A NOTE ON STAR-FREE EVENTS

By

Albert R. Meyer

Carnegie-Mellon University
Pittsburgh, Pennsylvania
February, 1968

This work was supported by the Advanced Research Projects Agency of the Office of the Secretary of Defense (SD-146) and is monitored by the Air Force Office of Scientific Research. Distribution of this document is unlimited.

ABSTRACT

A short proof of the equivalence of star-free and group-free regular events is possible if one is willing to appeal to the Krohn-Rhodes machine decomposition theorem.

1. INTRODUCTION

The star-free events are the family of regular events expressible in the extended language of regular expressions (using intersection and complementation, as well as union and concatenation of events) without the use of the Kleene star (closure) operator. The equivalence of the star-free and group-free events was first proved by Schützenberger [1966]. Papert and McNaughton [1966] show that the star-free events are precisely the events definable in McNaughton's L-language, and are thereby able to establish the above equivalence without extensive use of the properties of finite semigroups. However, if one is willing to appeal to the machine decomposition theorem of Krohn and Rhodes,the equivalence of star-free, group-free, and also noncounting regular events can be proved more simply. We present such a proof in this note.

2. PRELIMINARIES

We assume the reader is already familiar with regular events and finite automata. Our notation follows Yoeli [1965] and Ginzburg [1968]. In particular, if f and g are functions from a set S into itself, arguments are written on the left (so that sf = f(s)), and the composition $f \circ g$ means that f is applied first (so that $s(f \circ g) = (sf)g$).

A <u>semiautomaton</u> is a triple $A = \langle Q^A, \Sigma^A, M^A \rangle$ with Q^A a finite set (of <u>states</u>), Σ^A a finite set (of <u>inputs</u>), and M^A a set of functions $M^A_\sigma: Q^A \to Q^A$ indexed by $\sigma \in \Sigma^A$. The mapping M^A_σ is abbreviated "σ^A". The element $q\sigma^A \in Q^A$ is the <u>next state</u> of $q \in Q^A$ under input $\sigma \in \Sigma^A$. For $x \in (\Sigma^A)^*$ the mapping $x^A: Q^A \to Q^A$ is defined inductively: Λ^A is the identity map on Q^A where Λ is the null word in $(\Sigma^A)^*$, and if $x = y\sigma$ for $y \in (\Sigma^A)^*$ and $\sigma \in \Sigma^A$, then x^A is $y^A \circ \sigma^A$. Hence, $(xy)^A = x^A \circ y^A$ for all $x, y \in (\Sigma^A)^*$. For $x \in (\Sigma^A)^*$ and integers $k \geq 0$, x^k is the concatenation of x with itself k times; $x^0 = \Lambda$ by convention. Clearly, $(x^k)^A = (x^A)^k$ = the composition of x^A with itself k times. The (necessarily finite) set of distinct mappings x^A for $x \in (\Sigma^A)^*$ form a semigroup G^A under composition. G^A is called the <u>semigroup of A</u>.

Let A and B be semiautomata. B is a <u>subsemiautomaton</u> of A providing $\Sigma^B \subset \Sigma^A$, $Q^B \subset Q^A$ and the mapping σ^B is the restriction of σ^A to Q^B for each $\sigma \in \Sigma^B$. B is a <u>homomorphic image</u> of A providing that $\Sigma^A = \Sigma^B$ and there is an onto mapping $\eta: Q^A \to Q^B$ such that $\eta \circ \sigma^B = \sigma^A \circ \eta$ for each $\sigma \in \Sigma^A$. The mapping η is called a <u>homomorphism</u> of A onto B. A covers B, in symbols "$A \geq B$" if and only if B is a homomorphic image of a subsemiautomaton of A.

An <u>automaton</u> is a quintuple $\hat{A} = \langle Q^A, \Sigma^A, s^A, F^A, M^A\rangle$ where $A = \langle Q^A, \Sigma^A, M^A\rangle$ is a semiautomaton, called the <u>semiautomaton</u> of $\hat{A}$, s^A is an element of Q^A called the <u>start state</u>, and F^A is a subset of Q^A called the <u>final states</u>. The <u>event accepted by $\hat{A}$</u> is $\{x \in (\Sigma^A)^* \mid s^A x^A \in F^A\}$. This definition of automaton is merely a notational variant of the usual finite state acceptor (cf. Rabin and Scott [1959]), and the events accepted by such automata are precisely the regular events.

3. STAR-FREE AND NONCOUNTING EVENTS

The star-free events are defined inductively as follows:

<u>Definition 1</u>. Let Σ be a finite set (of inputs). The singleton $\{\sigma\}$ is a star-free event over Σ. If $U, V \subset \Sigma^*$ are star-free events over Σ,

then $U \cup V$, $\bar{U}$ (the complement of U relative to Σ^*), and UV (the concatenation of U and V) are star-free events over Σ. An event is star-free over Σ only by implication from the preceding clauses.

By DeMorgan's law, $U \cap V = \overline{\bar{U} \cup \bar{V}}$ and so star-free events are also closed under intersection. Since the regular events over Σ include the singletons and are closed under union, relative complementation, and concatenation, it follows that every star-free event is regular.

Definition 2. (Papert-McNaughton) A regular event $U \subset \Sigma^*$ is a noncounting regular event over Σ if and only if there is an integer $k_U \geq 0$ such that for all x, y, $z \in \Sigma^*$

$$x\,y^{k_U}z \in U \Leftrightarrow x\,y^{k_U+1}\,z \in U.$$

Intuitively, an automaton accepting a noncounting event U need never count (even modulo any integer greater than one) the number of consecutive occurrences of any word y once k_U consecutive y's have occurred in an input word.

Lemma 1. (Paper-McNaughton) Every star-free event is a noncounting regular event.

Proof. The singleton $\{\sigma\}$ is trivially a noncounting regular event for every $\sigma \in \Sigma$ (choose $k_{\{\sigma\}} = 2$), so it is sufficient to show that if U and V are noncounting regular events over Σ, then so are $U \cup V$, $\bar{U}$, and UV.

Let $k = \max\{k_U, k_V\}$. Then for any x, y, $z \in \Sigma^*$, $xy^kz \in U \cup V$ $\Leftrightarrow xy^{k_U}(y^{k-k_U}z) \in U$ or $xy^{k_V}(y^{k-k_V}z) \in V \Leftrightarrow xy^{k_U+1}(y^{k-k_U}z) \in U$ or

$xy^{k_V+1}(y^{k-k_V}z) \in V \Leftrightarrow xy^{k+1}z \in U \cup V$. Thus, $U \cup V$ is a noncounting regular event with $k_{U\cup V} = \max\{k_U, k_V\}$.

Similarly, $xy^{k_U}z \in \bar{U} \Leftrightarrow xy^{k_U}z \notin U \Leftrightarrow xy^{k_U+1}z \notin U \Leftrightarrow xy^{k_U+1}z \in \bar{U}$, so that $\bar{U}$ is a noncounting regular event with $k_{\bar{U}} = k_U$.

Finally, let $k = 2\cdot\underline{\max}\{k_U,k_V\}+1$ and suppose $xy^k z \in U\,V$. Then $xy^k z = uv$ for some $u \in U$, $v \in V$, and it must be the case that either $u = xy^{k/2}w$ for some $w \in \Sigma^*$, or that $v = w'y^{k/2}z$ for some $w' \in \Sigma^*$. In the first case, $u = xy^{k/2}w = xy^{k_U}(y^{k/2-k_U}w) \in U$ implies that $xy^{k_U+1}(y^{k/2-k_U}w) = xy^{k/2+1}w \in U$ since U is noncounting. In the second case, $v = w'y^{k/2}z \in V$ similarly implies that $w'y^{k/2+1}z \in V$. Hence, in either case $xy^{k+1}z \in UV$. Conversely, if $xy^{k+1}z \in UV$ the argument can clearly be reversed to show that $xy^k z \in UV$. Thus, UV is a noncounting regular event with $k_{UV} = 2\cdot\underline{\max}\{k_U,k_V\}+1$. Q.E.D.

If U is a noncounting regular event over Σ and $\sigma \in \Sigma$, then $\sigma^{k_U} \in U$ implies that U contains all words in σ^* of length at least k_U. Therefore, either $\overline{U \cap \sigma^*}$ or $U \cap \sigma^*$ is a finite event. The regular event $(\sigma\sigma)^*$ is neither finite nor has finite complement, which proves:

<u>Corollary 1</u>. The noncounting (and hence the star-free) regular events are a proper subfamily of the regular events.

4. GROUP-FREE EVENTS

Associated with any event $U \subset \Sigma^*$ is a congruence relation, $\equiv \pmod{U}$, on Σ^* defined for $w, y \in \Sigma^*$ by:

$$w \equiv y \pmod{U} \Leftrightarrow (\forall x, z \in \Sigma^*)[x\,w\,z \in U \Leftrightarrow x\,y\,z \in U].$$

Noncounting regular events are thus those regular events U such that $y^{k_U} \equiv y^{k_U+1} \pmod{U}$ for all $y \in \Sigma^*$.

The relation between this congruence and automata is an immediate consequence of the familiar theorems of Nerode and Myhill (cf. Rabin and Scott [1959]): if U is a regular event, then there is an automaton $\hat{A}$ accepting U (viz., the reduced automaton accepting U) such that $x \equiv y \pmod{U} \Leftrightarrow x^A = y^A$.

Definition 3. A subgroup of a semigroup S is a subsemigroup of S whose elements form an abstract group under multiplication in S. A semigroup is group-free if and only if all its subgroups are isomorphic to the trivial group with one element. A semiautomaton is group-free if and only if the semigroup of the semiautomaton is group-free. A regular set U is group-free if and only if there is an automaton $\hat{A}$ accepting U such that the semiautomaton A of $\hat{A}$ is group-free.

Lemma 2. Let S be a semigroup. If there is an integer $k \geq 0$ such that $s^k = s^{k+1}$ for all $s \in S$, then S is group-free.

Proof. Let G be a subgroup of S, and let g be an element of G. Then $g^k = g^{k+1}$ implies $e = g^k(g^{-1})^k = g^{k+1}(g^{-1})^k = g$ where g^{-1} is the inverse of g in G and e is the identity of G. Hence, $G = \{e\}$ is the trivial group. Q.E.D.

Corollary 2. Every noncounting regular event is a group-free regular event.

Proof. If U is a noncounting regular event, then $y^{k_U} \equiv y^{k_U+1} \pmod{U}$

implies that $(y^{k_U})^A = (y^{k_U+1})^A$ in the reduced automaton $\hat{A}$ accepting U. Hence, $(y^A)^{k_U} = (y^A)^{k_U+1}$ for every element $y^A \in G^A$, and G^A is group-free by lemma 2.

Q.E.D.

5. DECOMPOSITION INTO RESETS

The machine decomposition theorem of Krohn and Rhodes supplies the key step in the proof that group-free events are star-free.

Definition 4. Let A and B be semiautomata and $\omega: Q^A \times \Sigma^A \to \Sigma^B$. The cascade product $A \overset{o}{\omega} B$ of A and B with mapping ω is the semiautomaton C with $Q^C = Q^A \times Q^B$, $\Sigma^C = \Sigma^A$ and σ^C for $\sigma \in \Sigma^C$ defined for all $s^A \in Q^A$, $s^B \in Q^B$ by:

$$\langle s^A, s^B \rangle \sigma^C = \langle s^A \sigma^A, s^B (\langle s^A, \sigma \rangle \omega)^B \rangle.$$

A cascade product of three or more automata is defined by association to the left, e.g., a cascade product of semiautomata A, B, and C is any semiautomaton $(A \overset{o}{\omega}_1 B) \overset{o}{\omega}_2 C$ for any mappings ω_1 and ω_2 with appropriate domain and range.

Definition 5. A semiautomaton R is a reset providing $Q^R = \{1,2\}$, and Σ^R is the union of three mutually exclusive sets Σ_1^R, Σ_2^R, Σ_I^R such that: $\sigma \in \Sigma_1^R \Rightarrow$ range $(\sigma^R) = \{1\}$; $\sigma \in \Sigma_2^R \Rightarrow$ range $(\sigma^R) = \{2\}$; and $\sigma \in \Sigma_I^R \Rightarrow \sigma^R$ = the identity on Q^R.

The following weak form of the decomposition theorem is sufficient for our purposes (for a constructive proof of the general theorem see Ginzburg [1968]):

Theorem. (Krohn-Rhodes) Every semiautomaton A is covered by a cascade product of semiautomata $A_1, A_2, \ldots, A_n$ such that for $1 \leq i \leq n$, A_i

is a reset or else G^{A_i} is a non-trivial homomorphic image of a subgroup of G^A.

Since the trivial group has only itself as a homomorphic image, the following lemma is immediate:

Lemma 3. Every group-free semiautomaton is covered by a cascade product of resets.

Corollary 3. Every group-free regular event is accepted by an automaton whose semiautomaton is a cascade product of resets.

Proof. Let $\hat{A}$, with group-free semiautomaton A, be an automaton accepting a group-free regular event U. By lemma 3 and the definition of covering, A is the image under a homomorphism η of a subsemiautomaton of a cascade product C of resets. There is no loss of generality in assuming that $\Sigma^A = \Sigma^C$, since the subsemiautomaton of C obtained by restricting Σ^C to Σ^A is also a cascade product of resets which covers A. Choose any $s^C \in Q^C$ such that $s^C\eta = s^A$ (the start state of $\hat{A}$) and define $F^C = \{q \in Q^C \mid q\,\eta \in F^A\}$. Then for any $x \in (\Sigma^A)^*$, $x \in U \Leftrightarrow s^A x^A \in F^A \Leftrightarrow s^C\eta\, x^A \in F^A \Leftrightarrow s^C x^C \eta \in F^A \Leftrightarrow s^C x^C \in F^C$. Hence, the automaton $\hat{C}$ with semiautomaton C, start state s^C, and final states F^C is the required automaton accepting U. Q.E.D.

6. THE MAIN THEOREM.

The behavior of cascades of resets can be described in terms of star-free events using

<u>Definition 6</u>. For a semiautomaton A and states $p,\ q \in Q^A$, the set A_{pq} of <u>p-q-inputs</u> is $\{x \in (\Sigma^A)^* \mid px^A = q\}$.

<u>Lemma 4</u>. Let $C = B \overset{\circ}{\omega} R$ with B a semiautomaton, R a reset, and ω: $Q^B \times \Sigma^B \rightarrow \Sigma^R$. If B_{pq} is a star-free event (over Σ^B) for all $p,\ q \in Q^B$, then C_{ab} is a star-free event (over $\Sigma^C = \Sigma^B$) for all $a,\ b \in Q^C$.

<u>Proof</u>. Write "Σ" for the (equal) sets Σ^B and Σ^C. By the definition of cascade product, the first component of $<p,\ 1>\ y^C$ is simply py^B for any $p \in Q^B$, $y \in \Sigma^*$. Since R is a reset, in order for the second component of $<p,\ 1>\ y^C$ to be 2, R must receive an input $<r,\ \sigma>\ \omega \in \Sigma_2^R$ for some $r \in Q^B$, $\sigma \in \Sigma$.

Suppose $x \in C_{<p,1><q,2>}$. Then $px^B = q$ and so $x \in B_{pq}$, but also x must equal $y\ \sigma\ z$ for some $y,\ z \in \Sigma^*$, $\sigma \in \Sigma$ such that: $py^B = r$ for some $r \in Q^B$ and $<r,\ \sigma>\ \omega \in \Sigma_2^R$. Choose the shortest z such that $x = y\ \sigma\ z$ for y and σ satisfying the preceding conditions. Then no prefix of z causes R to receive an input $<s,\ \delta>\ \omega \in \Sigma_1^R$ (where $s \in Q^B$, $\delta \in \Sigma$), i.e., $z \notin B_{r\sigma^B,s}\,\delta\ \Sigma^*$.

Conversely, if $py^B = r$ for $<r,\ \sigma>\ \omega \in \Sigma_2^R$ and $z \notin B_{r\sigma^B,s}\ \delta\ \Sigma^*$ for any $<s,\delta>\ \omega \in \Sigma_1^R$, then $y\sigma z \in C_{<p,1><q,2>}$ providing $y\sigma z \in B_{pq}$. Altogether, one has:

$$C_{<p,1><q,2>} = B_{pq} \cap [\cup\ B_{pr}\ \sigma\ \overline{(\cup\ B_{r\sigma^B,s}\ \delta\ \Sigma^*)}]$$

the lefthand union being over all $r \in Q^B$, $\sigma \in \Sigma$, such that $<r,\sigma>\omega \in \Sigma_2^R$, and the righthand union being over all $s \in Q^B$, $\delta \in \Sigma$ such that $<s,\delta>\omega \in \Sigma_1^R$.

The unions in the expression for $C_{<p,1><q,2>}$ are finite, and Σ^* is a star-free event ($\Sigma^* = \bar{\emptyset}$ and $\emptyset = \{\sigma\} \cap \overline{\{\sigma\}}$), so that $C_{<p,1><q,2>}$ is a star-free event. The set of $x \in C_{<p,1><q,1>}$ is precisely the set of

$x \in \Sigma^*$ such that $px^B = q$ and $x \notin C_{<p,1><q,2>}$, i.e., $C_{<p,1><q,1>} = B_{pq} \cap \overline{C_{<p,1><q,2>}}$, and so $C_{<p,1><q,1>}$ is also a star-free event.

Since the argument is symmetric in states 1 and 2 of Q^R, C_{ab} is a star-free event for all a, $b \in Q^C$. Q.E.D.

