBLK   STM   8,11,TSTBSAVE      STORE WORK REG CONTENTS
         N     CELL,MASK          REMOVE FLAGS
         CLI   INDIC,'N'          BLOCK IN NOTION OR VALUE LIST?
         BE    TSTBNOT            NOTION->TSTBNOT
         LR    11,STADDR          START OF CHAIN
         LA    10,16(11)          POINTER CELL OF 1ST BLOCK OF CHAIN
TSTBCOMP CR    CELL,10            BLOCK TO BE TESTED IS CURRENT ONE?
         BL    TSTBEND            YES->TSTBEND
TSTBEND  CLC   0(4,10),LSTPT      END OF LIST?
         BE    TSTBERR            YES->ERROR
         ST    10,TSTBPTR         SAVE ADDR OF POINTER CELL
         L     11,0(10)           NEXT BLOCK
         LA    10,16(11)          POINTER CELL

         B     TSTBCOMP
TSTBERR  ERRCALL ERRM10           ERROR MESSAGE AND TERMD
TSTBEND  LR    9,11               ADDRESS OF NEXT BLOCK
         CR    9,STADDR           1ST BLOCK OF VALUE LIST IS NEVER
         BE    TSTBNEMP           REMOVED
         LA    8,4                LOOP CONTROL
TSTBLOOP CLC   0(4,9),EMPTY       CELL EMPTY?
         BNE   TSTBNEMP           NO->TSTBNEMP
         LA    9,4(9)             NEXT CELL
         BCT   8,TSTBLOOP
         TM    0(STADDR),X'10'    DOES EMPTY BLOCK ALREADY EXIST?
         BO    TSTBRETB           YES->TSTBRETB

         OI    0(STADDR),X'10'    INDICATE EMPTY BLOCK EXISTS
```

M SOURCE

```
TSTBNEMP LM    8,11,TSTBSAVE        RESTORE WORK REGS
         BR    LINK                 EXIT
TSTBRETB LR    ROOTAD,11            ADDR OF BLOCK,RETPLK PARAM
         N     ROOTAD,MASK          MASK FLAG BITS
         L     10,TSTBPTR           UPDATE CHAIN
         MVC   0(4,10),0(9)         TO EXCLUDE BLOCK REMOVED
         MVC   0(4,9),LSTPT         END OF CHAIN FOR RETPLK ROUTINES
         ST    LINK,TSTBSAVE+16     SAVE CURRENT RETURN ADDRESS
         BAL   LINK,RETBLK          CALL ROUTINE WHICH WILL RETURN GIVEN
*                                    BLOCK TO FREELIST
         L     LINK,TSTBSAVE+16     RESTORE RETURN ADDRESS
         CLI   INDIC,'V'            VALUE LIST TESTED?
         BNE   TSTBNEMP             NO
         L     8,0(STADDR)          UPDATE COUNTER

         S     8,=F'4'              CELL OF
         ST    8,0(STADDR)          VALUE LIST
         B     TSTBNEMP
TSTBNOT  LA    11,36(STADDR)        ADDRESS OF POINTER CELL OF 2ND BLOCK
         CR    CELL,11              BLOCK TO BE TESTED ONE OF 1ST 2?
         BL    TSTBNEMP             YES->TSTBNEMP,THE 1ST 2 BLOCKS DO
*                                    NOT COUNT
         LR    10,11                POINTER CELL
         B     TSTBEND
```

M SOURCE

```
*INPUT PARAMLTERS
*NOBLCKS INDICATES THE NO OF (CONTIGUOUS) BYTES TO BE ALLOCATED
*NEWADD CONTAINS THE ADDRESS TO WHICH THE NEW BLOCK(S) WILL BE CHAINED,
*(PROVIDES A BASE FOR THE SEARCH)
*OUTPUT PARAMETERS
*NEWADD CONTAINS THE START ADDRESS OF THE NEW BLOCK(S)
*NOBLCKS IS UNCHANGED
*DESCRIPTION:GETBLK ROUTINE
*THE ADDRESS IN NEWADD IS USED TO GET THE FREE LIST HEADER CELL OF THE
*PAGE REQUESTING NEW BLOCK(S).IF THE PAGE IS FULL THE PAGETABLE IS
*SEARCHED FOR ANOTHER PAGE BELONGING TO THE SAME 3-GRAPH SECTION AND
*NOT FULL.IF THERE IS NO SUCH PAGE AVAILABLE THE REQM SYSTEM MACRO IS
*USED TO ALLOCATE A NEW PAGE TO THE PROGRAM.THE NEW PAGE IS INITIALISED
*AND ENTERED IN PAGETAB.
*ONCE A PAGE WITH ROOM AVAILABLE IS FOUND,NOBLCKS (PARAMETER) IS TESTED
*TO SEE HOW MANY 20-BYTE BLOCKS ARE REQUIRED.IF ONLY 1 BLOCK IS WANTED
*THE 1ST BLOCK OF THE FREE LIST IS EXTRACTED AND INITIALISED,THE FREE
*LIST HEADER(AND PAGETABLE,IF NECESSARY) IS UPDATED AND CONTROL RETURNS
*TO THE CALLING ROUTINE
*IF 2 BLOCKS ARE NEEDED THINGS ARE MORE COMPLICATED,SINCE THE BLOCKS
*MUST BE PHYSICALLY CONTIGUOUS AND THERE MAY NOT BE 2 SUCH BLOCKS IN THE
*FREE LIST OF THE 1ST PAGE AVAILABLE,WITH THE RESULT THAT THE PAGETABLE
*MAY HAVE TO BE SEARCHED MORE THAN ONCE AND EVENTUALLY A REQM MAY BE
*REQUIRED.
*CALLS:0
*CALLED BY:FNDVALSU,FNDATSUB,SIMCREN,SIMREP,SIMTMV,SIMLAT
GETBLK   STM   9,11,GETBSAVE        SAVE WORK REG CONTENTS
*NOTE:SAVE AREA=4 WORDS,1 WORD NEEDED IN ROUTINE
         SRL   NEWADD,12            ISOLATE
         STC   NEWADD,PAGENO        PAGE NO
         SLL   NEWADD,12            FORM FREE LIST
         O     NEWADD,FRLSTAD       HEADER ADDRESS
         XR    9,9                  ZERO REG 9,DETERMINES INITIALISATION
*                                   OF TRT INSTRUCTION
         CLC   0(4,NEWADD),EMPTY    PAGE FULL?

         BE    GETBFDPG             YES->GETBFDPG
GETBCLI  CLI   NOBLCKS,X'28'        1 OR 2 BLOCKS REQUIRED?
         BL    GETB2P               2 BLOCKS->GETB2B
*EXTRACT TOP BLOCK FROM FREE LIST AND INITIALISE
GETB1B   L     11,0(NEWADD)         ADDRESS OF 1ST BLOCK IN FREE LIST
         L     10,16(11)            ADDRESS OF 2ND BLOCK OR LSTPT
         LR    WKREG1,11
         LA    WKREG2,4             INITIALISE
GETB1IN  MVC   0(4,WKREG1),EMPTY
         LA    WKREG1,4(WKREG1)     NEW
         BCT   WKREG2,GETB1IN
         MVC   0(4,WKREG1),LSTPT    BLOCK
         C     10,LSTPT             PAGE NOW FULL?
         BE    GETBPGFL             YES->GETBFGFL

         ST    ,10,0(NEWADD)        UPDATE FREE LIST HEADER
```

M SOURCE

```
GETBEXIT LR    NEWADD,11            RETURN ADDRESS OF NEW BLOCK
         LM    9,11,GETBSAVE        RESTORE WORK REGS
         BR    LINK                 EXIT
GETBPGFL MVC   0(4,NEWADD),EMPTY    UPDATE FREE LIST HEADER
         XR    WKREG1,WKREG1
         IC    WKREG1,PAGENO
         A     WKREG1,APAGETAB      INDICATE
         NI    0(WKREG1),X'FE'      PAGE FULL
         B     GETBEXIT
*SEARCH FREE LIST OF PAGE FOR TWO CONTIGUOUS BLOCKS
GETB2B   L     11,0(NEWADD)         ADDRESS OF 1ST BLOCK IN FREE LIST
         LA    11,16(11)            ADDRESS OF POINTER CELL
         LA    10,4(11)             ADDRESS OF BLOCK CONTIGUOUS TO 1ST
         L     WKREG1,0(11)         MASK FLAGS
         N     WKREG1,MASK          BEFORE COMPARISON

         CR    10,WKREG1            CONTIGUOUS BLOCK ALSO FREE?
         BE    GETB2FND             YES->GETB2FND
         C     WKREG1,LSTPT         END OF LIST?
         BE    GETB2NO              YES->GETB2NO
         LR    NEWADD,11            SAVE POINTER CELL ADDRESS
         B     GETB2B
GETB2NO  XR    11,11                FORM FREE LIST
         IC    11,PAGENO            HEADER ADDRESS
         O     11,FRLSTAD           OF ORIGINAL PAGE
         CR    11,NEWADD            STILL IN ORIGINAL PAGE?
         BNE   GETB2REP             NO->GETB2REP
         XR    9,9                  ZERO REG 9,DETERMINES INITIALISATION
*                                   OF TRT INSTRUCTION
         B     GETBFDPG
GETB2REP LA    9,1(9)               INCR REG 9,TRT TO CONTINUE FROM LAST
*                                   POSITION SAVED
         B     GETBFDPG
*EXTRACT CONTIGUOUS BLOCKS FROM FREE LIST AND INITIALISE
GETB2FND S     11,=F'16'            ADDRESS OF NEW BLOCKS

         L     10,16(10)            ADDRESS OF FOLLOWING BLOCK
         LA    9,2                  LOOP CONTROL
         LR    WKREG1,11            INITIALISE
GETB2INN LA    WKREG2,4             LOOP CONTROL
GETB2IN  MVC   0(4,WKREG1),EMPTY    INITIALISE
         LA    WKREG1,4(WKREG1)
         BCT   WKREG2,GETB2IN
         L     WKREG1,0(WKREG1)
         BCT   9,GETB2INN
         MVC   36(4,11),LSTPT       BLOCKS
         C     10,LSTPT             PAGE NOW FULL?
         BE    GETBPGFL             YES->GETBPGFL
         ST    10,0(NEWADD)         UPDATE FREE LIST

         B     GETBEXIT
*SEARCH PAGE TABLE FOR PAGE IN SAME SECTION WITH FREE LIST
```

M SOURCE

```
GETBFDPG XR     11,11
         XR     WKREG1,WKREG1
         IC     WKREG1,PAGENO          GET PAGE TABLE
         A      WKREG1,APAGETAB        ENTRY FOR ORIGINAL
         IC     11,0(WKREG1)           PAGE (SECTN NO *2)
         LA     11,1(11)               (SECTNNO*2)+1=OFFSET IN TRTTAB
         A      11,=A(TRTTAB)          + BASE ADDRESS
         MVI    0(11),X'FF'            INITIALISE TRTTAB
         LTR    9,9                     IST TRT FOR THIS PAGE?
         BZ     GETBFRST               YES->GETBFRST
         L      1,GETBSAVE+12          ADDR TRT STOPPED AT LAST TIME
         LA     1,1(1)                 NEW START ADDRESS
         L      10,APAGETAB            NEW
         LA     10,256(10)
         SR     10,1                   TRT

         STC    10,GETBTRT+1           LENGTH
GETBTRT  TRT    0(256,1),TRTTAB        TEST FOR NON-FULL PAGE
         BC     8,GETBREQM
         ST     1,GETBSAVE+12          STORE HALT ADDRESS
         S      1,APAGETAB             GET PAGE NO
         SLL    1,12                   FORM
         O      1,FRLSTAD              FREE LIST
         LR     NEWADD,1               HEADER ADDRESS
         MVI    0(11),X'00'            RESET TRTTAB
         B      GETBCLI
GETBFRST L      1,APAGETAB             INITIALISE
         MVI    GETBTRT+1,X'FF'        TRT
         B      GETBTRT
*GET NEW PAGE AND INITIALISE IT
GETBREQM REQM   1
*ALLOCATES 1 PAGE OF VIRTUAL MEMORY,ADDRESS RETURNED IN REG 1,ERROR
*CODES IN REG 15
         LTR    15,15                   REQM CARRIED OUT?
         BZ     GETBRQOK               YES->GETBRQOK
         ERRCALL ERRMESS3               ERROR MESSAGE AND TERMD
GETBRQOK MVI    0(11),X'00'            RESET TRTTAB
         LR     11,1                   ADDRESS OF NEW PAGE
         LA     WKREG1,204             NO OF BLOCKS IN PAGE(LOOP CONTROL)
GETBINPG LA     WKREG2,4               NO OF CELLS IN BLOCK
GETBLOOP MVC    0(4,11),EMPTY          INITIALISE

         LA     11,4(11)               BLOCK
         BCT    WKREG2,GETBLOOP
         LA     10,4(11)               POINTER CELL
         ST     10,0(11)               INITIALISED
         LA     11,4(11)               NEXT BLOCK
         BCT    WKREG1,GETBINPG
         S      11,=F'4'               LAST POINTER
         MVC    0(4,11),LSTPT          CELL IN PAGE
         LA     11,4(11)               FREE LIST HEADER
         ST     1,0(11)                INITIALISED
```

M SOURCE

```
         LR     NEWADD,11          ADDRESS OF FREE LIST HEADER
         LA     WKREG1,3           INITIALISE
GETBLO   MVC    4(4,11),EMPTY      LAST

         LA     11,4(11)           CELLS IN
         BCT    WKREG1,GETBLO      PAGE
         XR     WKREG2,WKREG2      ZERO WORK
         XR     WKREG1,WKREG1      REGS
         IC     WKREG1,PAGENO      GET SECTN NO(X'2') OF ORIGINAL
         A      WKREG1,APAGETAB    PAGE,ALSO SECTN NO OF NEW PAGE
         IC     WKREG2,0(WKREG1)   +1:INDICATES FREE LIST NOT EMPTY
         LA     WKREG2,1(WKREG2)
         SLL    1,12               ISOLATE PAGE NO
         SRL    1,24               OF NEW PAGE
         A      1,APAGETAB
         STC    WKREG2,0(1)        ENTRY FOR NEW PAGE IN PAGETAB
         B      GETBCLI
```

```
M SOURCE

*INPUT PARAMETERS
*VALADD CONTAINS THE REQUIRED NOTION'S ADDRESS
*ATADD CONTAINS THE ADDRESS OF THE VALUE CELL POINTING TO THE LIST TO
*BE SEARCHED.
*OUTPUT PARAMETERS
*VALADD CONTAINS THE ADDRESS OF THE CELL CONTAINING THE REQUIRED NOTION
*OR 'LSTPT' IF THE SEARCH HAS FAILED
*C(ATADD) MAY BE CHANGED DURING ROUTINE.
*DESCRIPTION:FNDVALSU ROUTINE
*THE VALUE LIST INDICATED IS SCANNED CELL BY CELL TO SEE IF THE NOTION
*SPECIFIED IN VALADD IS PRESENT.IF IT IS,THE ADDRESS OF THE CELL THAT
*CONTAINS IT IS RETURNED TO THE CALLING ROUTINE.IF THE SCAN FAILS THE
*ROUTINE CHECKS VALADD TO SEE IF AN EMPTY CELL IS REQUIRED.IF SO THE
*GETBLK ROUTINE IS CALLED TO ALLOCATE A NEW BLOCK WHICH IS CHAINED TO
*THE EXISTING ONE BEFORE THE ADDRESS OF THE 1ST CELL IN THE NEW BLOCK
*IS PASSED BACK TO THE CALLING ROUTINE OTHERWISE THE ROUTINE RETURNS
*'LSTPT' TO INDICATE FAILURE.
*CALLS:GETBLK
*CALLED BY:SIMCREN,SIMCRER,SIMSUPR,SIMTNV,SIMLAT
FNDVALSU STM  10,11,FNDVSAVE          WORK REGS
         L    10,0(ATADD)             START OF VALUE LIST

         ST   VALADD,VALZONE          STOCK ADDRESS OF NOTION REQUIRED
         LA   10,4(10)                SKIP COUNTER CELL
         LA   11,3                    LOOP CONTROL
FNDVCOMP CLC  1(3,10),VALZONE+1            VALUE REQUIRED?(IGNORE FLAGS)
         BE   FNDVFND                 YES->FNDVFND
         LA   10,4(10)                NEXT CELL
         BCT  11,FNDVCOMP
         CLC  0(4,10),LSTPT           END OF LIST?
         BE   FNDVFAIL                YES->FNDVFAIL
         L    10,0(10)                ADDRESS OF NEXT BLOCK
         LA   11,4                    LOOP CONTROL
         B    FNDVCOMP
FNDVFAIL C    VALADD,EMPTY            EMPTY CELL REQUIRED?