Lemma 5. If C is a cascade product of resets, then C_{ab} is a star-free event for all $a,b \in Q^C$.

Proof. Let R be a reset and B a semiautomaton such that $Q^B = \{p\}$ and $\Sigma^B = \Sigma^R$. For $\sigma \in \Sigma^B$, define ω: $Q^B \times \Sigma^B \to \Sigma^R$ by the condition $<p,\sigma>\omega = \sigma$. In this trivial case of cascade product, $R_{ij} = (B \,\omega\, R)_{<p,i><p,j>}$ for all i, $j \in Q^R$. Since $B_{pp} = (\Sigma^B)^*$ is star-free, lemma 4 implies that R_{ij} is star-free.

The rest of the proof follows immediately by lemma 4 and induction on the number of resets in C. Q.E.D.

Corollary 4. Every event accepted by an automaton $\hat{A}$, whose semiautomaton A is a cascade product of resets, is a star-free event.

Proof. Let $a \in Q^A$ be the start state of $\hat{A}$, and F^A the final states. The event accepted by $\hat{A}$ is $\bigcup_{b \in F^A} A_{ab}$ which is a star-free event since the union is finite and A_{ab} is star-free by lemma 5. Q.E.D.

This completes the proof of the following

Theorem. (Schützenberger, Papert-McNaughton) The following are equivalent for events $U \subset \Sigma^*$:

1) U is a star-free event.

2) U is a noncounting regular event.

3) U is a group-free event.

4) U is accepted by a cascade product of resets.

References

1. Ginzburg, A., Algebraic Theory of Automata, to appear (1968).

2. Papert, S. and R. McNaughton, "On topological events", Theory of Automata, University of Michigan Engineering Summer Conferences (1966).

3. Rabin, M. O. and D. Scott, "Finite automata and their decision problems", IBM J. Res. and Dev., 3,2 (April, 1959), 114-125.

4. Schützenberger, M. P., "On a family of sets related to McNaughton's L-language", Automata Theory, Caianiello, E. R. (ed.), Academic Press, N. Y. (1966), 320-324.

5. Yoeli, M., "Generalized Cascade Decompositions of Automata", JACM, 12, 3 (July, 1965), 411-423.

SCOPE USER MANUAL

By

Alan H. Bond

1968

Carnegie-Mellon University
Pittsburgh, Pennsylvania
February 6, 1968

This work was supported by the Advanced Research Projects Agency of the Office of the Secretary of Defense (SD-146) and is monitored by the Air Force Office of Scientific Research.

CONTENTS

FIGURES

1. PREFACE

THIS MANUAL DESCRIBES HOW TO USE THE SCOPES. IT IS CONCERNED MAINLY WITH SOFTWARE, AS THE HARDWARE IS TREATED DEFINITIVELY IN THE CARNEGIE TECH. DOCUMENT: 'A VISUAL DISPLAY SYSTEM SUITABLE FOR TIMESHARED USE' BY QUATSE, JESSE T., LATEST VERSION DECEMBER 1966, OBTAINED FROM THE COMPUTATION CENTER DOCUMENTATION STAFF. IN ORDER TO USE THE SCOPES, IT IS SUFFICIENT TO READ THE QUATSE DOCUMENT AND THE SCOPE USERS MANUAL. THE QUATSE MANUAL SHOULD BE READ FIRST. A BRIEF DESCRIPTION OF HARDWARE CONCEPTS AND TERMS IS GIVEN IN SECTION 3 OF THIS MANUAL.

THE SCOPES ARE SITUATED IN ROOM PH18A, COMPUTATION CENTER, TELEPHONE EXTENSION 27. THEY ARE ON WHEN TELETYPES ARE ON, USUALLY 10:00 AM TO MIDNIGHT AND HAVE NORMAL TELETYPE TURN-ROUND TIME, EXCEPT THAT PROGRAMS SUBMITTED FROM SCOPES RUN AT THE BEGINNING OF THE WAIT TIME, I.E., IMMEDIATELY, TO ALLOW THE USER TO BE PRESENT AT RUN TIME AND TO INTERACT WITH HIS PROGRAM. AT PRESENT ONLY 3 MINUTE PROGRAMS CAN BE RUN, AND ONLY PROGRAMS SUBMITTED FROM SCOPES CAN INTERACT WITH THE SCOPES.

ONLY ALLOWED USERS CAN USE THE SCOPES. IN ORDER TO BECOME AN ALLOWED USER, ONE SHOULD CONTACT A. H. BOND, C. C. EXTENSION 66. THE MAIN USES OF THE SCOPES ARE EXPECTED TO BE FOR: (A) PROGRAMS NEEDING ON-LINE DYNAMICAL GRAPHICAL DISPLAY; AND (B) INTERACTIVE PROGRAMS, THAT IS, PROGRAMS WHICH COMMUNICATE WITH THE HUMAN WHILE RUNNING, AND CAN BE GUIDED AND INFLUENCED BY THE HUMAN. THE VERY GENERAL DISPLAY EQUIPMENT ALLOWS A GREAT VARIETY OF METHODS OF MAN-PROGRAM INTERACTION.

THE SYSTEM IS STILL UNDER DEVELOPMENT AND ATTEMPTS TO USE SOME FEATURES WILL YIELD THE ERROR MESSAGE 'SORRY, NOT YET IMPLEMENTED'. HOWEVER, THE SCOPE USERS MANUAL WILL BE KEPT STRICTLY UP TO DATE WITH CURRENT IMPLEMENTATION. THE DATE OF REWRITING IS SHOWN ON THE FRONT COVER. BETWEEN REWRITINGS, ANY CORRECTIONS TO THE MANUAL ARE KEPT ON AN AND FILE, AND CAN BE OBTAINED BY EXECUTING

```
USER CR38AB14; FILE 81/P; GET TO $; RUN,AND,TAPE;
   ALLOW 5 PAGES AND 2 MINUTES.
```

FURTHER COPIES OF THE FULL MANUAL CAN BE OBTAINED BY EXECUTING

```
USER CR38AB14; FILE 82/P; GET TO $; RUN,AND,TAPE;
   ALLOW 60 PAGES AND 6 MINUTES.
```

OR FROM A H BOND.

2. INTRODUCTION AND SUMMARY

THE SCOPES CAN BE USED OFF-LINE, THAT IS, WITHOUT USING THE CENTRAL PROCESSOR OF THE G-21, IN FACT, ONLY USING ONE 8K MODULE OF MEMORY. OFF-LINE, ONE CAN ENTER CHARACTERS ONTO THE SCOPE FACE FROM THE KEYBOARDS AND ENTER VECTORS (LINES). ONE CAN ALSO ALTER EXISTING DISPLAY BY DELETION AND INSERTION OF ELEMENTS, AND ONE CAN TRANSLATE (MOVE) PARTS OF THE DISPLAY TO OTHER PARTS OF THE SCREEN. THE REST OF THE G-21 CAN OPERATE NORMALLY. THE OFF-LINE CAPABILITIES ARE THE SUBJECT OF J. QUATSE'S MANUAL. THEY CAN ALSO BE USED WITH THE SCOPE MONITOR LOADED. THE SCOPE MONITOR IS AUXILIARY TO THE MAIN G-21 MONITOR AND WORKS ON AN INTERRUPT BASIS. NORMAL USER PROGRAMS CAN BE PROCESSED BY THE G-21 AND WHEN SOME SCOPE COMPUTATION IS NEEDED, THE USER PROGRAM IS INTERRUPTED FOR A FEW MILLISECONDS.

IN THIS WAY THE SCOPE MONITOR CAN SNATCH BRIEF SPELLS OF COMPUTATION TO CARRY OUT MANAGERIAL FUNCTIONS AS DESIRED BY THE USER. THIS IS DONE BY PRESSING THE APPROPRIATE INTERRUPT BUTTONS. THE MEANINGS CURRENTLY ASSOCIATED WITH THE BUTTONS ARE SHOWN BY AN EXPLANATORY DISPLAY. THE FACILITIES PROVIDED BY THE SCOPE MONITOR ARE DESCRIBED IN DETAIL IN SECTION 4. THEY INCLUDE STORAGE OF DISPLAY MATERIAL ON 'SCOPE FILES', SUBMISSION OF PROGRAMS TYPED ON THE SCOPES, THE PERUSAL AND EDITING OF TEXT, AUXILIARY DRAWING OPERATIONS LIKE LIGHT-PEN TRACKING. THERE ARE DEBUGGING FACILITIES WITH A DYNAMIC CORE DISPLAY AND ON-LINE PATCHING AND TRANSFER FACILITIES.

IN ADDITION TO INTERRUPTS PRODUCED BY THE INTERRUPT BUTTONS, THE SCOPE MONITOR RECEIVES INTERRUPTS ONCE EVERY SECOND, TRIGGERED BY THE G-20 REAL-TIME CLOCK. RELYING ONLY ON THESE CLOCK PULSES TO PROCESS REQUESTS WOULD LEAD TO TOO LONG A RESPONSE TIME. THE CLOCK PULSE ENABLES THE SCOPE MONITOR TO PROVIDE CONTINUOUS MODE OPERATIONS SUCH AS THE DYNAMIC CORE DUMP, THE ROTATION MODE AND THE CURVE DRAWING MODE.

INTERACTIVE PROGRAMS CAN BE WRITTEN IN ANY PROGRAMMING LANGUAGE. THEY CAN COMMUNICATE WITH THE SCOPES BY USING THE 'B ROUTINES' PROVIDED BY THE SCOPE MONITOR (THESE ARE LIKE I ROUTINES IN THE MAIN MONITOR). USING THESE, A PROGRAM CAN SET UP A GENERAL GRAPHICAL DISPLAY AND CAN EXAMINE A GRAPHICAL DISPLAY ENTERED BY A HUMAN. THE HUMAN AND PROGRAM ARE TREATED MORE OR LESS EQUIVALENTLY BY THE SCOPES, AND THE SCOPES PROVIDE A GENERAL, RAPID AND TRANSPARENT INTERFACE BETWEEN THEM TO PERMIT MAN-MACHINE COOPERATION ON A PROBLEM.

INTERACTION WITH THE PROGRAM CAN CONSIST OF EACH READING DISPLAY MATERIAL SET UP BY THE OTHER; AND, IN ADDITION, THERE ARE 8 'STATE SWITCHES' AND 2 'ANALOG KNOBS' (GIVING A QUASI-CONTINUOUS VARIABLE), WHICH CAN BE SET BY THE HUMAN AND READ BY THE PROGRAM USING B ROUTINES. ALSO THE USER CAN DEFINE HIS OWN INTERRUPTS AND THE SCOPE MONITOR WILL PASS CONTROL TO THE DEFINED POINTS IN HIS PROGRAM, WHEN HE PRESSES THE APPROPRIATE BUTTON.

THE B ROUTINES ARE DESCRIBED IN SECTION 7. THERE IS A 'B-PROCEDURE' IN ALGOL AND FORML, WHICH CALLS THE B ROUTINES, AND ALSO MANY USEFUL SUBPROGRAMS IN THESE LANGUAGES AND IN SPITE. THESE ARE KEPT ON AND FILES AND ARE DESCRIBED IN SECTION 8. SIMILAR SUBPROGRAMS CAN BE WRITTEN IN ANY LANGUAGE AVAILABLE ON THE G-21.

INTERACTION WITH USER PROGRAMS CAN ONLY OCCUR DURING THE SHORT RUN TIME OF THE PROGRAM, BUT WE ARE TRYING TO MAKE IT EASY FOR ANY USER TO WRITE A 'SUBSYSTEM', WHICH WOULD BE ESSENTIALLY PART OF THE SCOPE MONITOR AND OPERATE ON AN INTERRUPT BASIS. MODULES OF CODE ARE KEPT ON SCOPE FILES AND SWAPPED IN BY THE SCOPE MONITOR AS NEEDED AND AS SPACE ALLOWS. IT IS ONLY POSSIBLE TO WRITE SUCH MODULES IN ASSEMBLY LANGUAGE AND THEIR SIZE IS RESTRICTED TO <3K; HOWEVER, A SUBSYSTEM CAN CONSIST OF AN ARBITRARY NUMBER OF LINKED REENTRANT MODULES. SUBSYSTEMS ARE DISCUSSED IN SECTION 9.

G-21 SYSTEMS AUXILIARY TO THE SCOPE MONITOR ARE DESCRIBED IN SECTION 10; FOR EXAMPLE, A USER SYSTEM IS NEEDED TO MOVE MATERIAL BETWEEN AND FILES AND SCOPE FILES. IN SECTION 11, WE OUTLINE THE INTERNAL WORKING OF THE SCOPE MONITOR PROGRAM.

3. OFF LINE USE, TERMS AND CONCEPTS

THERE ARE 3 SCOPES, NUMBERED 1, 2, 3 FROM THE LEFT OF THE ROOM. THE SCOPE FACE IS 10 INCHES BY 10 INCHES AND HAS 1024 X 1024 RASTER POINTS. THE 32 BUTTONS ALONGSIDE THE FACE ARE THE STATE SWITCHES AND CONSTITUTE THE STATE WORD. THE LOWER 2 ROWS ARE COLORED GREEN AND ARE FOR USE BY A USER PROGRAM. WHEN A SWITCH IS ON IT IS LIT AND THE VALUE OF THE SWITCH IS 1. ON THE LEFT OF THE SCREEN ARE 2 'ANALOG KNOBS' KNOB 1 ABOVE KNOB 2. THESE CAN BE USED BY A USER PROGRAM; THEIR VALUE VARIES FROM 0 TO 63 AND THE FULL RANGE IS OBTAINED IN HALF A TURN.

ON THE DESK, THERE ARE TWO KEYBOARDS, USED EQUIVALENTLY, AND A CONTROL PANEL CONSISTING OF:

(I) 20 INTERRUPT BUTTONS NUMBERED 0-19

(II) A CURSOR CONTROL CONSISTING OF 4 BUTTONS TO INDICATE WHICH DIRECTION TO MOVE THE CURSOR ON THE SCREEN. THE CENTRAL BUTTON IN THE CONFIGURATION MAKES THE CURSOR MOVE FASTER, AND THE SLEW BAR MAKES IT MOVE EVEN FASTER.

(III) THE MARK BAR

TO ENTER DISPLAY MATERIAL ON THE SCREEN ONE FIRST NOTES FROM THE 6TH ROW OF STATE SWITCHES THAT THERE ARE 4 'PAGES' FOR USE. THIS MEANS THAT ONE CAN HAVE 4 DIFFERENT INDEPENDENT DISPLAYS AVAILABLE, WHICH CAN BE MADE VISIBLE BY USING THE APPROPRIATE STATE SWITCH AND SUPERIMPOSED AS DESIRED. HOWEVER, ONE SHOULD ONLY ENTER MATERIAL INTO ONE PAGE AT A TIME.

BEFORE ONE CAN ENTER MATERIAL, ONE MUST USE THE SCOPE MONITOR, DESCRIBED LATER, TO RESERVE SOME MEMORY SPACE FOR THE MATERIAL AND TO DEFINE THAT SPACE TO CORRESPOND TO THE REQUIRED PAGE NUMBER. ALSO ONE MUST ENABLE THE PAGE ONE IS USING AND TURN THE PAGE STATE SWITCH ON. THIS ALLOWS DISPLAY MATERIAL TO BE ENTERED MANUALLY. WHEN A PAGE IS ENABLED, IT HAS A CURSOR VISIBLE AND THIS DEFINES THE PLACE ON THE SCREEN WHERE ATTENTION IS FOCUSED. CHARACTERS MAY NOW BE ENTERED FROM THE KEYBOARD IF ONE SETS THE STATE SWITCHES TO ENTER AND CHARACTER.

VECTORS (LINES) MAY BE ENTERED MANUALLY AND THE DISPLAY CAN BE MANIPULATED WITH CORRECTIONS, DELETIONS, INSERTIONS,

TRANSLATIONS, ETC., AS DESCRIBED IN J.Q.M. MOST OPERATIONS ARE DONE BY SETTING THE STATE SWITCHES TO THE APPROPRIATE VALUES, POSITIONING THE CURSOR AND PRESSING THE MARK BAR. ONE CAN INPUT DOUBLE SIZE CHARACTERS BY SETTING A STATE SWITCH AND ONE CAN GET SUBSCRIPT SIZE CHARACTERS BY PRESSING ¬. ONE GETS BACK TO NORMAL SIZE BY PRESSING . MARGINS ARE ENTERED BY POSITIONING THE CURSOR AND SETTING THE MARGIN STATE SWITCHES AND PRESSING MARK. MARGINS AND ALL OTHER CONTROL SYMBOLS CAN NOT ONLY BE MADE VISIBLE BY A STATE SWITCH, THEY CAN ALSO BE MANIPULATED IN EXACTLY THE SAME WAY AS NORMAL SYMBOLS.

TO CLEAR A PAGE OF ALL MATERIAL, IT MUST BE VISIBLE AND ENABLED, AND ONE THEN SETS THE CLEAR STATE SWITCH. THE SPACE RESERVED FOR THE PAGE IS STILL THERE AND IT IS STILL ENABLED AFTER THE CLEAR OPERATION.

ONE SHOULD ONLY HAVE ONE PAGE ENABLED AT ANY ONE TIME, AS THERE IS ONLY ONE CURSOR POSITION.