         BE   FNDVGBLK                YES->FNDVGBLK
         L    VALADD,LSTPT            REPORT FAILURE
FNDVEXIT LM   10,11,FNDVSAVE          RESTORE WORK REGS
         BR   LINK                    EXIT
FNDVFND  LR   VALADD,10               ADDR OF CELL CONTAINING VALUE REQ
         B    FNDVEXIT
FNDVGBLK GBLCK FNDVSAVE+8,10,ATADD,VALADD
*SET UP PARAMETERS   AND CALL GETBLK ROUTINE
         B    FNDVEXIT
```

M SOURCE

```
*INPUT PARAMETERS
*ROOTAD CONTAINS START ADDRESS OF NOTION CONCERNED
*NATADD CONTAINS ADDRESS OF NOTION WHICH IS NATURE OF ATTRIBUTE WANTED
*OUTPUT PARAMETERS
*ATADD CONTAINS ADDRESS OF VALUE CELL OF ATTRIBUTE FOUND OR 'LSTPT' IF
*SEARCH FAILED
*ROOTAD,NATADD UNCHANGED
*IF NATURE OF ATTRIBUTE WANTED WAS SPECIAL,THE ADDRESS OF THAT ATTRIBUTE
*IN THE ROOT NOTION IS RETURNED IN ADDSAVE.
*DESCRIPTION:FNDATSUB ROUTINE
*AFTER MASKING THE FLAG BITS IN NATADD,THE ROUTINE TESTS TO SEE IF THE
*ATTRIBUTE REQUIRED IS ONE OF THE SPECIAL ONES,IF SO THE VALUE CELL OF
*THE ATTRIBUTE IS TESTED,IF IT IS EMPTY THE ROUTINE REPORTS FAILURE,
*OTHERWISE IT RETURNS THE VALUE CELL'S ADDRESS IN ATADD.
*WHEN THE ATTRIBUTE REQUIRED IS NOT A SPECIAL ONE THE ATTRIBUTE BLOCKS
*ARE SCANNED UNTIL AN ATTRIBUTE IS FOUND WITH THE GIVEN NATURE(IN WHICH
*CASE THE ADDRESS OF THE CORRESPONDING VALUE CELL IS RETURNED) OR THE
*END OF THE NOTION IS REACHED.IF THIS HAPPENS NATADD IS TESTED TO SEE
*IF AN EMPTY ATTRIBUTE IS REQUIRED IN WHICH CASE THE GETBLK ROUTINE IS
*CALLED AND AN EXTRA BLOCK IS ADDED TO THE NOTION,THE ADDRESS OF AN
*EMPTY VALUE CELL IS THEN RETURNED TO THE CALLING ROUTINE,OTHERWISE THE
*ROUTINE REPORTS FAILURE.
*CALLS:GETBLK
*CALLED BY:SIMCREN,SIMCRER,SIMSUP,SIMSUPR,SIMTMV,SIMREP
FNDATSUB STM   10,11,FNDASAVE      SAVE WORK REG CONTENTS
*NOTE:SAVE AREA=3 WORDS,1 WORD NEEDED IN ROUTINE
         N     NATADD,MASK         MASK FLAG BITS
         C     NATADD,=A(ARG4)     ATTR REQUIRED ONE OF SPECIAL ATTR?
         BNH   FNDASPEC            YES->FNDASPEC
         LA    11,36(ROOTAD)       POINTER CELL OF 2ND BLOCK
FNDALOOP LA    10,2                LOOP CONTROL
         CLC   0(4,11),LSTPT       END OF NOTION?
         BE    FNDAFAIL            YES->FNDAFAIL
         L     11,0(11)            ADDRESS OF NEXT BLOCK

FNDACOMP L     WKREG1,0(11)        NATURE OF ATTR
         N     WKREG1,MASK         MASK FLAGS
         CR    NATADD,WKREG1       NATURE OF ATTR REQUIRED?
         BE    FNDAFND             YES->FNDAFND
         LA    11,8(11)            NEXT ATTRIBUTE
         BCT   10,FNDACOMP
         B     FNDALOOP
FNDAFND  LA    ATADD,4(11)         ADDRESS OF VALUE CELL OF ATTRIBUTE
FNDAEXIT LM    10,11,FNDASAVE      RESTORE WORK REGS
         BR    LINK                EXIT
FNDAFAIL C     NATADD,EMPTYMSK     EMPTY ATTRIBUTE REQUIRED?
         BE    FNDAGBLK            YES->FNDAGBLK
FNDANOAT L     ATADD,LSTPT         REPORT FAILURE
         B     FNDAEXIT
FNDASPEC LR    10,NATADD           NATURE OF ATTRIBUTE REQUIRED

         LA    11,SPATTAB          TABLE-SPECIAL ATTRIBUTE ADDRESSES
```

N SOURCE

```
FNDAC      C       10,0(11)              ATTRIBUTE WANTED?
           BE      FNDAOFFS              YES
           LA      11,4(11)              NO:NEXT ATTRIBUTE
           B       FNDAC
FNDAOFFS   S       11,=A(SPATTAB)        TABLE
           LA      11,4(11)              ADDRESS
           C       11,=F'16'             INTO
           BL      FNDATEST              OFFSET
           LA      11,4(11)              IN ROOT NOTION
FNDATEST   AR      11,ROOTAD             ADD ROOT NOTION ADDRESS
           ST      11,ADDSAVE            SAVE ADDR OF SPECIAL ATTRIBUTE
           CLC     0(4,11),EMPTY         ATTRIBUTE EXISTS?
           BE      FNDANOAT              NO->FNDANOAT

           LR      ATADD,11              ADDRESS OF VALUE CELL OF ATTRIBUTE
           B       FNDAEXIT
FNDAGBLK   GBLCK   FNDASAVE+8,11,ROOTAD,ATADD
*ALLOCATE EMPTY BLOCK,CHAIN TO END OF NOTION,RETURN ADDRESS OF BLOCK
           LA      ATADD,4(ATADD)        ADDRESS OF VALUE CELL
           B       FNDAEXIT
```

M SOURCE

```
*INPUT PARAMETERS
*ROOTAD CONTAINS ADDRESS OF BEGINNING OF VALUE LIST
*NOTSTEND CONTAINS ADDRESS OF CURRENT POSITION IN NOTSTACK
*SW INDICATES THAT THE LIST BELONGS TO AN OCCDE ATTRIBUTE IF SET TO X'01
*IT IS SET TO X'00' OTHERWISE
*OUTPUT PARAMETERS
*NOTSTEND CONTAINS (UPDATED) POSITION IN NOTSTACK
*SW IS RESET TO X'00'
*DESCRIPTION:RETVALST ROUTINE
*AFTER INITIALISATION OF REGISTERS ETC;THE BLOCKS OF THE VALUE LIST ARE
*SCANNED CELL BY CELL.AN EMPTY CELL IS SKIPPED.WITH A NON-EMPTY CELL
*THE REFERENCE COUNTER OF THE NOTION REFERRED TO IN THE CELL IS
*DECREMENTED.IF IT BECOMES ZERO THE NOTION IS ADDED TO THE STACK OF
*NOTIONS (NOTSTACK) THAT NO LONGER EXIST AND WHOSE SPACE ALLOCATION CAN
*BE RETURNED TO THE FREE LIST.IF THE CELL BELONGS TO AN 'OCCDE' VALUE
*LIST IT IS TESTED FOR A BASIC ACTION OCCURRENCE IN WHICH CASE THE CELL
*IS SKIPPED
*AT THE END OF A BLOCK THE ROUTINE TESTS FOR THE END OF THE LIST.IF NOT
*THE END IT TESTS WHETHER THE FOLLOWING BLOCK IS IN THE SAME PAGE AS THE
*PREVIOUS ONES,IN WHICH CASE IT CARRIES ON WITH THE NEXT ONE.IF NOT,THE
*PREVIOUS BLOCKS (ALL IN THE SAME PAGE) ARE RETURNED TO THE FREE LIST
*OF THE PAGE BY THE RETBLK ROUTINE:BEFORE THE NEXT BLOCK IS DEALT WITH.
*AT THE END OF THE VALUE LIST THE REMAINING BLOCKS ARE RETURNED TO THE
*FREE LIST BEFORE EXIT FROM THE ROUTINE.
*CALLS:RETBLK
*CALLED BY:RETNOTSP
RETVALST STM   8,11,RETVSAVE        SAVE WORK REG CONTENTS
         CLI   SW,X'01'             OCCDE LIST?
*SW IS SET TO X'01' IF THE VALUE LIST BELONGS TO THE OCCDE ATTRIBUTE,TO
*X'00' OTHERWISE

         BNE   RETVLA               NO->RETVLA
         MVI   SW,X'00'             RESET INDICATOR
         MVI   RETVSW+1,X'00'       BRANCH TO NOP
RETVLA   LA    11,4(ROOTAD)         ADDRESS OF 1ST CELL
         L     10,NOTSTEND          CURRENT POSITION IN STACK
         PAGNO ROOTAD,PAGENO        INITIALISE CURRENT PAGE NO
         LA    9,3                  LOOP CONTROL:3 CELLS IN 1ST BLOCK
*TREATMENT OF A BLOCK
RETVSW   B     RETVCELL             SWITCH
         TSTFLAG (BASIC,0(11),RETVNEXT)
RETVCELL CLC   0(4,11),EMPTY        CELL EMPTY
         BE    RETVNEXT             YES->
         TSTFLAG (OBJUN,0(11),RETVNEXT)
         DECRF 11,RETVNEXT
*DECREASE REFERENCE COUNT,IF 0 ADD NOTION TO STACK
RETVNEXT LA    11,4(11)             ADDRESS OF NEXT CELL
         BCT   9,RETVSW
*END OF BLOCK/END OF VALUE LIST
         CLC   0(4,11),LSTPT        END OF LIST?
         BE    RETVTERM             YES->RETVTERM
         CPAGE 11,PAGENO,RETVSAME,RETVSAVE+16
```

M SOURCE

```
*IF BLOCK NOT IN SAME PAGE RETURN PRECEDING BLOCKS TO FREE LIST
*OTHERWISE->RETVSAME
            LR      11,ROOTAD           START ADDRESS
            PAGNO   ROOTAD,PAGENO       NEW CURRENT PAGE NO
            B       RETVSAME+4
RETVSAME    L       11,0(11)            ADDRESS OF NEXT BLOCK
            LA      9,4                 4 CELLS/BLOCK:LOOP CONTROL
            B       RETVSW
*END OF VALUE LIST
RETVTERM    ST      LINK,RETVSAVE+16    RETURN
            BAL     LINK,RETBLK         LAST BLOCKS
            L       LINK,RETVSAVE+16    TO FREE LIST
            ST      10,NOTSTEND         CURRENT POSITION IN STACK

            MVI     RETVSW+1,X'F0'      BRANCH:SKIP BASIC ACTION TEST
            LM      3,11,RETVSAVE       RESTORE REFS
            BR      LINK                EXIT
```

M SOURCE

```
*INPUT PARAMETERS
*ROOTAD CONTAINS THE START ADDRESS OF THE NOTION WHOSE SPACE ALLOCATION
*IS TO BE RETURNED TO THE FREE LIST
*OUTPUT PARAMETERS
*NONE
*DESCRIPTION:RETNOTSB ROUTINE
*THE ROUTINE SCANS THE NOTION CHECKING THE CONTENTS OF EACH CELL.WHERE
*THE CELL REFERS TO A NOTION,THE REFERENCE COUNT OF THIS NOTION IS
*DECREMENTED.WHERE THE CELL POINTS TO A LIST OF VALUES,THE RETVALST
*ROUTINE IS CALLED TO REMOVE THE VALUE LIST.AN EMPTY CELL OR AN 'OCCDE'
*CELL CONTAINING A BASIC ACTION REFERENCE IS SKIPPED.WHEN,AT THE END OF
*A BLOCK,THE ROUTINE DETECTS A CHANGE OF PAGE FOR THE NEXT BLOCK IT
*CALLS THE RETBLK ROUTINE TO REMOVE THE BLOCKS ALREADY SCANNED BEFORE
*CONTINUING WITH THE NEXT BLOCK.AT THE END OF THE NOTION IT CALLS THE
*RETBLK ROUTINE TO REMOVE THE REMAINING BLOCKS OF THE NOTION.
*EACH TIME A REFERENCE COUNT IS DECREMENTED TO 0 THE ADDRESS OF THE
*NOTION IS ADDED TO A STACK OF SUCH NOTIONS,THE ROUTINE CHECKS THIS
*STACK WHEN IT HAS FINISHED WITH 1 NOTION AND REMOVES,IN TURN,EACH
*NOTION ON THE STACK UNTIL IT IS EMPTY.'SPECIAL' NOTIONS ON THE STACK
*ARE NOT REMOVED FROM THE GRAPH.
*CALLS:RETVALST,RETBLK
*CALLED BY:SIMSUP,SIMSUPR,SIMREP
*INITIALISATION
RETNOTSB STM   8,11,RETNSAVE        SAVE WORK REG CONTENTS
*NOTE:SAVE AREA=6 WORDS,2 WORDS NEEDED FOR STORAGE BY ROUTINE
         L     10,=A(NOTSTACK)      INITIALISE NOTION STACK
         MVC   0(4,10),=F'0'        AND POINTER
RETNBEG  PAGNO ROOTAD,PGNUMB        INITIALISE PAGE NO
         LA    11,4(ROOTAD)         START OF SCAN
*REMOVE NAME BLOCK IF NOTION NAMED
         NAMBLK ROOTAD,RETN10C,KEY,DICT,RETNSAVE+16
*TREATMENT OF 1ST TWO BLOCKS
RETN10C  LA    8,2                  2 BLOCKS,LOOP CONTROL

         LA    9,3                  3 CELLS IN 1ST BLOCK,LOOP CONTROL
*OCCDE CELL
         TSLIST 11,,RETN1BA          OCCDE LIST?
         MVI   SW,X'01'             YES,SET FLAG
RETN1LST RETVC RETNSAVE+16,0,11     CALL RETVALST
         B     RETN1SKP
RETN1BA  TSTFLAG (BASIC,0(11),RETN1SKP) IF BASIC ACT OCC->SKIP
RETN1EMP CLC   0(4,11),EMPTY        CELL EMPTY?
         BE    RETN1SKP             YES
         DECRF 11,RETN1SKP DECREASE REF COUNT
RETN1OBJ  TSTFLAG (OBJUN,0(11),RETN1SKP) IF OBJ OF UN ->SKIP
         TSLIST 11,RETN1LST,RETN1EMP IF LIST->RETN1LST,ELSE->RETN1SKP
RETN1SKP LA    11,4(11)             NEXT CELL
         BCT   9,RETN1OBJ

         L     11,0(11)             ADDR NEXT BLOCK
         LA    9,4                  LOOP CONTROL:4 CELLS/BLOCK
         BCT   8,RETN1OBJ
```

M SOURCE

```
          LA    11,36(ROOTAD)        PTR CELL TO REST OF NOTION
          B     RETNPTR
*ORDINARY BLOCK
RETNVAL   B     RETNNAT              SWITCH
          TSLIST 11,,RETNOBJ         LIST?NO->RETNOBJ
          MVI   SW,X'00'             NOT OCCDE LIST
          RETVC RETNSAVE+16,0,11     CALL RETVALST
          B     RETNSKIP
RETNOBJ   TSTFLAG (OPJUN,0(11),RETNSKIP) IF OBJ OF UN->SKIP
RETNNAT   CLC   0(4,11),EMPTY        CELL EMPTY?
          BE    RETNSKIP             YES->SKIP
          DECRF 11,RETNSKIP          DECREASE REF COUNT
RETNSKIP  LA    11,4(11)             NEXT CELL
          XI    RETNVAL+1,X'F0'      FLIP SWITCH

          BCT   9,RETNVAL
RETNPTR   CLC   0(4,11),LSTPT        END OF NOTION?
          BE    RETNTERM             YES
          CPAGE 11,PGNUMB,RETNSAME,RETNSAVE+16
*IF CHANGE OF PAGE RETURN PREVIOUS BLOCKS TO FREE LIST
          LR    11,ROOTAD            NEXT BLOCK
          PAGNO ROOTAD,PGNUMB        CURRENT PAGE NO
          B     RETNSAME+4           SKIP UPDATE
RETNSAME  L     11,0(11)             NEXT BLOCK
          LA    9,4                  LOOP CONTROL:4 CELLS/BLOCK
          B     RETNVAL              LOOP
*END OF NOTION
RETNTERM  ST    LINK,RETNSAVE+16     SAVE CURRENT RETURN ADDRESS
          BAL   LINK,RETBLK          RETURN LAST BLOCKS TO FREE LIST
          L     LINK,RETNSAVE+16
          CLC   NOTSTACK,=F'0'       STACK OF NOTIONS EMPTY?