MARGINS CONTROL ONLY TEXT ENTERED AFTER THEM AND ONE CAN HAVE SEVERAL MARGINS ON ONE PAGE. IN THE ABSENCE OF MARGINS THE END OF THE SCREEN IS AN EFFECTIVE MARGIN WHEN ONE DOES A RETURN CHARACTER. IF DISPLAY MOVES OFF THE SCREEN IN ANY DIRECTION IT 'WRAPS ROUND' AND APPEARS ON THE OTHER EDGE OF THE SCREEN; SIMILARLY FOR THE CURSOR POSITION.

THE LIGHT PEN CAN BE USED TO POSITION THE CURSOR AT AN EXISTING DISPLAY ELEMENT. ONE POINTS THE LIGHT PEN AT THE ELEMENT AND THE CURSOR WILL AUTOMATICALLY MOVE THERE. IT MAY BE NECESSARY TO INCREASE THE BRIGHTNESS TO GET IT TO WORK.

THE SCREEN FACE CAN BE PHOTOGRAPHED IN COLOR OR BLACK AND WHITE WITH AN ORDINARY CAMERA. THE ENGINEERING GROUP HAS A POLAROID CAMERA FOR THIS PURPOSE. ALSO THERE IS A SPECIAL HARD COPY DEVICE,UNDER DEVELOPMENT BY THE ENGINEERING GROUP,WHICH TAKES AN ACTUAL SIZE NEGATIVE IMAGE OF THE SCREEN DIRECTLY ONTO PHOTOSENSITIVE PAPER, SO THAT THE LINES AND CHARACTERS ARE BLACK ON WHITE. ENQUIRIES ABOUT THIS EQUIPMENT SHOULD BE DIRECTED TO BEAU BRINKER,C.C. EXTENSION 75. OPINIONS AND IDEAS ON THE HARDWARE SHOULD BE SENT TO THE ENGINEERING GROUP. THERE IS NOW AVAILABLE A RAND TABLET,WHICH CAN BE ATTACHED TO EITHER SCOPE 1 OR SCOPE 2. IT CAN BE USED AS A POINTING DEVICE LIKE THE LIGHT PEN, BUT IN ADDITION IT ACTS LIKE THE MARK BAR. FURTHER, IT WILL ENTER LINES CONTINUOUSLY INTO THE PAGE GIVING CURSOR TRACKING AND CURVE DRAWING. INSTRUCTIONS ON ITS USE ARE TO BE FOUND IN A FOLDER WITH THE EQUIPMENT. ENQUIRIES ABOUT IT SHOULD BE SENT TO DICK SHOUP.

4. THE SCOPE MONITOR

A. GENERAL LAYOUT AND OPTION STATE

THE SCOPE MONITOR PROVIDES A RANGE OF FACILITIES WHICH ARE LINKED TO THE INTERRUPT BUTTONS. THE MEANING OF THE BUTTONS IS REDEFINED AS ONE USES VARIOUS 'STATES' OF THE SCOPE MONITOR. WHEN ONE FIRST APPROACHES A SCOPE, AFTER THE SCOPE MONITOR HAS BEEN LOADED, IT HAS 'THIS SPACE RESERVED FOR SYSTEM MESSAGES' ON THE BOTTOM OF THE SCREEN. IN THIS STATE, EVERY INTERRUPT BUTTON LEADS TO THE LOG-IN STATE (FIG. 1) AND THE USER MUST ENTER HIS FULL G-21 USAGE NUMBER AT THE POSITION OF THE CURSOR. THE CURSOR IS SET BY THE SCOPE MONITOR AND THE STATE SWITCHES ARE SET TO ENTER, CHARACTER, PAGE 1 (IF THIS DOESN'T HAPPEN, SET THEM BY HAND). AFTER TYPING THE USAGE NUMBER, PRESS RETURN. THE RETURN CHARACTER IS USED BY THE SCOPE MONITOR AS A COMPARE INTERRUPT, AND TELLS IT TO READ IN THE CHARACTER JUST TYPED BY THE USER. IF THE NUMBER WAS MISTYPED, OR DOES NOT BELONG TO AN ALLOWED USER, THE MESSAGE 'SORRY NOT ACCEPTABLE' WILL APPEAR. OTHERWISE, IT WILL GO TO OPTION STATE AND DISPLAY THE MEANINGS OF THE INTERRUPT BUTTONS IN THIS STATE. THE OPTION STATE IS THE TOP-LEVEL OF A HIERARCHY OF STATES AND WITH IT ONE SELECTS ANOTHER STATE.

NOTE THE WORD 'STATE' IS USED TO DESCRIBE THE CONDITION OF THE SCOPE MONITOR AND THE DEFINITION OF INTERRUPTS IN THAT CONDITION. EACH HAS AN ASSOCIATED SYSTEM 'PAGE' AND SO SOMETIMES THE WORD 'PAGE' REFERS TO A 'STATE'. OCCASIONALLY, THE CONDITION OF THE SCOPE MONITOR IS DESCRIBED AS A 'MODE', ESPECIALLY IF IT IS DOING AN OPERATION CONTINUOUSLY. THE USE OF THESE WORDS SHOULD BE DISTINGUISHED FROM THEIR USE IN OFF-LINE USE. THERE IS SOFTWARE STATE, PAGE AND MODE DISTINCT FROM HARDWARE STATE, PAGE AND MODE. IT IS HOPED THAT NO CONFUSION WILL ARISE. THUS THE MEANINGS OF THE BUTTONS IN THE OPTION STATE ARE ALL 'CHANGE STATE TO -----STATE'. THE VARIOUS STATES ARE DESCRIBED BELOW. IN EVERY STATE, INTERRUPT 0 ALWAYS MEANS GO BACK TO OPTION STATE. INTERRUPTS 17, 18, AND 19 ARE CURRENTLY USED FOR SYSTEM MAINTENANCE AND SHOULD NOT BE USED.

THE DISPLAYS USED BY THE SCOPE MONITOR CANNOT BE ALTERED BY THE USER AS THEY ARE IN ALTERNATE MODE. EVEN THOUGH PAGE 1 IS USED BY THE SCOPE MONITOR, IT CAN ALSO BE USED BY THE USER AS A NORMAL PAGE.

WHEN TYPING IN MORE THAN ONE VALUE TO THE SCOPE MONITOR, DO A RETURN AFTER EACH VALUE AND THE SCOPE MONITOR WILL REPOSITION THE CURSOR.

LOG - IN
ENTER YOUR USAGE NUMBER HERE

THIS SPACE RESERVED FOR SYSTEM MESSAGES

FIGURE 1

IN EACH STATE, THE MEANING OF THE INTERRUPTS ARE DISPLAYED BY A SYSTEM PAGE, THIS DOES NOT INTERFERE WITH THE USER DISPLAY AND CAN BE TURNED ON OR OFF (MADE VISIBLE OR INVISIBLE) IN ANY STATE BY USING INTERRUPT 1. ON PRESSING AN INTERRUPT BUTTON, ITS NUMBER IS DISPLAYED IN THE BOTTOM RIGHT HAND CORNER OF THE SCREEN, DURING THE PROCESSING OF AN INTERRUPT THE NUMBER IS MADE TO FLASH. THE USER SHOULD NOT PRESS ANOTHER INTERRUPT BUTTON UNTIL THE NUMBER HAS STOPPED FLASHING. USUALLY THE OPERATION IS VERY QUICK AND THE USER DOESN'T SEE ANY FLASHING; HOWEVER, OPERATIONS REQUIRING THE SCOPE FILES INVOLVE THE USE OF THE DISC AND ONE MAY HAVE TO WAIT FOR THE DISC TO BECOME AVAILABLE FOR A SECOND OR TWO. THE NUMBER WILL ALSO FLASH WHILE TYPING IN VALUES OF PARAMETERS TO THE SCOPE MONITOR, IN THIS CASE, ONE CAN CONTINUE TO ENTER PARAMETERS.

LOG OUT

PRESSING INTERRUPT 8, ON THE OPTION PAGE, LOGS THE CURRENT USER OUT AND THE MESSAGE 'LOGGED OUT' IS DISPLAYED.

B. MANAGEMENT STATE

THE MEANING OF THE INTERRUPTS IN THIS STATE ARE SHOWN BY THE SYSTEM DISPLAY, REPRODUCED IN FIGURE 2.

AN ALLOWED USER HAS RESERVED FOR HIM 20 SCOPE FILES NUMBERED 1 TO 20 WHICH ARE ARBITRARY IN SIZE. HE CAN SAVE DISPLAY MATERIAL ON THESE FILES PERMANENTLY BY USING INTERRUPT 2. HE CAN MOVE THE CONTENTS OF A PREVIOUSLY STORED FILE TO A PAGE DISPLAYED BY USING INTERRUPT 3. WHEN USING 3, SPACE DOES NOT HAVE TO BE RESERVED FOR THE PAGE. IT IS DONE AUTOMATICALLY, INDEED ANY MATERIAL ON THAT PAGE BEFORE IS CLEARED. ONE CAN GET A DIRECTORY OF THE SCOPE FILES BY PRESSING INTERRUPT 5. THE DISPLAY IS LIKE FIGURE 3. IT SHOWS THE BASE (RECORD NUMBER) AND LENGTH OF THE RECORD ON THE DISC. THIS IS NOT OF MUCH USE TO THE NORMAL USER EXCEPT TO SEE THAT A FILE IS PRESENT OR HAS CHANGED IN LENGTH.

INTERRUPTS 4, 6 - 9 HANDLE THE RESERVED SPACE FOR THE PAGES. INTERRUPT 6, RESERVES SOME SPACE FOR A GIVEN PAGE. THE UNIT USED IS THE BLOCK, WHICH IS 160_8 WORDS. THERE ARE 30 BLOCKS AVAILABLE FOR USE BY 3 SCOPES. A PAGE PACKED SOLID WITH DISPLAY PROBABLY NEEDS 4 BLOCKS OF SPACE.

INTERRUPT 7 ENABLES A PAGE, AND 8 DISENABLES A PAGE.

INTERRUPT 9 DELETES A PAGE; I.E., IT REMOVES THE SPACE RESERVED FOR THAT PAGE AND MAKES IT AVAILABLE FOR OTHER USE. USING 8 MERELY DISENABLES A PAGE AND KEEPS THE SPACE RESERVED.

MANAGEMENT PAGE

PRESS INTERRUPT NUMBER

2. SAVE PAGE AS SCOPE FILE

3. READ IN SCOPE FILE AS PAGE

4. APPEND PAGE TO PAGE

5. DISPLAY DIRECTORY OF SCOPE FILES

6. GET BLOCKS FOR PAGE

7. ENABLE PAGE

8. DISENABLE PAGE

9. DELETE PAGE

FIGURE 2

DIRECTORY FOR LC02

FILE	BASE	LENGTH
00.	000	000
01.	576	002
02.	535	002
03.	570	004
04.	533	002
05.	530	003
06.	525	003
07.	000	000
08.	000	000
09.	000	000
10.	000	000
11.	000	000
12.	000	000
13.	000	000
14.	000	000
15.	000	000
16.	000	000
17.	000	000
18.	000	000
19.	000	000
20.	000	000

THIS SPACE RESERVED FOR SYSTEM MESSAGES

FIGURE 3

INTERRUPT 4 WILL APPEND ONE PAGE TO ANOTHER SO THAT THE SECOND PAGE THEN HAS THE DISPLAY MATERIAL OF BOTH, AND THE FIRST IS UNCHANGED .

BELOW IS GIVEN THE SEQUENCE OF ACTIONS REQUIRED TO LOG IN AND SET UP THE SCOPE FOR ENTERING CHARACTERS AND LINES ON THE SCREEN.

1. IF NO ONE IS LOGGED IN YET, THERE WILL JUST BE THE ONE LINE MESSAGE ON THE BOTTOM OF THE SCREEN, OR ELSE THE MESSAGE 'LOGGED OUT'. IN THIS CASE, PRESS INTERRUPT 0. THIS GIVES THE LOG-IN PAGE. TYPE IN YOUR USER NUMBER AND PRESS RETURN. THIS WILL GIVE THE OPTION PAGE.

2. IF SOMEONE IS LOGGED IN ALREADY, PRESS INTERRUPT 0 - THIS GIVES THE OPTION PAGE.

3. IN THE OPTION STATE, PRESS INTERRUPT 2. THIS GIVES THE MANAGEMENT PAGE.

4. IN THE MANAGEMENT STATE, PRESS INTERRUPT 6. THIS PUTS THE CURSOR AFTER 'GET' AND DISPLAYS THE NUMBER 6 BLINKING IN THE BOTTOM RIGHT HAND CORNER. TYPE THE FIGURE 2 FROM THE KEYBOARD AND PRESS RETURN. THIS RESETS THE CURSOR TO AFTER 'PAGE'. TYPE 2 AND RETURN. YOU NOW HAVE RESERVED 2 BLOCKS OF SPACE ON YOUR PAGE 2.

5. PRESS INTERRUPT 7. THE CURSOR WILL APPEAR AFTER 'PAGE' ON LINE 7 OF THE MANAGEMENT PAGE. TYPE 2 AND RETURN. PAGE 2 IS NOW ENABLED, AND WILL ALLOW DISPLAY MATERIAL TO BE ENTERED FROM THE CONSOLE.

6. PRESS INTERRUPT 1. THIS MAKES THE MANAGEMENT PAGE DISPLAY INVISIBLE.

7. PRESS THE STATE SWITCH FOR PAGE 2. YOU SHOULD SEE A CURSOR. USE THE CURSOR CONTROL TO POSITION THE CURSOR. TO TYPE IN CHARACTERS, PRESS STATE SWITCHES ENTER AND CHARACTER AND THEN TYPE FROM THE KEYBOARD. TO DRAW LINES, PRESS STATE SWITCHES ENTER AND VECTOR AND USE THE CURSOR CONTROL AND THE MARK BAR.

C. THE PROGRAM STATE

SEE FIGURE 4. THIS STATE ORGANIZES THE INITIATION OF USER PROGRAMS AND USER SYSTEMS FROM THE SCOPE MONITOR. WHEN A PROGRAM IS ACTUALLY INTERACTING WITH THE SCOPES, THE SCOPE MONITOR SHOULD BE PUT IN USER PROGRAM INTERACTION STATE OBTAINABLE FROM THE OPTION STATE, HOWEVER, ALL ORGANIZATION PRIOR TO AND AFTER THE RUN IS DONE WITH THE PROGRAM STATE.

TO SUBMIT A PROGRAM,ONE SHOULD GET SOME BLOCKS FOR A PAGE AND ENABLE IT, THEN TYPE THE PROGRAM ONTO THAT PAGE. NOTE THAT THERE ARE NO TAB SETTINGS ON THE SCOPES; EVERYTHING MUST BE SPACED BY HAND. ONE CAN KEEP PROGRAMS ON SCOPE FILES ALSO AND PUT THEM ON THE PAGE THAT WAY. ONE WOULD USUALLY SET UP THE PROGRAM WITH THE PROGRAM PAGE SYSTEM DISPLAY TURNED OFF. THEN ONE SHOULD TURN OFF THE PAGE AND TURN ON THE SYSTEM DISPLAY AGAIN USING INTERRUPT 1. THE SUBMISSION OF A PROGRAM TAKES PLACE IN TWO STAGES. FIRST IT MUST BE MOVED TO THE 'INPUT FILE'. THIS IS NOT TO BE CONFUSED WITH A SCOPE FILE. IT IS A PSEUDO TELETYPE BUFFER. SECOND, THE INPUT FILE MUST BE 'SUBMITTED' TO RUN ON THE G-21. TO MOVE IT TO THE INPUT FILE ONE SHOULD USE INTERRUPT 2. THIS CONVERTS THE PROGRAM TO (UPPER CASE) G-21 CHARACTERS AND PUTS IN A BLANK JOB CARD AT THE TOP. INTERRUPT 3 MOVES A PAGE WITHOUT CONVERSION AND IS RARELY USED.

USING INTERRUPT 4, ONE CAN NOW SUBMIT THE INPUT FILE. THE VALUES OF TIME, PAGES AND SYSTEM REQUESTED ARE TYPED IN AND PUT INTO THE JOB CARD; AND THE JOB IS PLACED IN THE G-21 QUEUE TO BE RUN.

WHEN IT RUNS, ANY TELETYPE OUTPUT IS PUT IN THE 'OUTPUT FILE'. ONE CAN LOOK AT THE INPUT FILE OR THE OUTPUT FILE BY USING INTERRUPTS 5 AND 6. THESE MOVE THEM TO A DESIGNATED PAGE; SPACE DOES NOT HAVE TO BE RESERVED FOR THE PAGE IN THIS OPERATION.

INTERRUPTS 7 AND 8 ARE NOT YET IMPLEMENTED BUT WILL PERMIT A PERUSAL OF THE INPUT OR OUTPUT FILE. THESE FILES ARE VERY MUCH LARGER THAN CAN BE FITTED ONTO A PAGE, AND INTERRUPTS 5 AND 6 JUST LOOK AT THE FIRST FEW BLOCKS. AT THE MOMENT, ONE CAN ONLY LOOK AT THE REST OF ONE'S OUTPUT BY GETTING THE LINE PRINTER OUTPUT. THE SCOPES 1, 2, AND 3 ARE EQUIVALENT TO TELETYPES NUMBER 5, 6, AND 7 RESPECTIVELY, AND LINE PRINTER OUTPUT IS NUMBERED WITH THESE REMOTE NUMBERS. ALSO THE JOB CARD HAS THE WORDS SCOPES AND COURIER. WHEN THE COURIER SERVICE IS IN OPERATION; OUTPUT IS PLACED ON THE TABLE IN PORTER HALL BASEMENT NEAR THE SCOPES ROOM. OTHERWISE, ASK FOR IT AT THE I/O COUNTER.