          BNE   RETNOTST             NO->RETNOTST
          LM    8,11,RETNSAVE        WORK REGS
          BR    LINK                 EXIT
RETNOTST  SH    10,=H'4'             REMOVE TOP
          L     ROOTAD,0(10)         ENTRY OF STACK
          MVC   0(4,10),=F'0'        REINITIALISE STACK
          LA    ROOTAD,0(ROOTAD)     REMOVE FLAG BITS
          C     ROOTAD,=A(ARG4)      DO NOT REMOVE
          BNH   RETNOTST             'SPECIAL' NOTIONS
          B     RETNBEG
```

```
* SOURCE

*INPUT PARAMETERS
*ROOTAD CONTAINS ADDRESS OF NOTION
*OUTPUT PARAMETERS
*ATADD CONTAINS ADDRESS OF VALUE CELL OF ATTRIBUTE CHOSEN
*NATADD CONTAINS ADDRESS OF NOTION WHICH IS NATURE OF ATTRIBUTE CHOSEN
*ROOTAD IS UNCHANGED
*DESCRIPTION:CHRATSUB ROUTINE
*THE NOTION INVOLVED IS SCANNED TO A)DISCOVER IF IT HAS ANY ATTRIBUTES,
*(IF NOT THEN 'NIL' IS RETURNED TO THE CALLING ROUTINE);B)COUNT THE NO
*OF POSSIBLE ATTRIBUTES.AS SOON AS AN ATTRIBUTE IS FOUND SCAN A) IS
*ABANDONED.WHEN THE END OF THE NOTION IS REACHED THE TASK CPU TIME IS
*OBTAINED AND DIVIDED BY THE COUNT OF SCAN B) THIS PRODUCES A'RANDOM' NO
*WHICH IS USED TO PICK OUT AN ATTRIBUTE OF THE NOTION,(IF THE ATTRIBUTE
*IS ONE OF THE SPECIAL ONES A DIFFERENT TECHNIQUE IS USED),IF THE ATTR
*IS NON-EMPTY ITS VALUE CELL ADDRESS AND NATURE ARE PASSED BACK TO THE
*CALLING ROUTINE.OTHERWISE A NEW 'RANDOM' NO IS GENERATED,ETC.IF,AFTER
*3 NOS HAVE BEEN GENERATED,AN ATTRIBUTE HAS NOT BEEN FOUND THE FSTATSUB
*ROUTINE IS CALLED TO SEARCH SYSTEMATICALLY FOR ONE
*CALLS:FSTATSUB(2 ENTRY POINTS)
*CALLED BY:SIMLAT
CHRATSUB STM   8,11,CHRASAVE       SAVE WORK REG CONTENTS
*NOTE:SAVE AREA=5 WORDS,1 WORD NEEDED IN ROUTINE
         LA    NATADD,3            INIT TRY COUNT
         XR    8,8                 INITIALISE ATTR COUNT
         LR    11,ROOTAD           START ADDRESS OF NOTION
         LA    9,2                 LOOP

         LA    10,3                CONTROLS
CHRATEST LA    11,4(11)            NEXT CELL
         CLC   0(4,11),EMPTY       EMPTY?
         BNE   CHRANEMP            NO->CHRANEMP
         BCT   10,CHRATEST
         LA    11,4(11)            POINTER CELL
         LA    10,4                LOOP CONTROL
         BCT   9,CHRATEST
         LA    8,7                 ATTRIBUTE COUNT:7 SPECIAL ATTR
CHRAEND  CLC   0(4,11),LSTPT       END OF NOTION?
         BE    CHRAEMPT            YES->CHRAEMPT
         L     11,0(11)            ADDRESS OF NEXT BLOCK
         LA    10,4                LOOP CONTROL

CHRATST  CLC   0(4,11),EMPTY       CELL EMPTY?
         BNE   CHRANEMP            NO->CHRANEMP
         LA    11,4(11)            NEXT CELL
         BCT   10,CHRATST
         LA    8,2(8)              ATTR COUNT:+2 ATTR/BLOCK
         B     CHRAEND
CHRAEMPT L     NATADD,EMPTY        RETURN NIL AS NATURE AND
         LR    ATADD,NATADD        ADDRESS OF ATTR (NO ATTR)
         LM    8,11,CHRASAVE       RESTORE REGS
         BR    LINK                EXIT
CHRANEMP LTR   8,8                 NON-EMPTY ATTRIBUTE IN 1ST 2 BLOCKS?
```

```
* SOURCE

         BNZ    CHRACONT          YES->CHRACONT
         MH     10,=H'4'          MOVE TO POINTER
         AR     11,10             CELL OF CURRENT BLOCK
         LA     8,2(8)            ATTR COUNT:+2 ATTR/BLOCK

CHRACLC  CLC    0(4,11),LSTPT     END OF NOTION?
         BE     CHRAGEN           YES->CHRAGEN
         L      11,0(11)          MOVE TO POINTER
         LA     11,16(11)         CELL OF NEXT BLOCK
         LA     8,2(8)            ATTR COUNT:+2 ATTR/BLOCK
         B      CHRACLC
CHRACONT LA     11,36(ROOTAD)     POINTER CELL OF 2ND BLOCK
         LA     8,7               ATTR COUNT:+7 SPECIAL ATTR
         B      CHRACLC
CHRAGEN  TMODE  TASKINFO          TASK INFORMATION
         L      10,TASKINFO+24    CPU TIME

         SRDA   10,32             INITIALISE FOR DIVISION
         D      10,8              DIVIDE BY NO OF ATTRIBUTES
         LA     10,1(10)          REMAINDER BETWEEN 1 & NO OF ATTR
         C      10,=F'7'          SPECIAL ATTR?
         BNH    CHRASPEC          YES->CHRASPEC
         S      10,=F'8'          -(NO OF SPEC ATTR +1)
         LA     9,36(ROOTAD)      GET ADDRESS OF
         L      9,0(9)            3RD BLOCK
         SRDA   10,32             INITIALISE FOR DIVISION
         D      10,=F'2'          DIVIDE BY NO OF ATTR/BLOCK
         LTR    11,11             SKIP 0 BLOCKS?
         BZ     CHRAMVE           YES->CHRAMVE
CHRASKIP LA     9,16(9)           SKIP TO
         L      9,0(9)            BLOCK
         BCT    11,CHRASKIP       CHOSEN

CHRAMVE  MH     10,=H'8'          OFFSET TO CHOSEN ATTR
         AR     9,10              ADDRESS OF CHOSEN ATTR
         CLC    0(4,9),EMPTY      CELL EMPTY?
         BE     CHRARETR          YES->CHRARETR
         LA     ATADD,4(9)        ADDR OF VALUE CELL OF CHOSEN ATTR
         L      NATADD,0(9)       NATURE OF CHOSEN ATTR
         LM     8,11,CHRASAVE     RESTORE REGS
         BR     LINK              EXIT
CHRARETR BCT    NATADD,CHRAGEN    UP TO 3 TRIES
         ST     LINK,CHRASAVE+16  SAVE CURRENT RETURN ADDRESS
         SR     9,10              ADDRESS OF LAST
         LR     11,9              BLOCK TRIED
         BAL    LINK,FSTATSUB     CALL ROUTINE TO SCAN NOTION AND
*RETURN ADDRESS OF FIRST NON-EMPTY ATTRIBUTE IN REG 11
CHRARET  LR     ATADD,11          ADDRESS OF ATTRIBUTE
         L      LINK,CHRASAVE+16  RESTORE RETURN ADDRESS
         LM     8,11,CHRASAVE     RESTORE REGS

         BR     LINK              EXIT
```

M SOURCE

```
CHRASPEC LR     11,10                   NO OF ATTR CHOSEN
         MH     11,=H'4'                CONVERT TO
         CH     11,=H'16'               OFFSET IN NOTION
         BL     CHRALDAT                TO ATTRIBUTE
         AH     11,=H'4'                CHOSEN
CHRALDAT AR     11,ROOTAD               ADDR OF ATTR CHOSEN
         CLC    0(4,11),EMPTY           ATTR EXISTS?
         BNE    CHRALNAT                YES->CHRALNAT
         BCT    NATADD,CHRAGEN          UP TO 3 TRIES
         STM    8,10,FSTASAVE           INITIALISE FOR
         LA     0,1                     FSTATSUB,FSTAREG ENTRY POINT
         LR     11,ROOTAD               PARAM FOR FSTATSUB
         ST     LINK,CHRASAVE+16        SAVE CURRENT RETURN ADDRESS
         BAL    LINK,FSTAREG            CALL ROUTINE TO SCAN FOR 1ST NON-
*EMPTY ATTRIBUTE,RETURNING ADDRESS IN REG 11

         B      CHPARET
CHRALNAT LR     ATADD,11                ADDR OF ATTR
         BCTR   10,0                    CALCULATE OFFSET
         MH     10,=H'4'                IN SPATTAB FOR
         A      10,=A(SPATTAD)          ADDRESS OF
         L      NATADD,0(10)            NATURE OF ATTR
         LM     8,11,CHRASAVE           RESTORE REGS
         BR     LINK                    EXIT
```