WHILE A PROGRAM IS INTERACTING WITH THE SCOPES, THE SCOPE MONITOR CAN STILL BE USED IN ANY STATE. THE INTERRUPTS DEFINED BY THE USER WILL ONLY BE PASSED TO THE USER PROGRAM WHEN THE SCOPE MONITOR IS IN THE USER PROGRAM INTERACTION STATE.

PROGRAM PAGE

PRESS	INTERRUPT NUMBER
2.	CONVERT PAGE AND MOVE TO INPUT FILE
3.	MOVE PAGE (UNCONVERTED) TO INPUT FILE
4.	SUBMIT INPUT FILE : TIME PAGES SYSTEM
5.	DISPLAY INPUT FILE AS PAGE
6.	DISPLAY OUTPUT FILE AS PAGE
7.	FORWARD TEN LINES
8.	BACK TEN LINES
9.	LOAD MONITOR MODULE OF USER
10.	TRANSFER TO ENTRY POINT OF MODULE OF USER
11.	RELEASE MODULE OF USER
12.	ALLOW PROGRAM FROM SCOPE TO INTERACT

FIGURE 4.

THE INPUT FILE IS MOVED TO ANOTHER INACCESSIBLE INPUT FILE ON SUBMISSION, AND THIS LATTER INPUT FILE CANNOT BE LOOKED AT OR ALTERED. HENCE, IF YOU HAVE MADE A MISTAKE IN YOUR PROGRAM AND HAVE ALREADY SUBMITTED IT, YOU CANNOT RECALL IT; IT WILL BE RUN. IF YOU RESUBMIT, PROBABLY BOTH WILL RUN.

SCOPE PROGRAMS ONLY HAVE THE SAME PRIORITY AS NORMAL TELETYPE PROGRAMS, AND THEY CAN ONLY RUN FOR 3 MINUTES; HOWEVER,THE WAITING IN HANDLED DIFFERENTLY, TO MAKE IT EASIER FOR THE USER TO BE PRESENT WHILE HIS PROGRAM IS RUNNING. ON SUBMISSION OF THE PROGRAM IT GOES TO THE TOP OF THE QUEUE (SM PRIORITY) AND WILL PROBABLY RUN WITHIN 10 MINUTES OF SUBMISSION. THE SCOPE MONITOR COMPUTES, AT THIS TIME, THE ALLOWED TIME OF NEXT SUBMISSION. ALLOWED TIME = [REAL TIME - (TIME OF SUBMISSION OF CURRENTLY RUNNING PROGRAM)] + REAL TIME.

A SUBSEQUENT ATTEMPT TO SUBMIT A PROGRAM WILL YIELD THE ERROR MESSAGE 'SORRY NOT ACCEPTABLE', IF THE TIME THEN IS BEFORE THE ALLOWED TIME. WHEN A PROGRAM IS QUEUED THERE IS NO INDICATION THAT IT IS QUEUED. WHEN IT FINISHES, THE SCOPE MONITOR DISPLAYS THE MESSAGE 'OUTPUT READY' AND THE USER CAN FIND TELETYPE OUTPUT IN THE OUTPUT FILE.

INTERRUPTS 9 - 12 ARE NOT YET DEBUGGED AND ARE FOR WRITING 'USER SCOPE MONITOR SUBSYSTEMS' OR 'USER MODULES'. MODULES ARE DISCUSSED IN SECTION 8.

D. THE DEBUG STATE

SEE FIGURE 5. THIS DISPLAYS A DYNAMIC CORE DUMP OF ANY REGION OF CORE OF THE G-21. THE REGION DISPLAY IS SELECTED BY TURNING THE ANALOG KNOBS AND SETTING THE STATE SWITCHES. KNOB 1 IS THE LAST TWO OCTAL DIGITS; KNOB 2 THE MIDDLE TWO; AND THE BOTTOM ROW OF STATE SWITCHES IS THE TOP 4 BITS OF THE ADDRESS. WHILE THE DUMP IS BEING DISPLAYED, IT IS TYING UP THE G-21, AND THE USER PROGRAM IN LOWER CORE IS NOT BEING PROCESSED; HOWEVER, INTERRUPTS CAN BE PROCESSED. THUS THIS FACILITY SHOULD BE USED SENSIBLY AND CERTAINLY NOT LEFT DISPLAYING FOR A LONG TIME.

THE INTERRUPTS ALLOW ONE TO PATCH THE CORE. THIS IS DONE BY PUTTING A NUMBER INTO THE INPUT BOX. THE DEBUG STATE IS ENTERED IN CORRECT MODE, AND THE CURSOR IS ENABLED. ONE MOVES THE CURSOR TO THE INPUT BOX AND CORRECTS THE CONTENTS OF IT; THEN ONE SHOULD GET OUT OF CORRECT MODE.

INTERRUPT 1 CLEARS THE CONTENTS OF THE INPUT BOX TO ZERO.

INTERRUPT 2 STORES THE CONTENTS OF THE INPUT BOX IN THE LOCATION OF THE OCTAL DUMP WHICH IS UNDERLINED.

INTERRUPT 3 PUTS THE CONTENTS OF THE UNDERLINED LOCATION INTO THE INPUT BOX.

INTERRUPT 4 SWAPS THE CONTENTS OF THE INPUT BOX WITH THOSE OF THE UNDERLINED LOCATION.

INTERRUPT 5 ALLOWS ONE TO TRANSFER TO ANY LOCATION; ONE PLACED THE LOCATION IN THE INPUT BOX AND THEN PRESSES INTERRUPT 5. THIS DOES A TRM WITH CONTROL OFF; HOWEVER, NOTE THAT CE AND PE ARE SET FOR THE SCOPE MONITOR, SO THAT

(1) THE USER HAD BETTER RESET THEM TO HIS OWN VALUES.

(11) HE MUST KEEP CONTROL OFF.

ALSO NOTE THAT

(111) HE MUST RETURN THROUGH HIS MARK.

IF (I) (II) OR (III) ARE VIOLATED, YOU WILL PROBABLY DESTROY THE ENTIRE WORLD.

THE USER CAN LOOK AT ANY REGION OF CORE; HOWEVER, HE CANNOT ALTER OR TRANSFER TO AN ADDRESS IF IT IS NOT IN USER CORE, I.E., IN /70 TO /73000. IF HE TRIES TO DO SO, THERE WILL BE NO RESPONSE FROM THE SCOPE MONITOR.

DEBUG PAGE

0. OPTION PAGE
1. CLEAR INPUT
2. STORE INPUT
3. LOAD INPUT FROM MEMORY
4. SWAP INPUT
5. TRM

00000000000

005344	00000000467	00000073626	00000001453	04050005632
005350	01550000100	01730005632	00050000100	05550006732
005354	01730006732	01770007546	00000000000	00050000004
005360	00170005353	00050000002	01770076666	01770005300
005364	00000000000	01770003106	01770003106	00170004312

THIS SPACE RESERVED FOR SYSTEM MESSAGES

FIGURE 5

TEXT HANDLING MODE

PRESS INTERRUPT NUMBER

2. SELECT PAGE
3. SELECT FILE
4. FORWARD TEN LINES
5. BACKWARD TEN LINES
6. GET TO $
7. DUMP
8. NAME CURSOR POINT TO BE
   STRUCTURE POINT
9. UNNAME STRUCTURE POINT
10. GET TO POINT
11. DISPLAY DIRECTORY OF STRUCTURE POINTS
12. READ BLOCKS AT BLOCK FILE TO BLOCK PAGE
13. WRITE BLOCKS AT BLOCK FILE TO BLOCK PAGE

FIGURE 6

E. TEXT HANDLING STATE

SEE FIGURE 6. THIS STATE IS NOT YET DEBUGGED. IT DOES THE MOVEMENT AND SCROLLING ('ROLL ROUND') OF TEXT. IT IS DISTINCT FROM THE TEXT EDITING SYSTEM WHICH IS BEING DEVELOPED BY MIKE COLEMAN AND IS CONCERNED WITH TEXT MANIPULATION ON THE PAGE TO AUGMENT THE FACILITIES PROVIDED BY THE HARDWARE.

TO PERUSE SOME TEXT, IT MUST BE ON A SCOPE FILE. IT CAN BE MOVED ONTO A SCOPE FILE FROM AN AND FILE BY USING AN AUXILIARY SYSTEM (Q.V.). AUXILIARY SYSTEMS WILL MOVE TEXT FROM AN AND FILE IN G-20 CHARACTERS AND CONVERT AND MOVE TO A SCOPE FILE AND WILL MOVE IT BACK AND CONVERT IT BACK. OR WE CAN MOVE IT IN SCOPE CHARACTERS UNCONVERTED BETWEEN AND FILE AND SCOPE FILE AND ALWAYS KEEP IT IN SCOPE CHARACTERS, UNTIL IT IS NECESSARY TO PRINT IT OUT. IT IS SUGGESTED THAT DOCUMENTATION USE THE LEAD SYSTEM (SEE SEPARATE WRITE-UP), IN WHICH ONE INSERTS TYPESETTING COMMANDS INTO THE TEXT, SO IT IS PRINTED OUT IN A PRESCRIBED FORMAT. THE LEAD COMMANDS COULD BE KEPT IN ALL THE TIME AS PART OF THE TEXT. IT IS HOPED EVENTUALLY TO BE ABLE TO OUTPUT ON THE LINE PRINTER OF THE 360 WHICH HAS UPPER AND LOWER CASE CHARACTERS. THE G-20, OF COURSE, HAS ONLY 64 CHARACTERS, INCLUDING ONLY UPPER CASE LETTERS. HAVING GOT THE DOCUMENT INTO A SCOPE FILE, ONE SELECTS THAT FILE USING INTERRUPT 3 AND SELECTS A PAGE TO WORK ON USING INTERRUPT 2. THIS WILL AUTOMATICALLY GET 5 BLOCKS (AS MUCH AS CAN REASONABLY BE SEEN ON ONE PAGE) FOR THAT PAGE AND ENABLE IT. THERE IS A SPECIALLY RESERVED FILE USED FOR A SCRATCH AREA AND ONE CAN NOW ROLL THROUGH THE TEXT USING INTERRUPTS 4 AND 5. THIS SUCCESSIVELY BRINGS IN TEXT FROM THE SELECTED FILE ONTO THE BOTTOM OF THE SELECTED PAGE AND MOVES THE TOP OF THE PAGE INTO THE SCRATCH AREA. ONE CAN USE THE HARDWARE FEATURES TO ALTER THE TEXT, AND ALSO THE SOFTWARE TEXT EDITING FEATURES PROVIDED BY THE TEXT EDITING MODE. FINALLY, TO PUT THE EDITED TEXT ONTO A FILE (WHICH CAN BE THE SAME ONE) ONE EXECUTES GET TO $, WHICH PUTS EVERYTHING IN THE SCRATCH AREA, SELECTS A FILE, AND EXECUTES DUMP. ONE MAY NOT BE ABLE TO BACK UP THE TEXT ONTO THE SAME FILE AS IT MAY HAVE ALTERED IN LENGTH; HENCE THE DUMP PROCEDURE SHOULD ALWAYS BE FOLLOWED. IN ORDER TO WORK MORE EASILY, ESPECIALLY WITH LONG FILES, INTERRUPTS 8 TO 11 PROVIDE THE FACILITY OF IMPOSING STRUCTURE ON OTHERWISE AMORPHOUS TEXT. THE TEXT IS TREATED AS A VERY LONG STRING OF CHARACTERS AND CONTROL CHARACTERS. THE USER CAN NAME ANY POINT IN THE TEXT, BY A 6 CHARACTER NAME OF HIS OWN CHOICE, BY GETTING THE TEXT ONTO THE SCREEN, PLACING THE CURSOR AT THE POINT AND USING INTERRUPT 8. ONE CAN MOVE THE POINT REFERENCED BY A GIVEN NAME BY SIMPLY USING 8 AGAIN. ONE CAN REMOVE THE NAME ALTOGETHER BY USING 9, AND ONE CAN DISPLAY A DIRECTORY OF NAMED POINTS CURRENTLY USED BY PRESSING INTERRUPT 11. ONE CAN THEN GO IMMEDIATELY TO ANY NAMED POINT AND WORK FROM THERE WITH 4 AND 5. AS THE TEXT MOVES BACKWARD AND FORWARD, THE SCOPE MONITOR KEEPS TRACK OF THE LOCATIONS OF THE

NAMED POINTS; IT ACTUALLY PUTS A SCOPE NO-OP COMMAND (NO OPERATION COMMAND) AT THE NAMED POINT. THE USE OF LINE NUMBERS IS CUMBERSOME TO PROGRAM, WASTEFUL OF STORAGE SPACE, BUT, MORE IMPORTANT, VERY MISLEADING IF BACKWARD AND FORWARD MOTION AND ARBITRARY INSERTION AND DELETION ARE ALLOWED. HOWEVER, SOME STRUCTURE IS NEEDED, AND THIS HAS BEEN MADE AS FREE AS POSSIBLE.

F. USER MANUAL

IT IS HOPED THAT THIS USER MANUAL WILL BE DISPLAYABLE FROM THE SCOPE MONITOR; HOWEVER, THIS IS NOT YET IMPLEMENTED.

G. DRAWING STATE

SEE FIGURE 7. THIS STATE IS INTENDED TO PROVIDE EXTRA FACILITIES FOR CONSTRUCTING DISPLAY MATERIAL. NONE OF IT IS DEBUGGED.

INTERRUPT 2 SELECTS A PAGE FOR ATTENTION.

INTERRUPT 3 PUTS ONE IN A ROTATION MODE. IN THIS MODE, AS ONE TURNS ANALOG KNOB 1, THE VECTORS ON THE CURRENTLY SELECTED PAGE ARE ROTATED ABOUT THE POSITION OF THE CURSOR.

INTERRUPT 4 PUTS ONE IN TRACKING MODE. THIS PUTS A TRACKING FIGURE ON THE SELECTED PAGE. ONE CAN THEN USE THE LIGHT PEN TO MOVE THE CURSOR AROUND.

INTERRUPT 5 PUTS ONE IN CURVE DRAWING MODE. IN THIS CASE, AS ONE MOVES THE CURSOR WITH THE LIGHT PEN, A CURVE IS DRAWN PERMANENTLY INTO THE PAGE.

DRAWING MODE

PRESS INTERRUPT NUMBER

2 SELECT PAGE

3 ROTATIONAL MODE

4 TRACKING MODE

5 CURVE DRAWING MODE

FIGURE 7

H. USER PROGRAM INTERACTION STATE

IN THIS STATE, THE MEANING OF THE INTERRUPTS ARE AS DEFINED BY THE USER PROGRAM. THE USER PROGRAM DEFINES THEM BY CALLING R25, AND GIVING THE INTERRUPT ENTRY POINT IN THE PROGRAM. THIS IS EXPLAINED IN SECTION 6. ONE CAN ONLY GET INTO USER MODE WHILE THE PROGRAM IS ACTUALLY RUNNING.

I. TEXT EDITING STATE

THIS IS A SUBSYSTEM BEING DEVELOPED BY MIKE COLEMAN.

J. ERROR MESSAGES

ERROR MESSAGES FROM THE SCOPE MONITOR ARE FEW AND UNHELPFUL. IT IS USUALLY POSSIBLE TO RECOVER AND JUST CARRY ON FROM THE OPTION STATE AFTER AN ERROR.

1. SORRY ROUTINE NOT YET IMPLEMENTED.

2. SORRY NOT ACCEPTABLE. INDICATES AN ARGUMENT IS NOT ACCEPTABLE, USUALLY OUT OF BOUNDS. ATTEMPTS TO USE A PAGE WITH NUMBER NOT IN [1, 4], ATTEMPTS TO READ IN A SCOPE FILE WITH NOTHING ON IT, ATTEMPTS TO ALTER CORE LOCATIONS NOT IN USER CORE WILL EVOKE THIS MESSAGE. THE STACK IS CLEARED.

3. UNSPECIFIED INTERRUPT. IF ONE PRESSES BUTTONS NOT DEFINED BY THE SYSTEM DISPLAY.

4. MULTIPLE INTERRUPT ERROR WILL OCCUR IF MORE THAN ONE INTERRUPT IS REQUESTED; FOR EXAMPLE, IF ONE IS REQUESTED BEFORE A PREVIOUS ONE HAS BEEN PROCESSED. ALL INTERRUPT REQUESTS ARE REMOVED, AND YOU MUST REREQUEST.

5. PANIC. THIS INDICATES THAT YOU HAVE RUN OUT OF SPACE, EITHER CORE SPACE, DISC OR STACK SPACE. IT INITIALIZES THE STACK AND REMOVES CONTINUOUS MODE OPERATIONS. YOU SHOULD BE ABLE TO RECOVER. IF IT IS CORE SPACE, DELETING UNWANTED CORE BLOCKS WILL HELP.

6. ADDROP <ADDRESS>. THIS SHOULDN'T EVER HAPPEN. IF IT DOES, WRITE DOWN THE VALUE OF THE ADDRESS AND SEND IT TO A. H. BOND. YOU MAY WELL BE ABLE TO RECOVER FROM THIS ERROR CONDITION.

7. USER ERROR. THIS INDICATES AN ERROR HAS OCCURRED IN THE CALLING OF A B ROUTINE BY THE USER PROGRAM. YOU CAN REMOVE THE ERROR MESSAGE DISPLAY BY GOING BACK TO OPTION STATE MOMENTARILY. AN ERROR CONDITION IS INDICATED TO THE PROGRAM AND AN ERROR NUMBER IS PASSED TO IT. A LIST IS GIVEN AT THE END OF CHAPTER 7.

8. SOMETIMES, AS A RESULT OF A SERIES OF PARTIALLY RECOVERABLE ERRORS, THE SCOPE MONITOR GRADUALLY DEGENERATES AND EXHIBITS ANOMALOUS BEHAVIOR, LIKE SETTING RANDOM PATTERNS ON THE STATE LIGHTS, ETC. IN THIS CASE, IT IS TIME TO RELOAD. ALSO, IF

YOU DO NOT RECOVER CORRECTLY FROM ANY OF THE ERROR CONDITIONS, YOU CAN RELOAD. YOU RELOAD BY PHONING THE MACHINE ROOM (EXT. 60) AND ASKING FOR A RELOAD OF THE SCOPE MONITOR AT THE NEXT CONVENIENT TIME. HARDWARE ERRORS OR FAULTS SHOULD BE REPORTED TO THE RESIDENT PHILCO CUSTOM ENGINEERS, C. C. EXT. 59, WHO ARE IN CHARGE OF HARDWARE MAINTENANCE.

5. PROGRAMMING FOR GRAPHICS.

A TYPICAL OUTPUT DEVICE,LIKE A DISC OR PRINTER,WITH AN AUTONOMOUS CONTROL UNIT. WORKS AS FOLLOWS: THE OUTPUT MATERIAL IS PLACED IN A BUFFER WHICH IS PART OF THE ADDRESSABLE CORE, IT MAY HAVE TO BE A SPECIAL AREA OR CAN BE ANY LOCATION. THIS MATERIAL IS IN BIT PATTERNS CORRESPONDING TO OPERATIONS PERFORMED BY THE OUTPUT DEVICE. THE OBVIOUS CASE IS THE CHARACTER. THERE MAYBE OTHERS WHICH CONTROL THE OPERATION OF THE DEVICE LIKE NEW LINE ETC. THE TRANSFER IS THEN INITIATED BY THE CP WHICH CARRIES ON WITH OTHER TASKS WHILE THE I/O CONTROL UNIT PERFORMS THE I/O TRANSFER. THE CP AND THE I/O CONTROL UNIT COMMUNICATE EITHER BY MUTUALLY ALTERABLE SENSE SWITCHES OR BY INTERRUPT. THE I/O UNIT WILL SET AN INTERRUPT BIT WHEN READY TO START AND WHEN TRANSMISSION IS COMPLETE TYPICALLY. A CP CAN COMMUNICATE WITH SEVERAL DEVICES EACH HAVING ITS OWN CHARACTER SET, SO THAT THE INTERNALLY STORED VALUES DO NOT HAVE ANY INTRINSIC EXTERNAL REPRESENTATION. SUCH REPRESENTATIONS ARE PROPERTIES OF THE I/O DEVICE. INPUT FROM A TYPEWRITER USUALLY TRANSFERS A SINGLE CHARACTER OR LINE OF CHARACTERS TO A BUFFER AND INTERRUPTS THE CP WHICH READS FROM THE BUFFER TO A PACKING AREA. SCOPES ARE SOMEWHAT MORE GENERAL. THE I/O MATERIAL IS PLACED IN A BUFFER FOR THE SCOPE CONTROLLER TO DISPLAY. MOST OF THIS MATERIAL HAS TO BE INTERPRETED AS COMMANDS TO THE SCOPE. MOST SCOPES ARE RANDOM SCAN TUBES, MEANING THAT THE BEAM CAN MOVE EQUALLY EASILY TO ANYWHERE ON THE TUBE FACE AND FURTHER IMPLYING THAT ONLY THOSE POINTS EXPLICITLY MENTIONED WILL BE SCANNED. THIS IS IN CONTRAST TO A TELEVISION WHERE EVERY POINT ON THE FACE SCANNED IN TURN. OUR SCOPES ONLY HAVE TWO LEVELS OF BRIGHTNESS, BUT SOME HAVE FIVE OR MORE, TELEVISION HAS A LARGE RANGE OF BRIGHTNESS AVAILABLE. THUS THE I/O MATERIAL CONSISTS OF A SERIES OF COMMANDS TO THE BEAM TO MOVE TO A CERTAIN POINT, DRAW A LINE TO ANOTHER POINT, NOW MOVE SOMEWHERE ELSE, NOW DISPLAY A CERT AIN CHARACTER AND SO ON. THERE MAY BE SPECIAL BITS FOR BLANKING CERTAIN ELEMENT S, ALTERING THE SIZE ETC. ALSO THE SCOPES OUTPUT FUNCTION IS A REGENERATIVE PRO CESS AND WE HAVE TO INSTRUCT THE BEAM TO DO THE SAME SCAN SEVERAL TIMES A SECOND TO GIVE A CONTINUOUS DISPLAY. THUS A TYPICAL BLOCK USUALLY OF WORDS OF GRAPHIC
I/O MATERIAL IS SOMETHING LIKE THIS:

```
        A1      START SCANNING HERE
                MOVE TO X0,Y0
                DRAW LINE TO X1,Y1
                DRAW LINE TO X2,Y2
                MOVE TO X3,Y3
                DRAW CHARACTER NO 32
                LOOP BACK TO A1
```

OUR SCOPES HAVE A VERY NICE WAY OF SCANNING, THE DISPLAY MATERIAL IS SETUP AS RE LOCATABLE BLOCKS WITH TRANSFER COMMANDS WHICH MUST CONNECT UP TO GIVE A LOOP AROUND WHICH THE SCOPE SCANNER OPERATES. THE DISPLAY MATERIAL MUST BE IN ONE SP ECIAL REGION OF ADDRESSABLE

G-20 CORE VIZ. /160000 TO /177777, THE ADDRESSES USE D BY THE SCOPE SCANNER ARE RELATIVE TO /160000 AND THEREFORE RANGE FROM 0 TO /17 777. EACH OF THE THREE SCOPES CAN HAVE 4 PAGES AND INDEED EACH PICTURE IN THE CORE IS A SEPARATE MODULE OF DISPLAY MATERIAL. THE LAYOUT IS SOMETHING LIKE THIS:

```
          A1      DELIMIT A2 PAGE 1 SCOPE 1
                  DISPLAY MATERIAL
                  STORE COMMAND
          A2      DELIMIT A2 PAGE 2 SCOPE 1 AND 3
                  DISPLAY MATERIAL
                  STORE COMMAND
          A3      CYCLE TO A1
```

THE SCANNER ENTERS A MODULE, REMEMBERS THE FIRST WORD, UNTIL IT HITS A STORE COM MAND, THEN JUMPS TO THE ADDRESS MENTIONED IN THE FIRST WORD. EACH DISPLAY MODUL E CAN BE DISPLAYED ON ONE OF PAGES 1 THROUGH 4 ON ANY COMBINATION OF SCOPES 1, 2, AND 3. THE DISPLAY MATERIAL CAN BE CHANGED BY A PROGRAM FREELY ALTHOUGH ONE S HOULD ALWAYS PRESENT A WELL-FORMED DISPLAY TO THE SCANNER. INPUT OR CORRECTION OF DISPLAY MATERIAL FROM THE HUMAN AT THE SCOPE CAN BE ACHIEVED USING THE KEYBOA RD OR RAND TABLET. THE INPUT OF BITS INTO THE ADDRESSBLE MEMORY IS DONE BY THE SCANNER AS IT SCANS ROUND, ONE OF THE ADVANTAGES OF THE MODULAR LAYOUT IS THAT NEW MATERIAL IS SIMPLY APPENDED TO THE END OF THE APPROPRIATE MODULE AND THE STO RE COMMAND MOVED DOWN. THE SCANNER WILL KEEP ADDING NEW MATERIAL AS REQUESTED U NTIL IT HITS AGAINST THE NEXT DELIMIT AT WHICH TIME IT WILL GENERATE A MEMORY F ULL INTERRUPT, NOTIFYING THE SCOPE MONITOR, AND WILL REFUSE TO ENTER ANY MORE. I NPUT ACTUALLY WILL BE PLACED IN ANY DISPLAY MODULE DESIGNATED AS ENABLED FOR THA T SCOPE AND THAT INPUT DEVICE. THE DESIGNATION IS BY MEANS OF CERTAIN BITS IN T HE DELIMIT WORD. THERE IS ONE BIT TO ENABLE THE MODULE FOR ALL ENTRY, VECTORS AND CHARACTERS FROM ANY OF THE SCOPES DESIGNATED, AND TWO OTHER BITS FOR THE KEYBOA RDS FOR THE PARTICULAR SCOPE. THE FULL DELIMIT COMMAND IS

130		ADDRESS	PAGE	A	E	KEY	SCOPE	DELIMIT
31–24		23–10	9 8	7	6	5 4	3 2 1	0

THE PAGE FIELD IS 2 BITS,SO CAN BE 0=4,1,2,3. A INDICATES ALTERNATE MODE-USUALL Y ONLY USED BY SCOPE MONITOR. E IS THE GENERAL ENABLE BIT. KEY IS 2 BITS ONE F OR EACH KEYBOARD. FOUR CONSOLES ARE PROVIDED FOR BUT ONLY 3 INSTALLED. IF SEVER AL MODULES ARE ENABLED FOR THE SAME DEVICE, THE INPUT MATERIAL WILL BE ENTERED IN ALL OF THEM. THE NORMAL USER NEVER SEES OR HAS TO BOTHER WITH THE DELIMIT, STORE OR CYCLE COMMANDS, THESE ARE MANAGED FOR HIM BY THE B ROUTINES. IT IS ARRANGED AS A SET OF STRINGS,EITHER CHARACTER STRINGS OR VECTOR STRINGS WITH A HEADER COMMAND AT THE FRONT TO INDICATE THE STARTING POINT ON THE SCREEN. THUS A DISPLAY OF LINES AND CHARACTERS IS LIKE THIS

```
                  HEADER Y0 X0
                  VECTOR INCREMENT Y1 X1
```

VECTOR INCREMENT Y2 X2
HEADER Y3 X3
CHARACTERS C1 C2 C3
CHARACTERS C4 C5 C6
STORE

THE ACTUAL FORM OF THESE WORDS IS AS FOLLOWS

11	IRRELEVANT	Y_0	X_0	HEADER
31 30		19 10	9 0	

1 2 0	SY	SX	T	B	ΔY	ΔX	VECTOR INCREMENT
31 24	23	22	21	20	19 10	9 0	

SY, SX ARE SIGN BITS. B IS THE BLANKING BIT, IF SET THE VECTOR INCREMENT IS INV ISIBLE. T IS THE TAG BIT, IF SET THEN WILL BLINK OR INTENSIFY IF BLINK OR INTEN SIFY SWITCHES ARE SET.

10	C1	T1	S1	C2	T2	S2	C3	T3	S3	CHARACTERS
31 30	29 22	21	20	19 12	11	10	9 2	1	0	

T IS TAG BIT,S IS SIZE BIT, IF SET CHARACTER IS DOUBLE SIZE. CHARACTERS ARE SCOPE CHARACTER SET AS GIVEN IN THE HARDWARE MANUAL, IN A ALGOL-20 A WAY TO SET UP THESE WORDS IS SOMETHING LIKE

```
HEADER ← 8L3 + Y0 * 8R 4000 + X0;
VECTINC ← 8L120 + SY * 2↑23 + SX*2↑22
          + T * 2↑21 + B*2↑20
          + DELX*2↑10 + DELX;
CHARS ← 8L2 + (C1 * 4 + T1*2+S1)
          *2↑20
          + (C2*4 +T2*2 +S2) *2↑10
          + (C3*4 +T3*2+53);
```

SUCH MATERIAL SHOULD BE PACKED INTO AN ALGOL LOGIC ARRAY IN NORMAL CORE AND THEN B3 CALLED TO MOVE IT TO A DESIGNATED PAGE IN THE DISPLAY AREA. OTHER SCOPE OPERA TIONS AVAILABLE ARE

1 3 0		B		X	LEFT MARGIN
31 24		20		9 0	

1 7 0		B		X	RIGHT MARGIN
31 24		20		9 0	

WHICH SET MARGINS FOR MATERIAL APPEARING AFTER THEM IN THE MODULE.

1 5 0		C		COMPARE
31 24		9 2	1 0	

THIS SETS A COMPARE TRAP ON A CERTAIN CHARACTER KEYBOARD ENDING AFTER THIS WORD IN THIS MODULE. IF THIS CHARACTER IS TYPED IN, IT IS ENTERED IN THE MODULE AND AND INTERRUPT IS GENERATED BY THE SCANNER. THIS IS PROCESSED BY THE SCOPE MONITO R AND CAN BE PASSED TO A USER PROGRAM ENTRY POINT IF DESIRED.

0 1		NONZERO		NO-OP
31 30		27 24		

NO-OP HAS NO EFFECT ON THE SCANNER, CAN BE USED FOR KEEPING INFORMATION ABOUT TH E PICTURE WITH THE PICTURE FOR CONVIENT PROCESSING. NOTE THAT DOUBLE SIZE CHARA CTERS ARE OBTAINED BY SETTING THE SIZE BIT, BUT SUBSCRIPT SIZE CHARACTERS BY INC LUDING THE SUBSCRIPT SHIFT CHARACTER IN THE STRING. AN EXAMPLE OF A PROGRAM SEG MENT IN ALGOL-20 FOR DISPLAYING A SQUARE SIDE 100 STARTING AT (200,500).

```
LOGIC ARRAY A[1:20];
A[1]←HEADER(200,500);
A[2]←VECTOR(100,0,0);
A[3]←VECTOR(0,100,0);
A[4]←VECTOR(-100,0,0);
A[5]←VECTOR(0,-100,0);
B(3,LOC(A[1], PAGE, 0,0,0);
```

TO ALSO PUT THE WORD 'SQUARE' AT (200,300) WE CAN USE B0 TO CONVERT TO THE SCOPE CHARACTER CONVENTIONS

```
C[1]←'SQUA' ;
C[2]←'RE   ' ;
B(0,LOC(C[1]), LOC(A[7]), 6,0,0);
A[6]←HEADER (200,300);
B(3, LOC(A[6]),PAGE,0,0);
```

USING THE SCOPE ALGOL LIBRARY, ONE DOESN'T HAVE TO USE THIS GENERALITY IN SETTIN G UP A DISPLAY. THE SAME DISPLAY COULD BE ACHIEVED BY

```
LINE(200,500,300,500);
LINE(300,500,300,400);
LINE(300,400,300,400);
LINE(200,400,300,500);
A[1]←'SQUA';
A[2]←'RE   ';
B(2,LOC(A[1]),2,200,300,PAGE);
```

BUT WOULD BE MORE WASTEFUL, AS EACH CALL OF LINE PRODUCES A HEADER AND VECTOR IN CREMENT WORD.

6. WRITING INTERACTIVE PROGRAMS.

A. THE B ROUTINES

ALL INTERACTION BETWEEN PROGRAM AND THE SCOPES IS ACCOMPLISHED BY USING THE B ROUTINES. THE B ROUTINES ARE ACCESSED THROUGH A SPECIAL INTERFACING ROUTINE. THESE MECHANISMS NEED NEVER BOTHER THE AVERAGE USER, IF HE SIMPLY USES THE COPIES OF THE B ROUTINES IN THE SCOPE SUBPROGRAM LIBRARY (SECTION 7). THUS IN ALGOL-20 OR FORMULA ALGOL ONE SIMPLY WRITES:

B (BNUM, ARG1, ARG2, ARG3, ARG4, ARG5); AND IN SPITE

B BNUM, ARG1, ARG2, ...; BNUM IS THE NUMBER OF THE B ROUTINE REQUIRED. NOT ALL ARGUMENTS ARE USED FOR ALL B ROUTINES. MOST OF THEM HAVE TO DO WITH PASSING INFORMATION FROM THE PROGRAM TO THE SCOPES, BUT A FEW GO THE OTHER WAY; E.G., B8, WHICH READS THE ANALOG KNOBS. IN THE DESCRIPTION OF THE B ROUTINES, ARG1, ETC. ARE DENOTED BY R52 ETC. IT IS TO BE NOTED THAT THE VALUES OF THE ARGUMENTS ARE CHANGED BY A CALL ON A B ROUTINE, AND THIS CAN LEAD TO HAVOC; E.G., CALLING B8 WITH ZEROS FOR ARGUMENTS 4 AND 5 WILL CAUSE THE ALGOL CONSTANT 0 TO BE REPLACED BY ANOTHER VALUE. TO AVOID THIS ONE CAN USE A GLOBAL BOOLEAN VARIABLE OUT. IF OUT IS FALSE, THEN NO OUTPUT OF VALUES WILL OCCUR, AND NO OVERWRITING WILL OCCUR. IF OUT IS TRUE, OUTPUT WILL OCCUR, AND, IN THIS CASE, ONE CAN PUT SOME DUMMY ARGUMENTS IN THE PARAMETER LIST. IF THE LOCATION OF SOME DATA IN AN ARRAY OR SCALAR IDENTIFIER IS NEEDED, ONE MUST USE LIBRARY INTEGER PROCEDURE LOC IN ALGOL OR FORML, WHICH FINDS THE ADDRESS WHERE THE ACTUAL VALUES ARE STORED. THUS LOC (A [1]) IS THE ADDRESS CONTAINING THE VALUE OF A[1]. LOC IS IN THE SCOPE LIBRARY FOR ALGOL OR FORML. IF YOU NEED THE LOCATION OF A PROCEDURE ENTRY POINT OR LABEL, YOU USE THE LIBRARY INTEGER PROCEDURES PROCLOC OR LABELLOC RESPECTIVELY IN ALGOL-20. ALL THE ARGUMENTS TO B ARE INTEGERS. IF AN ERROR OCCURS ON CALLING B DUE TO INCORRECT ARGUMENTS, B WILL PRINT AN ERROR MESSAGE AND SET BNUM= -1. THE REASONS FOR ERRORS ARE DESCRIBED IN DETAIL IN THE DESCRIPTION OF THE B ROUTINES IN SECTION 6. THE DETAILS OF THE INTERFACE ARE GIVEN IN SECTION 10. IN ADDITION, AND ON A HIGHER LEVEL THAN THE B ROUTINES, THERE ARE SEVERAL USEFUL SUBPROGRAMS IN ALGOL, FORML AND SPITE IN THE LIBRARY FOR DOING HIGHER LEVEL TASKS. FOR EXAMPLE, PROCEDURE NUM (X, Y, N) WILL TAKE A REAL VARIABLE N AND DISPLAY IT AT X,Y IN -5D.3Z (OR F8.3) FORMAT. THE FULL I/O FACILITIES OF ALGOL-20 CAN BE USED IN READING FROM AND 'PRINTING' TO THE DISPLAY PAGE. THIS IS SIMPLY ACHIEVED USING THE SCOPE LIBRARY PROCEDURES READ.PAGE AND PRINT.ON.PAGE ,WHICH ARE EXACTLY ANALOGOUS TO READ(<W>) AND PRINT(<W>). E.G. READ.PAGE READS A CARD FROM THE SCOPE FACE INTO A BUFFER, WHICH CAN THEN BE READ IN THE USUAL WAY WITH A READ STATEMENT. AT PRESENT, A PROGRAM

CAN ONLY INTERACT WITH THE SCOPES IF IT HAS BEEN SUBMITTED FROM A SCOPE AND IF THE JOBCARD USER IS LOGGED IN ON THAT SCOPE. AT THE TERMINATION OF A USER PROGRAM CONTROL GOES TO 10 AND THENCE BACK TO SCOPE MONITOR TO ALLOW IT TO UNSET ALL THE SWITCHES SET BY THE PROGRAM. HENCE THE USER SHOULD NOT PATCH 10.