```
M SOURCE

*ERROR MESSAGE OUTPUT,FROM ERRCALL MACRO
WROUT    WROUT (1)
ERR      TERMD

WRPARAM  DS    A                    OPTION CODE AND MESSAGE ADDRESS
         DS    C                    RESERVED
         DC    AL3(ERR)             ERROR EXIT
*IOAREA FOR DICTIONARY

         DS    0F
INAREA   DS    0CL12
         DC    CL8' '

DATA     DS    CL4
*SAVE AREAS FOR REGISTERS IN ROUTINE NAMED
REPSAVE  DS    2F
*SAVE AREAS NEVER USED SIMULTANEOUSLY WITH REPSAVE
COMPSAVE EQU   REPSAVE
CRENSAVE EQU   REPSAVE
CRERSAVE EQU   REPSAVE
IFOUSAVE EQU   REPSAVE
LATSAVE  EQU   REPSAVE
SUCCSAVE EQU   REPSAVE
SUPSAVE  EQU   REPSAVE
SUPPSAVE EQU   REPSAVE
TMVSAVE  EQU   REPSAVE
RETNSAVE DS    6F
*SAVE AREAS NEVER USED SIMULTANEOUSLY WITH RETNSAVE
GETESAVE EQU   RETNSAVE
PARSAVE  EQU   RETNSAVE
RANSAVE  EQU   RETNSAVE
CHRASAVE EQU   RETNSAVE
TSTESAVE EQU   RETNSAVE
RETVSAVE DS    6F
*SAVE AREAS NEVER USED SIMULTANEOUSLY WITH RETVSAVE
ENDASAVE EQU   RETVSAVE
ENDVSAVE EQU   RETVSAVE
ESTASAVE EQU   RETVSAVE
ESTVSAVE EQU   RETVSAVE
RETSAVE  DS    2F
*SAVE AREAS FOR 3-GRAPH CELL CONTENTS
NATSAVE  DS    F
VALSAVE  DS    F
ADDSAVE  DS    F
VALZONE  DS    F
*ARGUMENTS USED IN SIMULATION ROUTINES
ESTARG   DS    F
SNDARG   DS    F
THDARG   DS    F
FTHARG   DS    F
*TABLE OF ADDRESSES OF SPECIAL NOTIONS
```

```
F SOURCE

SPATTAB  DC     A(OCCDE)
         DC     A(SUC1)
         DC     A(SUC2)
         DC     A(ARG1)
         DC     A(ARG2)
         DC     A(ARG3)
         DC     A(ARG4)
*TABLE OF BASIC ACTION SIMULATION ROUTINES

SIMTAB   DC     A(SIMCOMP)
         DC     A(SIMCREN)
         DC     A(SIMIFCU)
         DC     A(SIMLAT)
         DC     A(SIMREP)
         DC     A(SIMSUP)
         DC     A(SIMSUP)
         DC     A(SIMSUPR)
         DC     A(SIMTMV)
         DC     A(SIMCRER)
         DC     A(#ATHALT)          I/O HALT POINT ENTRY

TASKINFO DS     CL28                ZONE FOR TMODE SYSTEM MACRO
*SPECIAL CELL VALUES IN 3-GRAPH
EMPTY    DC     X'FFFFFFE'
LSTPT    DC     X'FFFFFFF'

FRLSTAD  DS     0F
         DC     XL4'FFD'
*MASKS IN CONNECTION WITH FLAG BITS

EMPTYMSK DS     0F
         DC     X'00FFFFFE'

MASK     DS     0F
         DC     X'00FFFFFF'

NOTSTACK DS     10F                 STACK FOR NOTIONS NO LONGER
*                                   REFERENCED

NOTSTEND DC     A(NOTSTACK)
STACK    DC     50X'FFFFFFE'        STACK FOR PARALLEL SUCCESSOR ACTIONS


STCKPNTR DC     A(STACK)
*EXTRN ADDRESSES
APAGETAB DC     A(PAGETAB)
*WORK ZONES

TSTPTR   DS     F
TRTTAB   DC     256X'00'
```

M SOURCE

```
VALFND   DC     X'0C'
SW       DC     X'00'
SAVE     DC     X'00'

PGNUMB   DS     C
PAGENO   DS     C
NOFLCKS  DS     C
INDIC    DS     C
WORK     DS     C
DWORD    DS     0D
         DC     PL8'0'
*MESSAGES TO USER
RLPLY    DC     CL12' '
ASK      DC     X'0C354040D1'
         DC     'PLEASE GIVE NAME OF FIRST ACTION TO BE SIMULATED'

ERRMESS2 DC     X'002B404040'
         DC     'AN ACCESSED NOTION IS COMPLETELY EMPTY'

ERRMESS3 DC     X'003D404040'
         DC     'REQM NOT CARRIED OUT,FURTHER EXPANSION IMPOSSIBLE'

ERRMESS4 DC     X'002C404040'
         DC     'NOT ENOUGH ROOM LEFT IN SUCCESSOR STACK'

MESS     DC     X'00394040D0'
         DC     'STACK OF NOTIONS TO BE RETURNED TO FREE LIST IS FULL'

ERRM10   DC     X'002A404040'
         DC     'ERROR IN PARAMETERS TO TSTBLK ROUTINE'


         END    SIMULATE
```

M SOURCE

```
                TITLE '3-GRAPH INPUT/OUTPUT ROUTINES'
#USERINT        CSECT
                PRINT NOGEN
                EXTRN NEXTACT                   AFTER HALT POINT RESUME AT NEXTACT
                EXTRN SPATTAB                   SPECIAL ATTRIBUTE NATURE ADDRESSES
                EXTRN ENDATSUB,BASE             FOR GETATTR MACRO
                ENTRY #ATHALT                   ENTRY WHEN HALT POINT REACHED
                ENTRY #RDATA                    IORETURN AFTER ERROR MESSAGES
*
BR1             EQU   3                         BASE REGISTER
BR2             EQU   4                         SIMULATOR BASE REGISTER(GETATTR)
WKREG1          EQU   14                        SHORT TERM
WKREG2          EQU   15                        WORK REGISTERS
CURRADD         EQU   12                        CURRENT POSITION IN 3-GRAPH
ATADD           EQU   5                         ATTRIBUTE ADDRESSES
NATADD          EQU   6                         NATURE OF ATTRIBUTE(GETATTR)
ROOTAD          EQU   7                         ROOT OF ATTRIBUTE(GETATTR)
PTR             EQU   6                         CURRENT POSITION IN OUTPUT LINE
LENG            EQU   7                         LENGTH OF NEXT ITEM IN OUTPUT LINE
VALPTR          EQU   11                        POSITION IN VALUE LIST
LINK            EQU   13                        LINKS BETWEEN ROUTINES
INDEX           EQU   10                        INDEX REGISTER
TRUE            EQU   X'01'                     BOOLEAN 'TRUE'
LIST            EQU   X'40'                     VALUE LIST FLAGS
BASIC           EQU   X'80'                     BASIC ACTION OCCURRENCE
NAMFL           EQU   X'80'                     NAMED NOTION FLAGS
                USING #USERINT,15               STANDARD LINKAGE
                STM   13,11,#SAVEREG            SAVE USER REGISTERS EXCEPT 12
*                                               (CURRADD)
                LA    BR1,#RDATA                NEW
                USING #RDATA,BR1                BASE
                DROP  15                        REGISTER
*REQUEST        INSTRUCTION FROM USER
#RDATA          MVI   #REQDATA,' '                    REINITIALISE
                MVC   #REQDATA+1(21),#REQDATA ZONE
                RDATA #REQUEST,#RDALRR,26
```

M SOURCE

```
*
*ANALYSE INSTRUCTION
*
#INSCHK  CLC   #REQDATA(3),='AT '   AT?
         BNE   #INSBR               NO->
         FNDNAME #REQDATA+3,#NOTION,#TRTTAB
*NAME OF NOTION ISOLATED AND STORED IN #NOTION
         LA    INDEX,4
         B     #INSTAB(INDEX)       BRANCH TO AT ROUTINE
*
#INSBR   CLC   #REQDATA(3),='BR '   BR?
         BNE   #INSRES              NO->
         FNDNAME #REQDATA+3,#NOTION,#TRTTAB
*ISOLATE NOTION NAME AND STORE IN #NOTION