BEFORE INTERACTION CAN OCCUR THE PROGRAM MUST ANNOUNCE ITSELF BY CALLING B-1.

TO DISPLAY TEXT, ONE'S PROGRAM WILL NORMALLY SET IT UP IN G-20 CHARACTERS, SO ONE HAS TO CONVERT TO SCOPE CHARACTERS AND MOVE IT TO THE SCOPE DISPLAY REGION.

B 0 AND B 1 WILL CONVERT TEXT BETWEEN G-20 CHARACTER SET AND SCOPE CHARACTER SET.

B 2 CONVERTS TEXT AND MOVES IT TO DISPLAY REGION IN ONE OPERATION.

B 3 MOVES A REGION ALREADY IN SCOPE FORMAT TO THE DISPLAY REGION.

TO DISPLAY VECTORS, ONE MUST SET THEM UP IN A LOGIC ARRAY AND USE B 3. ONE CAN EASILY SET UP A DESIRED LOGIC ARRAY USING PROCEDURES HEADER, VECTOR, LINE, CURVE, ETC. B15, B16, B17, B18

ONE MUST RESERVE SPACE IN THE DISPLAY AREA BY CALLING B15; THE PAGE DOES NOT NEED TO BE ENABLED FOR THE PROGRAM TO ENTER DISPLAY MATERIAL BUT NEEDS TO BE ENABLED FOR THE HUMAN USER TO ENTER DISPLAY MATERIAL.

B16, B17, B18 ENABLE, DISENABLE AND DELETE A PAGE RESPECTIVELY.

B19 APPENDS ONE PAGE TO ANOTHER

B20 DISENABLES ALL PAGES.

B2 AND B3 ACTUALLY APPEND NEW DISPLAY MATERIAL TO THE EXISTING PAGE.

B28 CLEARS A PAGE. B4 AND B5 PERFORM RECIPROCAL OPERATIONS TO B2 AND B3 IN COPYING DISPLAY MATERIAL FROM A GIVEN PAGE INTO A GIVEN ARRAY IN THE USER PROGRAM.

B4 CONVERTS ALL TEXT TO G-21 CHARACTER SET AND IGNORES ALL VECTORS. THE ARRAY COULD THEN BE PRINTED OUT IN A FORMAT.

B5 COPIES WITHOUT CONVERSION. A PROGRAM CAN ONLY DEDUCE INFORMATION ABOUT THE DISPLAY BY COPYING IT INTO AN ARRAY AND SEARCHING THE AREA FOR FEATURES LIKE KEYWORDS.

B6, B7, B8, B10, B11 PROVIDE COMMUNICATION WITH THE CURSOR, ANALOG KNOBS AND USER STATE SWITCHES.

B6 READS THE CURSOR.

B7 SETS THE CURSOR.

B8 READS THE ANALOG KNOBS AND STATE

SWITCHES.

B10 READS THE STATE SWITCHES ONLY.

B11 SETS THE STATE SWITCHES.

B. USER INTERRUPTS

(I) B12,B13,B22,B24 ARE FOR COMPARE INTERRUPTS. B13 DEFINES THE USER ENTRY POINT TO BE ENTERED WHEN A COMPARE INTERRUPT OCCURS ON ANY CHARACTER. THIS OCCURS IN ANY STATE OF THE SCOPE MONITOR, EXCEPT DURING TYPING INTO THE SCOPE MONITOR, WHICH USES A COMPARE CHARACTER. B12 SETS COMPARE INTERRUPT ON A SPECIFIED CHARACTER FOR A SPECIFIED PAGE. B22 RESETS THE COMPARE ROUTINE TO THE STANDARD SCOPE MONITOR ROUTINE. B21 REMOVES COMPARE ON A SPECIFIED CHARACTER ON A SPECIFIED PAGE. B24 SETS AN ENABLED CURSOR AND INTERRUPT ROUTINE ON A SPECIFIED CHARACTER.

(II) B14,B23 ARE FOR THE MEMORY FULL INTERRUPT. B 14 SETS THE USER ENTRY POINT WHICH IS ENTERED ON MEMORY FULL. B23 RESETS MEMORY FULL ROUTINE TO THE STANDARD SCOPE MONITOR ROUTINE.

(III) B25 DEFINES THE USER ENTRY POINT FOR THE INTERRUPT BUTTONS 1-15. AFTER B25 HAS BEEN EXECUTED AND PROVIDED THE SCOPE MONITOR IS IN USER MODE, THE INTERRUPT BUTTONS WILL CAUSE AN INTERRUPT IN THE USER PROGRAM AND FOR CONTROL TO BE PASSED TO THE SPECIFIED PROCEDURE OR ENTRY POINT.

INTERRUPTING USER PROGRAMS

ALGOL PROGRAMS

IN ALGOL-20 THE ENTRY POINT OF A PROCEDURE OR THE LOCATION OF A LABEL CAN BE USED AS THE USER INTERRUPT ENTRY POINT. THE CODE FOLLOWING WILL NORMALLY MAKE DECISIONS ABOUT THE COMPUTATION AND CAN BE CALLED THE USER INTERRUPT SERVICE ROUTINE (UISR). A SIMPLE WAY TO DEFINE THE INTERRUPTS AND ENTRY POINT IS:

```
WH            LBL      T3;
WH            CLA  0   T1;
AL   V5←ACC;
AL   B(25,V5,ETC);
```

THE INTERRUPT ENTRY POINT WOULD THEN OCCUR AT THE BOTTOM OF THE PROGRAM AND BE

```
WH T1         ENT       ;
AL   ETC
```

THIS CAN BE DONE SEPARATELY FOR INTERRUPTS FROM THE BUTTONS AND FROM COMPARE CHARACTERS. THE ENTRY POINT IS TRANSFERED TO WITH A TRE INSTRUCTION SO THAT CONTROL WILL BE ON IN THE UISR UNLESS THE FIRST INSTRUCTION AFTER THE ENTRY POINT IS

```
WH              EXR  0    /77776,CE;
```

IF YOU WISH TO RETURN TO THE INTERRUPTED COMPUTATION, YOU HAVE TO POP THE MAIN MONITOR INTERRUPT STACK. THIS CAN BE DONE BY RETURNING THROUGH YOUR MARK(A PROCEDURE END IN ALGOL), SINCE THE SCOPE MONITOR PUTS ITS OWN INTERRUPT ENTRY MARK IN THE USER INTERRUPT ENTRY POINT. IF CONTROL HAS WANDERED AROUND BEFORE THE DECISION TO RETURN TO THE INTERRUPTED COMPUTATION HAS BEEN MADE, YOU MAY EFFECT THE RETURN BY RETURNING TO THE SCOPE MONITOR; HOWEVER, THE SWITCHES WILL BE UNSET BY THE CLOCK INTERRUPTS WHILE IN THE UISR, SO WE PROVIDE A B ROUTINE (B27) TO SET UP THE SWITCHES CORRECTLY AND RETURN CONTROL TO THE MAIN MONITOR. IF YOU WISH TO PASS CONTROL TO ANOTHER POINT IN THE PROGRAM, A GO TO STATEMENT WILL EFFECT ALL THE STACK PUSHING REQUIRED SO THAT THE VARIABLES ARE CORRECTLY REDEFINED.

PRINTING HAS TO BE CAREFULLY CONTROLLED IN ALGOL PROGRAMS WHICH ALLOW INTERRUPTS. IF AN INTERRUPT OCCURS DURING PRINTING AND THE UISR PRINTS, THEN IT IS MORE OR LESS IMPOSSIBLE TO RETURN TO THE ORIGINAL LINE OF COMPUTATION. ONE CAN SAVE THE INTERRUPT PRINT LINE AND PRINT LINE POINTER BY

```
   LOGIC ARRAY A[-2:120]; TEMP←|251; BUFFERSET('PRINT',A[0]);
```

AND RESTORE IT LATER. HOWEVER, IF THE NAME AND PRINT STATEMENTS IN OPERATION HAVE BEEN CHANGED, ONE CANNOT RECOVER THEM. IF THE UISR PASSES CONTROL TO ANOTHER PART OF THE PROGRAM AND NEVER WISHES TO CONTINUE AT THE INTERRUPTED POINT, THEN THE PRINTING WILL WORK OUT ALL RIGHT. SOME SAFE RULES ARE (1) DON'T PRINT OUT IN THE UISR, (2) TURN OFF THE CONTROL SWITCH DURING PRINTING; BUT, AS DISCUSSED, THESE ARE NOT RIGID RULES.

EXAMPLE OF INTERRUPT DEFINITION IN ALGOL

```
I. USING A LABEL
   BEGIN INTEGER ENPT,CSW,IN,SN,CC;
WH              LBL       T1;
WH              CLA  0    T1;
AL  ENPT←ACC;
   B(25,ENPT,LOC(CSW),LOC(IN),LOC(SN),LOC(CC));
   LOOP; <CONTINUOUS ACTIONS> ; GO TO LOOP;
WH T1           ENT       ;                 INTERRUPT ENTRY POINT
AL  ;
   <ACTIONS>; GO TO NEWACTION;
WH              TRA  1    T1;               RETURN TO INTERRUPTED
                                            COMPUTATION
AL  ;
   END;
```

II. USING A PROCEDURE

```
BEGIN
LIBRARY PROCEDURE PROCLOC;
PROCEDURE UISR;
<ACTIONS>; GO TO NEWACTION;
END GOES BACK TO INTERRUPTED ACTION;
B(25,PROCLOC(UISR),LOC(CSW),LOC(IN),LOC(SN),LOC(CC));
<CONTINOUS ACTIONS> ;
END;
```

FORMULA ALGOL PROGRAMS

FORMULA ALGOL COMPILES CODE WHICH IS HEAVILY DEPENDENT ON RUN-TIME ROUTINES. IF ANY RUN-TIME ROUTINE IS INTERRUPTED BY THE SCOPE MONITOR WHICH THEN CALLS THE UISR WHICH IN TURN CALLS THE INTERRUPTED RUN-TIME ROUTINE,THEN GLOBAL PARAMETERS(LIKE RETURN MARKS,INDEX REGISTERS AND TEMPS) ARE SOON FORGOTTEN. THEREFORE THE ONLY CODE WHICH CAN BE USED WITHOUT DRASTIC SAFEGUARDS IN THE UISR IN FORMULA ALGOL IS CODE WHICH DOES NOT CALL ON RUN-TIME ROUTINES. HOWEVER IF THE UISR AND THE CODE FOLLOWING THE CALL ON B25 ARE COMPLETELY INDEPENDENT AND DO NOT CALL ON THE SAME ROUTINES THEN ONE HAS MORE FREEDOM. OPERATIONS WHICH DO NOT USE THE RUN TIME ROUTINES INCLUDE STORING AND ACCESSING OF SIMPLE VARIABLES(BUT NOT ARRAY ELEMENTS),AND THE OPERATIONS +,-,*,/,∧,∨,¬,IF THEN ELSE, SIGN,ABS,ENTIER, AND GO TO(LOCAL BACKWARD TRANSFERS ONLY). HOWEVER, WITH INTIMATE KNOWLEDGE OF FORMULA ALGOL AND A LISTING OF ITS RUN-TIME ROUTINES , THE EXPERIENCED USER CAN BUILD HIS UISR SO THAT IT CAN CALL ON ANYTHING. THIS WOULD PROBABLY BE DONE BY WRITING SMALL MACHINE CODE ROUTINES, CALLABLE ONLY WITH CONTROL OFF, WHICH WOULD SAVE AND RESTORE THE CONTENTS OF A LIST OF MACHINE LOCATIONS. THE UISR WOULD PROBABLY LOOK LIKE THIS:

EXAMPLE OF INTERRUPT DEFINITION IN FORMULA ALGOL

```
SN CDLC  0

    PROCEDURE INTERACT; BEGIN INTEGER LOCISR;

SN CMPL  07200          ERA       NC     READ NEXT COMMAND REGISTER

SN CMPL  0050000000     ADD  0    5      NUMBER OF INTERVENING COMMANDS

SN CMPL  1330011000     STI       UISR   SAVE LOCATION OF UISR

SN CMPL  1337700001     STI       LOCISR USED IN CALLING B25
```

```
     GO AROUND;
SN CMPL  0                 UISR ENTRY POINT
SN CMPL  0760067776        EXR  0   /77777-$13-$0  ;
                                            TURN OFF CONTROL AND H MOD
SN CMPL  3770011001        TRM      SAVE  SAVE VARIABLES
     CODE PREFERABLY WITH CONTROL OFF
SN CMPL  3770011002        TRM      RESTORE  RESTORE VARIABLES
SN CMPL  6370011000        TRE  3   UISR  GO BACK TO MONITOR
    AROUND; IF B(25,LOCISR,ETC) THEN PRINT(.CANT-.INTERACT);
     END IS TO INTERACT;
```

THE ABOVE CODE AND PARAGRAPH ON THE INTERRUPTION OF FORMULA ALGOL PROGRAMS IS BY RUDY KRUTAR WHO SHOULD BE CONSULTED ON ALL RELATED MATTERS. IN FORML, THE PRINT ROUTINES ARE RECURSIVE AND THEIR VARIABLES ARE IN THE GENERAL COMMUNAL RECURSION STACK. THUS, IT SEEMS THAT ONE CAN ONLY PRINT IF THE UISR DOES NOT PRINT, AND IF IT ALWAYS RETURNS TO THE INTERRUPTED COMPUTATION. ONE SHOULD NOT INTERRUPT DURING CALLS ON MAIN MONITOR ROUTINES, IF ONE IS GOING TO USE THEM IN THE UISR, AND THEN TRY TO RETURN TO THE INTERRUPTED COMPUTATION.

C. INTERACTION WITH MORE THAN ONE SCOPE

TO INTERACT WITH A DIFFERENT SCOPE FROM THE ONE SUBMITTED FROM, A PROGRAM SIMPLY USES THE B ROUTINES AS USUAL, BUT IN ADDITION SETS THE SCOPE NUMBER BY USING ALGOL PROCEDURE SETSCOPENUM(N). IT DOES NOT NEED TO BE SET BEFORE EVERY CALL OF A BROUTINE, JUST ONCE.
THUS, TO READ THE STATE SWITCHES ON SCOPE 2, ONE PERFORMS
NSAVE←SCOPENUM; SETSCOPENUM(2); ZERO←0; OUT←TRUE;
B(10,ZERO,STSW,DUM,DUM,DUM); OUT←FALSE; SETSCOPENUM(NSAVE);

NOTE WE SAVED THE NUMBER OF THE SUBMISSION SCOPE BY USING ROUTINE SCOPENUM. SETSCOPENUM AND SCOPENUM MERELY SET AND READ INDEX REGISTER 51.

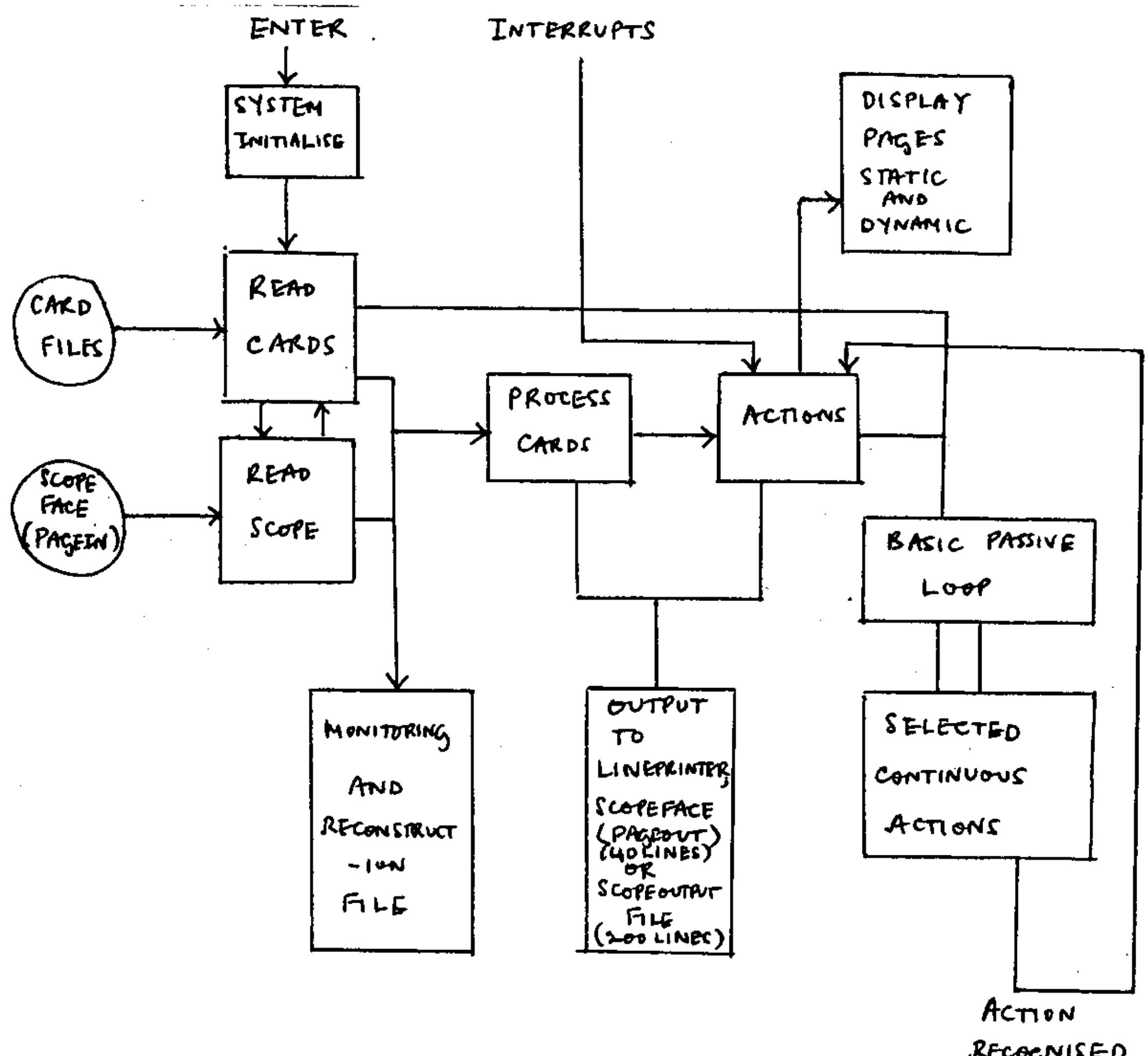
ONE CAN THUS DO ALL THE USUAL INTERACTIONS WITH ANY OTHER SCOPE. HOWEVER ,THE B ROUTINE CALLS WILL ALL GIVE ERROR EXITS UNLESS PERMISSION TO INTERACT HAS BEEN GIVEN BY THE USER AT THE SCOPE TO BE INTERACTED WITH, BY USING INTERRUPT 13 IN THE PROGRAM STATE.
USER INTERRUPTS FOR ANOTHER SCOPE ARE DEFINED BY USING B25. UPON ANY USER INTERRUPT, THE SCOPE NUMBER IS PASSED TO THE USER PROGRAM.

FINALLY,ONE CAN DISPLAY A GIVEN PAGE ON MORE THAN ONE SCOPE, BY USING B26. THIS TAKES PARAMETER R52,WHICH IS THE BIT PATTERN AT THE END OF THE DELIMIT CONSISTING OF 4 BITS FOR SCOPES 3,2,1 AND 4 RESPECTIVELY,SN CORRESPONDING TO SCOPE N.
THUS IF A PROGRAM WANTED TO DISPLAY A PAGE ON THE SUBMISSION SCOPE AND ANOTHER SCOPE N, IT WOULD DO SOMETHING LIKE THIS
N1←SCOPENUM; BITPAT←2↑N1∨2↑N; B(26,PAGE,BITPAT,0,0,0);

OF COURSE, IF PERMISSION HAS NOT BEEN GIVEN, IT WILL NOT DISPLAY.

D. OVERALL CONTROL STRUCTURE.

MIGHT BE LIKE THIS.



7. DESCRIPTION OF THE B ROUTINES.

```
B-1   ANNOUNCE AN ONLINE USER
      PARAMETERS:    USAGE NUMBER FROM LOG-IN
      OUTPUT: R51:   SCOPE NUMBER
      ERROR IF:       (I)  NOT SUBMITTED
                           FROM A SCOPE
            OR        (II)  NOT LOGGED IN
       PERMITS  A  PROGRAM TO INTERACT WITH SCOPE MONITOR.  USER
       SHOULD NOT PATCH 10.  DO NOT CALL B(-1) TWICE IN THE SAME
       RUN
B0    CONVERTS G21 CHARACTER STRINGS PACKED
      4 PER WORD INTO SCOPE STRINGS PACKED
      IN DISPLAY FORMAT
      PARAMETERS:    R52←LOCATION OF FIRST
                         WORD OF G21 TEXT.
                     R53←LOCATION TO RECEIVE FIRST WORD
                         OF THE CONVERTED TEXT BLOCK.
                         THIS ADDRESS MUST BE IN USER CORE.
                     R54←NUMBER OF CHARACTERS TO CONVERT.
      OUTPUT:        NONE
      ERROR IF:      R53 OUT OF BOUNDS.

B1    CONVERTS SCOPE CHARACTER STRINGS
      PACKED IN DISPLAY FORMAT INTO G21
      CHARACTERS PACKED 4 PER WORD.
      PARAMETERS:    R52←BASE OF SCOPE STRING.
                     R53←LOCATION TO RECEIVE FIRST
                         WORD OF THE G21 STRING.
                         THIS ADDRESS MUST BE IN
                         USER CORE.
                     R54←LENGTH OF SCOPE STRING
                         IN WORDS.
      OUTPUT:        NONE
      ERROR IF:      R53 OUT OF BOUNDS.

B2    CONVERT TEXT AND APPEND TO PAGE
      N AT POSITION (X,Y).
      PARAMETERS:    R52←BASE OF G21 TEXT
                         PACKED FOR 4 PER WORD.
                     R53←LENGTH OF TEXT IN WORDS.
                     R54←X
                     R55←Y
                     R56←PAGE NUMBER.
      OUTPUT:        NONE
      ERROR IF:       (I)  PAGE ALREADY FULL
                     (II)  STRING TOO LONG.
       DISPLAYS  TEXT  ON  SCOPE FACE.   ONE MUST HAVE REQUESTED
       AVAILABLE  SPACE  FOR  THE PAGE IN QUESTION IN ADVANCE OF
       CALLING B2.
```

```
B3    APPEND A LOGIC BLOCK OF
      (ALREADY CONVERTED) DISPLAY
      MATERIAL TO PAGE N.
      PARAMETERS:   R52←BASE OF BLOCK
                        TO BE MOVED.
                    R53←PAGE NUMBER.
      OUTPUT:       NONE.
      ERROR IF:      (I)  NO STORE IN PAGE.
           OR       (II)  ATTEMPT TO OVERLAY
                          A DELIMIT.
```

MOVES A BLOCK OF SCOPE COMMANDS INTO THE H-MODULE AND APPENDS IT TO AN EXISTING PAGE. CURRENTLY IT ONLY TESTS FOR DELIMITS AND CYCLES; OTHER ILLEGAL CONSTRUCTIONS ARE NOT TESTED FOR; THUS, THE DISPLAY SHOULD BE WELL FORMED AND SHOULD INCLUDE A STORE.

```
B4    MOVE PAGE N TO USER CORE, CONVERTING
      ALL SCOPE CHARACTERS.
      PARAMETERS:   R52←PAGE NUMBER.
                    R53←LOCATION IN USER CORE TO
                        RECEIVE CONVERTED TEXT.
      OUTPUT:       NONE.
      ERROR IF:      (I)  R53 OUT OF BOUNDS.
                    (II)  PAGE HAS NO BLOCKS.
```

TEXT IS ENTERED INTO USER MEMORY. NOTE THAT ONLY G21 CHARACTERS ARE CONVERTED AND ALL ELSE IS IGNORED IN THE CONVERSION PROCESS. NON-G21 CHARACTERS ARE CONVERTED TO BLANKS AND VECTORS ARE SKIPPED.

```
B5    MOVE A PAGE TO USER CORE
      WITHOUT CONVERSION.
      PARAMETERS:   R52←PAGE NUMBER.
                    R53←LOCATION IN USER CORE
                        TO WHICH THE BLOCK
                        WILL BE MOVED.
      OUTPUT:       NONE.
      ERROR IF:      (I)  R53 OUT OF BOUNDS.
           OR       (II)  PAGE HAS NO BLOCKS.
```

EVERYTHING FOLLOWING THE DELIMIT IS MOVED TO USER CORE. BE PREPARED TO ACCEPT THE FULL PAGE.

```
B6    READ THE CURSOR.
      PARAMETERS:   NONE
      OUTPUT:       R52←X
                    R53←Y
```

THE POSITION OF THE CURSOR IS OBTAINED FROM THE POSITION WORD IN THE H-MODULE.

```
B7    SET THE CURSOR.
      PARAMETERS:   R52←X
```

```
                R53←Y
     OUTPUT:    NONE.
     THE POSITION WORD IS CHANGED SO THAT THE CURSOR IS
     REPOSITIONED AT (X,Y).

B8   READ THE ANALOG KNOBS.
     PARAMETERS:  NONE.
     OUTPUT:      R52←KNOB 1
                  R53←KNOB 2
                  R54←USER STATE SWITCHES
                      BIT PATTERN.
                  R55←ALT(1) OR NORMAL(0) MODE.
     GETS THE POSITIONS OF THE ANALOG KNOBS FROM THE POSITION
     WORD IN THE H-MODULE.

B9   NOT YET SPECIFIED.

B10  READ THE STATE SWITCHES.
     PARAMETERS:  R52←ALT(1) OR NORMAL(1) MODE.
     OUTPUT:      R53←STATE WORD.

B11  SET THE STATE SWITCHES.
     PARAMETERS:  R52←DESIRED SETTING
                      OF STATE WORD.
     OUTPUT:      NONE.
     LOADS R52 INTO THE STATE WORD.

B12  SET COMPARE ON CHARACTER
     ON PAGE N.
     PARAMETERS:  R54←CHARACTER TO
                      COMPARE ON.
                  R55←PAGE NUMBER.
     OUTPUT:      NONE.
     ERROR IF:      (I)   ILLEGAL PAGE NUMBER
                   (II)   NO ROOM LEFT ON PAGE
          OR      (III)   DELIMIT FOLLOWS STORE.
     CREATES A COMPARE COMMAND FOR THE SUPPLIED SCOPE
     CHARACTER,E.G.RETURN WOULD BE 8R75, AND INSERTS IT IN
     PAGE N IMMEDIATELY FOLLOWING THE DELIMIT. THE PROGRAM
     SHOULD DEFINE THE COMPARE ROUTINE BEFOREHAND.

B13  SET COMPARE ROUTINE.
     PARAMETERS:  R55←COMPARE CHARACTER LOCATION FOR VALUE
                  R56←ADDRESS OF
                      USER ROUTINE.
     OUTPUT:      NONE.
     ERROR IF:    R56 OUT OF BOUNDS.
     SETS USER ROUTINE TO BE EXECUTED WHEN A COMPARE INTERRUPT
     OCCURS. NOTE THAT THE ROUTINE MAY BE EXECUTED AT ANY
     TIME.
```

```
B14  SET MEMORY FULL ROUTINE.
     PARAMETERS:   R56←ADDRESS OR USER
                        ROUTINE.
     OUTPUT:       NONE.
     ERROR IF:     R56 OUT OF BOUNDS.
      SETS USER ROUTINE TO BE EXECUTED WHEN A MEMORY FULL
      INTERRUPT IS GENERATED. NOTE THAT THIS ROUTINE MAY BE
      EXECUTED AT ANY TIME.
B15  GET N BLOCKS FOR PAGE M.
     PARAMETERS:   R52←PAGE NUMBER
                   R53←NUMBER OF BLOCKS
     OUTPUT:       NONE.
     ERROR IF:     ILLEGAL PAGE NUMBER.
     SAME TASK AS ON MANAGEMENT PAGE.

B16  ENABLE PAGE N.
     PARAMETERS:   R52←PAGE NUM≠ER.
     OUTPUT:       NONE.
     ERROR IF:     ILLEGAL PAGE NUMBER.

B17  DISENABLE PAGE N
     PARAMETERS:   R52←PAGE NUMBER.
     OUTPUT:       NONE.
     ERROR IF:     ILLEGAL PAGE NUMBER.

B18  DELETE PAGE N.
     PARAMETERS:   R52←PAGE NUMBER
     OUTPUT:       NONE.
     ERROR IF:     ILLEGAL PAGE NUMBER.
      SAME AS TASK IN OPTION STATE. PAGE IS RETURNED TO
      AVAILABLE SPACE AND INFORMATION IS LOST.

B19  APPEND PAGE N TO PAGE M.
     PARAMETERS:   R52←PAGE NUMBER N.
                   R53←PAGE NUMBER M.
     OUTPUT:       NONE.
     ERROR IF:     ILLEGAL PAGE NUMBFR.
      SAME TASK AS IN OPTION STATE.

B20  DISENABLE ALL INPUT FROM THIS
     SCOPE.
     PARAMETERS:   NONE.
     OUTPUT:       NONE
      DISENABLES ALL PAGES FOR THE GIVEN SCOPE.

B21  REMOVE COMPARE ON CHARACTER
     ON PAGE N.
     PARAMETERS:   R54←CHARACTER TO
                        COMPARE ON.
                   R55←PAGE NUMBER.
     OUTPUT:       NONE.
```

ERROR IF: ILLEGAL PAGE NUMBER.
SEARCHES THE PAGE FOR AN OCCURENCE OF A COMPARE COMMAND ON THE SPECIFIED CHARACTER AND IF FOUND, CONVERTS IT TO A STORE COMMAND.

B22 RESET COMPARE ROUTINE.
PARAMETERS: NONE.
OUTPUT: NONE.
RESETS THE STANDARD MONITOR ROUTINE FOR THE COMPARE ROUTINE.

B23 RESET MEMORY FULL ROUTINE.
PARAMETERS: NONE.
OUTPUT: NONE.
RESETS THE STANDARD MONITOR ROUTINE FOR THE MEMORY FULL ROUTINE.

B24 SET ENABLED CURSOR AND INTERRUPT ON CHARACTER.
PARAMETERS: R52←X
R53←Y
R54←CHARACTER FOR COMPARE.
R55←PAGE NUMBER.
R56←COMPARE ROUTINE.
OUTPUT: NONE.
ERROR IF: (I) ILLEGAL PAGE NUMBER
(II) NO ROOM LEFT ON PAGE,
(III) DELIMIT FOLLOWS STORE,
OR (IV) R56 OUT OF BOUNDS.
THIS ROUTINE DIRECTLY CALLS B7, B12, B13, AND B16. IT ENABLES THE PAGE, POSITIONS THE CURSOR AT (X, Y), SETS A COMPARE ON THE SPECIFIED CHARACTED AND SETS THE COMPARE ROUTINE. TO OBTAIN THE CHARACTER WHICH CAUSED THE INTERRUPT,B13 SHOULD BE ALSO CALLED,PASSING THE IDENTIFIER IN WHICH THIS INFORMATION SHOULD BE PUT. ALSO TO OBTAIN THE SCOPENUMBER,AND TO USE A CONTROL SWITCH,B25 SHOULD BE CALLED AS WELL.

B25 DEFINE USER INTERRUPTS.
PARAMETERS: R52←USER ENTRY POINT.
R53←USER CONTROL SWITCH.
R54←INTERRUPT NUMBER.
R55←SCOPE NUMBER.
R56←COMPARE CHARACTER.
OUTPUT: NONE.
ERROR IF: USER ENTRY POINT DOES NOT LIE IN USER CORE.
IN USER MODE, CONTROL IS PASSED TO THE USER ENTRY POINT, AND THE INTERRUPT NUMBER, THE SCOPE NUMBER OF THE SCOPE WHICH INTERRUPTED, AND THE COMPARE CHARACTER, IF THIS APPLIES, ARE PLACED IN THE LOCATIONS SET ASIDE FOR THEM IN THE USER PROGRAM. THESE LOCATIONS ARE DECLARED IN R54,

R55, R56 WHEN USING B25. THE USER CONTROL SWITCH ALLOWS THE USER PROGRAM TO DECLARE ITSELF INTERRUPTABLE AS DESIRED. IF IT IS NOT EQUAL TO ZERO WHEN THE INTERRUPT OCCURS, THE SCOPE MONITOR DOES NOT PASS CONTROL BUT KEEPS LOOKING ONCE A SECOND UNTIL THE VALUE OF THE SWITCH IS ZERO. A SECOND INTERRUPT DURING THIS TIME WILL GIVE MULTIPLE INTERRUPT ERROR AND BE IGNORED, BUT THE FIRST ONE WILL STILL BE PROCESSED CORRECTLY.