         LA    INDEX,8
         B     #INSTAB(INDEX)       BRANCH TO BR ROUTINE
*
#INSRES  CLI   #REQDATA,'R'         R?
         BNE   #INSD                NO->
         LA    INDEX,12
         B     #INSTAB(INDEX)       BRANCH TO R ROUTINE
*
#INSD    CLC   #REQDATA(2),='D '     D?
         BNE   #INSERR              NO->ERROR
         FNDNAME #REQDATA+2,#ROOT,#TRTTAB
*ISOLATE NAME OF NOTION AND STORE IN #ROOT
*REG 1 CONTAINS LENGTH OF NAME STORED IN #ROOT
         A     1,=A(#REQDATA+3)     ADDRESS END OF NAME
         CLC   0(4,1),=' OF '       2ND TYPE OF D?
         LA    INDEX,16
         BNE   #INSTAB(INDEX)       NO->BRANCH TO D ROUTINE
         MVC   #NATURE,#ROOT        YES,SWITCH 1ST NOTION NAME TO
*                                   #NATURE
*ISOLATE 2ND NOTION NAME AND RETURN IN #ROOT
#DISPAT  FNDNAME 4(1),#ROOT,#TRTTAB
         LA    INDEX,20
         B     #INSTAB(INDEX)       BRANCH TO DISPLAY ATTR ROUTINE
*
*ERROR ROUTINES
#RDAERR  L     WKREG1,=F'-12'       -12
         AR    WKREG1,15            +ERROR CODE
         BZ    #RDAERR1             =0->READ TRUNCATION
         ERRCALL #RDAM2             OTHER ERROR CODES
*
#RDAERR1 WRERR #RDAM1               ERROR MESSAGE AND RETURN TO #RDATA
*ERROR IN USER INSTRUCTION
#INSERR  WRERR #ANALERR             ERROR MESSAGE AND RETURN TO #RDATA
*
*BRANCH ADDRESSES TO OUTPUT ROUTINES
*
#INSTAB  NOP   *                    MINIMUM OFFSET=4
         B     #SETHALT             SET HALT POINT
         B     #BRANCH              BRANCH TO NOTION GIVEN
         B     #RESUME              RESUME SIMULATION
         B     #PRINOT              PRINT NOTION
         B     #PRTATTR             PRINT ATTRIBUTE
```

M SOURCE

```
*
*BRANCH ROUTINE
*USER GIVES BR <NOTION NAME>;ANALYSIS PROVIDES NAME OF NOTION IN #NOTION
*ROUTINE RETURNS ADDRESS OF NEXT ACTION TO BE SIMULATED IN CURRADD
*NOTE:PREVIOUS CONTENTS OF CURRADD ARE LOST
*SIMULATION IS RESUMED IMMEDIATELY
*
#BRANCH  GETADD #NOTION,#TEMP,#NOFIND
*GETS ADDRESS OF ACTION AND RETURNS IT IN #TEMP
         L     CURRADD,#TEMP        RESET CURRADD
*NOW GO TO #RESUME
*


*RESUME ROUTINE
*USER GIVES R;ROUTINE RESTORES REGISTERS SAVED ON ENTRY (EXCEPT CURRADD
*PRESUMED UNALTERED OR RESET) AND RETURNS CONTROL TO SIMULATOR
*
#RESUME  L     15,#SAVEREG+8        ORIGINAL
         USING #USERINT,15          BASE ADDRESS
         DROP  BR1
         LM    13,11,#SAVEREG       RESTORE SIMULATOR REGS
         BR    14                   EXIT

*
*2ND ENTRY POINT PROVOKED BY ARRIVAL AT HALT POINT IN 3-GRAPH (SEE
*#SETHALT ROUTINE)
*
#ATHALT  EQU   *
         USING #ATHALT,15
         L     14,=A(NEXTACT)       RETURN ADDRESS
         STM   13,11,#SAVEREG       SAVE SIMULATOR REGISTERS
         L     BR1,=A(#RDATA)       LOAD BASE
         USING #RDATA,BR1           REGISTER
         DROP  15
         MVC   #SAVEREG+8,=A(#USERINT) NORMAL ENTRY POINT FOR #RESUME

         MVC   4(4,CURRADD),#HALTPT RESTORE ORIGINAL 'OCCDE' CELL CONT-
*                                   ENTS(CURRADD PRESUMED SET TO START O
*                                   NOTION)
         WRERR #HALTMES             MESSAGE TO USER AND RETURN TO #RDATA


*
*SET HALT POINT ROUTINE
*USER GIVES AT <NOTION NAME>;ANALYSIS PROVIDES NAME OF NOTION IN #NOTION
*ROUTINE SAVES CONTENTS OF OCCDE CELL OF NOTION IN #HALTPT AND REPLACES
*IT WITH A SPECIAL VALUE INDICATING A HALT POINT
*
#SETHALT GETADD #NOTION,#TEMP,#NOFIND      GET ADDRESS OF NOTION
         L     WKREG1,#TEMP

         MVC   #HALTPT,4(WKREG1)    SAVE OCCDE CELL CONTENTS
         MVC   4(4,WKREG1),#HALT    SET HALT POINT
         B     #RDATA               FURTHER INSTRUCTIONS FROM USER
```

```
M SOURCE

*
*PRINT ATTRIBUTE ROUTINE
*USER GIVES D <NOTION NAME1> OF <NOTION NAME2>
*ANALYSIS PROVIDES NAME OF ROOT NOTION (NOTION NAME2) IN #ROOT AND NAME
*OF NATURE NOTION (NOTION NAME1) IN #NATURE
*ROUTINE OUTPUTS THE ROOT,NATURE AND ALL VALUES OF THE GIVEN ATTRIBUTE,
*THEN RETURNS TO #RDATA FOR FURTHER INSTRUCTION
*
#PRTATTR GETADD #ROOT,#ROOTADD,#NOFIND
         GETADD #NATURE,#NATADDR,#NOFIND
*GET ADDRESSES OF ROOT AND NATURE NOTIONS IN #ROOTADD AND #NATADDR RESP
         GETATTR #ROOTADD,#NATADDR,ATADD
*FIND ATTRIBUTE ASKED FOR
         C     ATADD,LSTPT          ATTR EXISTS?
         BE    #NOATTR              NO->
*INITIALISE OUTPUT ZONE POINTERS
         LA    PTR,#OUTZONE         POSITION AND
         LA    LENG,8               LENGTH OF 1ST NAME
         PUTNAM PTR,#ROOT,LENG
*PUTS NAME OF ROOT NOTION IN OUTPUT ZONE
         PUTNAM PTR,#NATURE,LENG
*PUTS NAME OF NATURE NOTION IN OUTPUT ZONE
         TSLIST ATADD,#VALST        LIST OF VALUES->#VALST
*ONE VALUE ONLY
         GETNAM ATADD,#NOTION,NIL
*GETS NAME OF VALUE NOTION  IN #NOTION
         PUTNAM PTR,#NOTION,LENG
*PUTS NAME OF VALUE NOTION IN OUTPUT ZONE
         PUTLINE #OUTLINE,PTR,#TERMD
*WRITE LINE TO SYSOUT FILE
         B     #RDATA               LOOP
*VALUE LIST
#VALST   L     VALPTR,0(ATADD)      START OF LIST
         BAL   LINK,#NEXTVAL        ROUTINE TO OUTPUT VALUE LIST
         B     #RDATA               LOOP
#NO ATTRIBUTE
#NOATTR  WRERR #NOATTM              ERROR MESSAGE AND RETURN TO #RDATA
```

M SOURCE

```
*
*PRINT NOTION ROUTINE
*USER GIVES D <NOTION NAME>
*ANALYSIS GIVES NAME OF ROOT NOTION IN #ROOT
*ROUTINE OUTPUTS ALL NON-EMPTY ATTRIBUTES OF THE NOTION THEN RETURNS TO
*#RDATA FOR FURTHER INSTRUCTION
*
#PRINOT  GETADD #ROOT,#ROOTADD,#NOFIND
*GET ADDRESS OF ROOT NOTION IN #ROOTADD
         LA    LENG,8                 INITIALISE OUTPUT
         LA    PTR,#OUTZONE           ZONE POINTERS
         PUTNAM PTR,#ROOT,LENG
*PUTS NAME OF ROOT NOTION IN OUTPUT ZONE
*
*INITIALIZATION FOR OUTPUT OF SPECIAL ATTRIBUTES
         L     ATADD,#ROOTADD         ADDR OF

         LA    ATADD,4(ATADD)         CCODE CELL
         L     8,=A(SPATTAB)          1ST ENTRY IN SPATTAB
         LA    9,3                    LOOP CONTROL:3 CELLS IN 1ST BLOCK
         LA    10,2                   LOOP CONTROL:2 SPEC ATTR BLOCKS
#SPATLO  OUTATT ATADD,#NXTCEL,#SPLIST,SPEC,8
*OUTPUT SPECIAL ATTRIBUTE AT ATADD,IF NONE->#NXTCEL,IF LIST->#SPLIST

         LA    PTR,9(PTR)             REINITIALISE POINTER IN OUTPUT ZONE
#NXTCEL  LA    ATADD,4(ATADD)         NEXT CELL
         LA    8,4(8)                 NEXT ENTRY IN SPATTAB
         BCT   9,#SPATLO
*END OF A BLOCK
         L     ATADD,0(ATADD)         NEXT BLOCK
         LA    9,4                    LOOP CONTROL:4CELLS IN 2ND BLOCK
         BCT   10,#SPATLO
*END OF SPECIAL ATTRIBUTES
         C     ATADD,LSTPT            END OF NOTION?
         BE    #ENDNOT                 YES->
         B     #ATTR                  NO->CONTINUE
*SPECIAL ATTRIBUTE VALUE LIST
#SPLIST  L     VALPTR,0(ATADD)        START OF LIST
         BAL   LINK,#NEXTVAL          ROUTINE TO OUTPUT VALUE LIST
         B     #NXTCEL                NEXT CELL
*
*OTHER ATTRIBUTES OF ROOT NOTION
*
#ATTR    LA    9,2                    LOOP CONTROL:2 ATTR/BLOCK
*                                     ATADD ASSSUMED OK
#NEXTATT OUTATT ATADD,#SKIPATT,#LISTV,NORM
*OUTPUT NEXT ATTRIBUTE,IF EMPTY->#SKIPATT,IF LIST->#LISTV

         LA    PTR,9(PTR)             REINITIALISE POINTER
         LA    ATADD,4(ATADD)         NEXT ATTRIBUTE
         BCT   9,#NEXTATT
*END OF BLOCK
```

```
M SOURCE

#ENDEL   CLC   0(4,ATADD),LSTPT     END OF NOTION?
         BE    #ENDNOT               YES->#ENDNOT
         L     ATADD,0(ATADD)        NO:ADDRESS OF NEXT BLOCK
         B     #ATTR
*
#SKIPATT LA    ATADD,8(ATADD)        NEXT ATTRIBUTE
         BCT   9,#NEXTATT
         B     #ENDPL
*
*LIST OF VALUES
#LISTV   L     VALPTR,0(ATADD)       START OF LIST
         BAL   LINK,#NEXTVAL         ROUTINE TO OUTPUT VALUE LIST
         LA    ATADD,4(ATADD)        NEXT ATTRIBUTE

         BCT   9,#NEXTATT
         B     #ENDBL
*END OF NOTION
#ENDNOT  CLI   #OUTZONE,' '          LINE EMPTY?
         BE    #RDATA                YES->EXIT
         PUTLINE #OUTLINE,PTR,#TERMD NO->OUTPUT LINE

         B     #RDATA
```

```
N SOURCE

*INPUT PARAMETERS
*VALPTR CONTAINS START ADDRESS OF VALUE LIST
*PTR INDICATES CURRENT POSITION IN OUTPUT LINE
*OUTPUT PARAMETERS
*PTR IS REINITIALISED FOR A NEW OUTPUT LINE
*DESCRIPTION:#NEXTVAL ROUTINE
*THE ROUTINE GETS OR CREATES THE NAME FOR EACH ELEMENT IN THE LIST AND
*STORES IT IN THE OUTPUT LINE.WHEN THE LINE IS FULL AND/OR IT REACHES
*THE END OF THE LIST THE ROUTINE OUTPUTS THE CURRENT LINE AND REINITIAL
*ISES THE OUTPUT ZONE.
*CALLS:0
*CALLED BY:#PRINOT,#PRTATTR
*
#NEXTVAL ST    8,#NEXTSAV              SAVE WORK REGISTER
         LA    8,3                     NO OF CELLS/CLOCK
         LA    VALPTR,4(VALPTR)        NEXT (1ST) VALUE CELL
#TSEMP   CLC   0(4,VALPTR),EMPTY       CELL EMPTY?
         BE    #SKIP                   YES->#SKIP
         GETNAM VALPTR,#NOTION         NAME OF VALUE NOTION
         LENGTH #NOTION,LENG,#NROOM,#TRTTAB,PTR,#OUTZONE
*CALCULATES LENGTH OF NAME AND SETS #NROOM=1 IF NEW LINE NEEDED
         CLI   #NROOM,TRUE             NO ROOM?
         BE    #PUTL                   NONE->
#PUTN    PUTNAM PTR,#NOTION,LENG
*PUTS NAME OF VALUE NOTION IN OUTPUT ZONE
#SKIP    LA    VALPTR,4(VALPTR)        NEXT CELL
         BCT   8,#TSEMP
*END OF BLOCK
         CLC   0(4,VALPTR),LSTPT       END OF LIST?
         BE    #ENDOFLS                YES->
         L     VALPTR,0(VALPTR)        NEXT BLOCK
         LA    8,4                     NO OF CELLS/BLOCK
         B     #TSEMP                  LOOP
*
#PUTL    PUTLINE #OUTLINE,PTR,#TERMD   OUTPUT CURRENT LINE TO SYSOUT
         LA    PTR,18(PTR)             RESET POINTER PAST ROOT,NATURE POS
         MVI   #NROOM,X'00'            RESET #NROOM
         B     #PUTN
*
#ENDOFLS PUTLINE #OUTLINE,PTR,#TERMD   OUTPUT LAST LINE
         LA    PTR,9(PTR)              PTR TO SKIP ROOT POSITION
         L     8,#NEXTSAV              RESTORE WORK REG
         BR    LINK                    EXIT
```

```
M SOURCE

   *SAVE AREAS

   #SAVEREG DS    16F                    SIMULATOR REGS SAVE AREA
   #NEXTSAV DS    F                      SAVE AREA,#NEXTVAL ROUTINE
   #GETSAVE DS    2F                     SAVE AREA,GETATTR MACRO
   *INPUT/OUTPUT AREAS
   #OUTLINE DS    0CL120
            DC    X'0000404041'
   #OUTZONE DC    CL115' '



   #REQUEST DS    0CL26                  INPUT AREA
            DS    CL4                    FOR USER
   #REQDATA DS    CL22                   INSTRUCTIONS
   *ERROR MESSAGES
   #RDAM2   DC    X'0026404000'
            DC    'RDATA ERROR AT #RDATA,TERMINATING'
   #RDAM1   DC    X'0C2F404000'
            DC    'INPUT TRUNCATED,MESSAGE TOO LONG,RE-ENTER'

   #NOATTM  DC    X'0027404041'
            DC    'ATTRIBUTE REQUESTED DOES NOT EXIST'

   #ANALERR DC    X'0025404000'
            DC    'INSTRUCTION CANNOT BE IDENTIFIED'
   *ZONES FOR NOTION NAMES

   #NOTION  DC    CL8' '
   #ROOT    DC    CL8' '
   #NATURE  DC    CL8' '
   *ZONES FOR NOTION ADDRESSES

   #TEMP    DS    F
   #ROOTADR DS    F
   #NATADDR DS    F
   *
   #HALTPT  DS    F                      SAVEAREA-HALT POINT
   #HALT    DC    X'8F000028'            HALT POINT INDICATOR
   #HALTMES DC    X'000F404000'          HALT POINT MESSAGE
            DC    'HALT POINT'
   *BASIC ACTION TABLE
   #BASACT  DC    'COMP'
            DC    'CREM'
            DC    '    '
            DC    'LAT '
            DC    'REF '
            DC    'SUP '
            DC    '    '
            DC    'SUPR'
            DC    'TMV '
```

```
M SOURCE

          DC     'CREA'

          DS     0F
EMPTY     DC     X'FFFFFFFE'
LSTPT     DC     X'FFFFFFFF'
#NRROM    DC     X'00'                    ROOM-IN-LINE INDICATOR

#DWORD    DS     0D
          DC     PL8'0'
*TRT TABLE FOR ISOLATING NOTION NAMES
#TRTTAB   DC     X'FF'

          DS     CL63
          DC     X'FF'

          DS     CL191
          ORG    #TRTTAB+1                REDEFINE UNUSED PART OF TABLE
*TR TABLE FOR CREATING NAMES
#TAB1     DC     X'FAFBFCFDFEFF'

          DS     CL41
          DC     '0123456789'

          ORG
*TR TABLE FOR NAMES->ADDRESSES (GETADD)
#TAB2     DC     '0123456789ABCDEF'

*OUTPUT MESSAGES,EPICALL & WRERR MACRO CALLS
WROUT     WROUT (1)
EPR       TERMD

WRPARAM   DS     A                        PARAM LIST FOR WROUT
          DS     C
          DC     AL3(EPR)
WROUTRET  WROUT (1)
          P      #RDATA
#TERMD    TERMD


          END    #USERINT
```