B26 SET CRT FIELD ON PAGE N.

PARAMETERS: R52←N
R53←BITS FOR CRT FIELD.

OUTPUT: NONE.

THIS ROUTINE ALLOWS THE USER PROGRAM TO DISPLAY ON MORE THAN ONE SCOPE. THE T22 TABLE IN THE SCOPE MONITOR HAS BIT PATTERNS FOR EACH SCOPE INDICATING THAT THE HUMAN HAS ALLOWED INTERACTION WITH PROGRAMS FROM OTHER SCOPES. THE NORMAL ENTRIES ARE $1, $2, AND $3, RESPECTIVELY. IF SCOPE 2 ALLOWED INTERACTION WITH PROGRAM FROM SCOPE 1, ITS ENTRY WOULD BE CHANGED TO $1 + $2. THE USER PROGRAM INDICATES ITS DESIRE TO DISPLAY IN BOTH SCOPES USING B26 AND PASSING THE LIST PATTERN $1 +$2. THE USER PROGRAM CAN ONLY DISPLAY ON THOSE SCOPES FOR WHICH PERMISSION HAS BEEN GIVEN, SINCE THE BIT PATTERN IS EXTRACTED WITH THE ENTRY IN THE T22 TABLE. THE EXTRACTED PATTERN BECOMES THE CRT FIELD OF THE DELIMIT OF SUBSEQUENTLY PRODUCED DISPLAY PAGES.

B27 RETURN TO INTERRUPTED COMPUTATION.

PARAMETERS: NONE.
OUTPUT: NONE.

USE IN THE USER INTERRUPT SERVICE ROUTINE TO CONTINUE THE INTERRUPTED COMPUTATION. IF YOU WISH TO CHANGE TO A DIFFERENT LINE OF COMPUTATION, USE A GO TO STATEMENT.

B28 CLEAR A PAGE.

PARAMETERS: R52←PAGE NUMBER.
OUTPUT: NONE.
ERROR IF: ILLEGAL PAGE NUMBER.

INSERTS A STORE COMMAND AFTER THE DELIMIT ON PAGE N. NOTE THAT ATTEMPTING TO CLEAR A PAGE OF ZERO LENGTH WILL ZERO A DELIMIT AND PERHAPS DESTROY INFORMATION.

THE FOLLOWING ARE NOT YET IMPLEMENTED

B29 MOVE PAGE N TO FILE M.

PARAMETERS: R52←N
R53←M

```
        OUTPUT:          NONE.
B30     MOVE LOGIC BLOCK BASE N LENGTH
        L TO FILE M.
        PARAMETERS:      R52←N
                         R53←L
                         R54←M
        OUTPUT:          NONE
B31     MOVE FILE M TO PAGE N.
        PARAMETERS:      R52←M
                         R53←N
        OUTPUT:          NONE.

B32     MOVE FILE M TO LOCATION N.
        PARAMETERS:      R52←M
                         R53←N
        OUTPUT:          NONE.
B33     READ IN A 3-DIGIT INTEGER
        AT [X, Y].
        ERROR IF:        ILLEGAL PAGE NUMBER.
        PARAMETERS:      R52←X
                         R53←Y

B34     READ IN A STRING OF CHARACTERS
        AT [X,Y].
        PARAMETERS:      R52←X
                         R53←Y

B40     GETS SCOPE MAN NUMBER,
        GIVEN G-20 MAN NUMBER.
        PARAMETERS:      R52←G-20 MAN NUMBER.
        OUTPUTS:         R53←SCOPE MAN NUMBER.

B41     GET SCOPE MONITOR SYMBOL N.
        PARAMETERS:      R52←N
        OUTPUTS:         R53←VALUE OF SCOPE
                             MONITOR SYMBOL.
        N =     1     U35,   FETCH A MODULE
                2     T80,  PAGE ADDRESS TABLE
                3     U29,   RELEASE A MODULE
                4     U5,    PUSH THE STACK
                5     U6,    POP THE STACK
                6     U17,   EXIT
                7     T15,   CONVERSION TABLES FROM
                             G-20 → SCOPE CHARACTERS
                8     T31,   SCOPE MONITOR TIME USED TODAY
                9     Y6,    TRACE
               10     Y105,  TRACE BREAKPOINTS
               11     Y72,   TRACE TABLES
               12     T74,   ISR RETURN POINT
```

TO OBTAIN THE ADDRESS OF A PAGE

```
 AL  DUM←2; OUT←TRUE; B(41,DUM,T80,DUM,DUM,DUM);
       OUT←FALSE; ADDR←T80-1+SCOPENUM+3*PAGE;
```

ERROR NUMBERS.

THE SCOPE MONITOR PASSES AN INTEGER IN THE ACCUMULATOR WHICH IS THE LOCATION IN THE SCOPE MONITOR WHERE THE ERROR WAS DETECTED. THE FOLLOWING TABLE RELATES THESE INTEGERS TO THEIR MEANINGS.

ERROR NO	BROUTINE	MEANING
1	-1	INTERACTION UNACCEPTABLE,EITHER (A)REMOTE FROM WHICH JOB WAS SUBMITTED IS NOT A SCOPE I,E. NOT IN [5,7]. OR(B)MANNUMBER OF USER LOGGED IN ¬=MANNUMBER ON JOB CARD OF PROGRAM.
167003	SEVERAL	PROGRAM ATTEMPTING TO INTERACT WITH A SCOPE FOR WHICH PERMISSION HAS NOT BEEN GIVEN.
167014		BROUTINE WITH THIS NUMBER DOESN'T EXIST.
167145	2	PAGE DOESN'T EXIST
167164	2	NO ROOM LEFT ON PAGE
167211	3	NOT ENOUGH SPACE.
167355	12	EITHER(A) NO PAGE EXISTS OR(B)NO STORE COMMAND FOUND ON PAGE OR(C) A DELIMIT IMMEDIATELY FOLLOWS THE STORE(PAGE FULL)
167446	17 OR 18	PAGE NO. NOT IN[1,4].
171365	SEVERAL	PAGE NO.NOT IN [1,4].
171372	SEVERAL	PAGE DOESN'T EXIST.
171402	SEVERAL	LOCATION GIVEN IS NOT IN USER CORE I.E. NOT IN [/10000,/73000].
171406	SEVERAL	LOCATION GIVEN IS NOT IN UPPER CORE I.E. NOT IN [/160000,/177777].

8. SUBPROGRAM LIBRARY.

WE HAVE ONLY JUST STARTED TO SET UP THIS LIBRARY. LISTINGS CAN BE OBTAINED FROM THE RESPECTIVE AND FILES. USER CR38AB14:

ALGOL SUBPROGRAMS FILE 32/P;
FORMULA ALGOL SUBPROGRAMS FILE 31/P;
SPITE SUBPROGRAMS FILE 33/P;

PROCEDURES IN ALGOL AND FORML THE SCOPE ALGOL LIBRARY CAN NOW BE USED AS AN OUTER BLOCK TO ANY ALGOL PROGRAM. YOU NEED AN EXTRA END,OF COURSE. THESE PROCEDURES WERE WRITTEN BY RUDY KRUTAR,JIM KING,ALAN BOND AND DAVE VAVRA. THE LIBRARY IS CURRENTLY BEING MAINTAINED AND EXTENDED BY RUSSELL MOORE,TO WHOM SUGGESTIONS AND QUERIES SHOULD BE DIRECTED.

1. INTEGER PROCEDURE LOC(N); INTEGER N; GIVES THE ADDRESS WHERE THE VALUE OF AN IDENTIFIER IS STORED. FOR ARRAYS, LOC (A[1]) WILL GIVE THE 1ST WORD OF THE ARRAY.

2. LOGIC PROCEDURE DECML (NUMBER); INTEGER NUMBER; GETS THE DECIMAL G-20 CHARACTERS FOR THE VALUE OF NUMBER AND PACKS THEM IN DECML.

3. BOOLEAN PROCEDURE B(BNUM, B52, B53, B54, B55, B56) VALUE BNUM; INTEGER BNUM. B52, B53, B54, B55, B56; CALLS B-ROUTINE NUMBER BNUM. ON ERROR EXIT, B IS TRUE. NORMAL EXIT FALSE. HENCE, IF B( ) THEN GO TO EXIT; WILL CALL THE B ROUTINE.

4. AN ALTERNATIVE VERSION OF B, WHICH HAS GLOBAL BOOLEAN VARIABLES OUT AND PR. IF OUT IS TRUE PARAMETERS ARE OUTPUT. IF PR IS TRUE, THE VALUES OF PARAMETERS AND NATURE OF EXIT ARE PRINTED.

5. PROCEDURE BA(BNUM,ETC) IS A PROCEDURE RATHER THAN A FUNCTION AND CALLS ON B.

6. LOGIC PROCEDURE HEADER (X,Y); VALUE X,Y; INTEGER X,Y; COMPUTES A HEADER INSTRUCTION AT X,Y. NOTE THAT X,Y MUST LIE IN [0, 1023].

7. LOGIC PROCEDURE VECTOR (X,Y,SG); VALUE X,Y,SG; INTEGER X,Y; LOGIC SG; COMPUTES A VECTOR STRING ELEMENT WITH DX = X, DY = Y. SG = 0 USUALLY, SG = 2 FLAGS THE VECTOR SO THAT IT WILL BLINK

OR INTENSIFY ACCORDING TO THE SETTING OF STATE SWITCHES.

8. LOGIC PROCEDURE CHARAC (C, SG, I); INTEGER C SG,I; PRODUCES A SCOPE CHARACTER IN A WORD IN POSITION I = 1, 2, OR 3. SG IS THE TAG FIELD, SO SG = 2 GIVES BLINKING AND INTENSIFICATION. C IS THE SCOPE CHARACTER NUMBER AS GIVEN IN THE QUATSE MANUAL.

9. LOGIC PROCEDURE CHARSTR (C1, C2, C3, SG1, SG2, SG3);

10. PROCEDURE CHARACTER (X, Y, C); INTEGER X, Y; LOGIC; PUTS A CHARACTER ON THE SCREEN AT POINT X, Y. C IS A SCOPE CHARACTER-STRING WORD AS OBTAINED BY USING CHARAC OR CHARSTR. IT CAN ALSO BE OBTAINED BY

C←8L2 + 8RN * 4; WHERE N IS THE SCOPE CHARACTER NUMBER FROM THE QUATSE MANUAL. CHARACTER HAS ONE CHARACTER PER WORD AND A SEPARATE HEADER FOR EACH CHARACTER, AND IS, THEREFORE, WASTEFUL OF SPACE. G-20 CHARACTERS ARE BEST PUT ON THE SCREEN USING 32.

11. PROCEDURE NUM(X, Y, N); VALUE N; INTEGER X, Y; REAL N; TAKES A REAL NUMBER N, FINDS DECIMAL CHARACTER FORM, AND PUTS IT ON THE SCREEN AT X,Y IN -5D.3Z FORMAT.

12. PROCEDURE LINE (X1, Y1, X2, Y2); INTEGER X1, Y1, X2, Y2; PUTS A LINE FROM (X1, Y1) TO (X2, Y2), WITH A SEPARATE HEADER.

13. PROCEDURE GENERATE (X, Y, T, DT, NORM); VALUE DT, NORM; REAL X, Y, T, DT, NORM; GENERATES A CURVE WITH PARAMETER T WHOSE X, Y ARE GIVEN BY EXPRESSION INVOLVING T WHEN ACTUALLY CALLING GENERATE. THUS

GENERATE(A * SIN(T), B * COS(T),T,DT,NORM);
WILL PLOT AN ELLIPSE.

IT DOES IT BY LINE SEGMENTS, AND IT CALCULATES THESE FOR INTERVALS IN T OF DT. IT ASSUMES A SQUARE SCREEN WHOSE LINEAR SIZE IS NORM IN RELATION TO THE VALUES OF X, Y.

14. CURVE (X, Y, T, DT, TA, TB); INTEGER X, Y; REAL T, DT, TA, TB; PLOTS FROM TA TO TB.

15. INTEGER PROCEDURE SCALEX (X); REAL X;

INTEGER PROCEDURE SCALEY (Y); REAL Y; THESE ALLOW EASY SCALING. GLOBAL VARIABLES XA, XB, YA, YB, SXA, SXB, SYA, SYB INDICATE THAT THE PART OF THE SCREEN USED WILL BE FROM SXA TO SXB AND SYA TO SYB, WHERE THESE LIE IN [0, 1023], AND THAT THIS WILL CORRESPOND TO VALUES XA, XB, YA, BY IN THE REST OF THE COMPUTATION. THUS

CHARACTER (SCALEX (X), SCALEY (Y),C); PUTS A CHARACTER ON THE SCREEN AT POINT X, Y IN THE USERS SCALE.

THERE ARE AN EQUIVALENT SET OF PROCEDURES TAKING REAL ARGUMENTS FOR POSITION AND USING SCALE X AND SCALE Y. THESE ARE DESIGNATED BY AN ADDED 1 ON THE NAME. THUS, CHARACTER1, NUM1, LINE1, CURVE1, ETC.

16. READ.PAGE(N,RBUFF); READS THE CONTENTS OF PAGE N(MAX LENGTH 1 BLOCK) INTO THE READ BUFFER RBUFF PACKED 1 CHARACTER PER WORD ,SO IT IS LIKE A NORMAL CARD READ. YOU MAY THEN READ FROM RBUFF USING AL THE NORMAL FORMATTING POWER OF ALGOL.

17. PRINT.ON PAGE(N,WBUFF,X,Y); PUTS THE CONTENTS OF PRINT BUFFER WBUFF AFTER NORMAL ALGOL PRINTING (WHICH CAN BE WITH OR WITHOUT <E> OR <W>) ONTO PAGE N AT X,Y. THUS THE FULL GENERALITY OF ALGOL I/O IS AVAILABLE FOR COMMUNICATION WITH THE SCOPES.

18. SETSCOPENUM(N); INTEGER N; SETS SCOPE NUMBER TO N, SO THAT FURTHER CALLS OF B ROUTINES APPLY TO THIS SCOPE. THEY OF COURSE GIVE AN ERROR IF PERMISSION HAS NOT BEEN GIVEN MANUALLY AT THE SCOPE.

19. INTEGER PROCEDURE SCOPENUM; GIVES THE SCOPE NUMBER CURRENTLY SELECTED. IT SHOULD NOT BE CO*FUSED WITH THE SCOPE NUMBER PASSED UPON INTERRUPT.

20. BUTTIN(ENPT,CNTRLSW,INTNUM,SCOPNUM,PAGEIN,PAGEOUT); INTEGER ARGUMENTS. DEFINES BUTTON INTERRUPTS,DISPLAYS 'INTERRUPTS NOW DEFINED' ON P1AGEOUT AND CONTINUES COMPUTING. ON INTERRUPT, IT PASSES CONTROL TO ENPT, WHICH CAN BE A CLOSED PROCEDURE OR A LABEL. IT PUTS'INTERRUPT NUMBER ...' ON PAGEOUT UPON INTERRUPT. CNTRLSW=0 INHIBITS INTERRUPTS. INTNUM IS THE NUMBER OF THE BUTTON INTERRUPTING AND SCOPNUM IS THE NUMBER OF THE SCOPE INTERRUPTING.

21. COMIN(ENPT,CNTRLSW,SCOPNUM,CHAR,PAGEIN,PAGEOUT,CH); SETS COMPARE INTERRUPT ON CHARACTER CHAR ON PAGE PAGEIN. SIMILAR TO BUTTIN. ON INTERRUPT, THE CHARACTER WHICH CAUSED THE INTERRUPT WILL BE FOUND IN IDENTIFIER WHOSE LOCATION IS CH.

NOTE THAT CONTIGUOUS DECLARATION OF SCALARS IN FORML GIVES ALLOCATIONS IN SUCCESSIVE WORDS, WHEREAS IN ALGOL IT GIVES CONTIGUOUS LOCATIONS BUT IN THE REVERSE ORDER TO THE ORDER OF DECLARATION.

THUS, LOGIC D1, D2, D3; DUMPS (3, D3); PRINTS THE CONTENTS OF D3, D2 AND D1.

MACROS AND ROUTINES IN SPITE

1. MACRO BC XX1,XX2,XX3,XX4,XX5,XX6; CALLS INTERFACE ROUTINE IO. EXPECTS ALL ARGUMENTS TO BE CONSTANTS, I.E., FIXED AS ASSEMBLY TIME.

2. MACRO BV XX1, XX2, XX3, XX4, XX5, XX6. EXPECTS XX1 TO BE CONSTANT AND XX2 ... XX6 TO BE VARIABLES, I.E., BE LOCATIONS WHICH CONTAIN THE DESIRED ARGUMENTS.

3. IO THE INTERFACE ROUTINE.

4. THERE IS A VARIANT ON B WHICH PUTS A MESSAGE ON THE G-20 TYPEWRITER ASKING FOR THE H-MODULE TO BE SWITCHED, IF IT ISN'T.

5. SOME MACROS TO EASILY GENERATE SCOPE DISPLAY MATERIAL HEADR, VEC, CWD, STOR.

9. USER SUBSYSTEMS.

AN INTERACTIVE PROGRAM ON THE G-20 IS INEFFICIENT IN ITS USE OF COMPUTER TIME IN THAT IT OFTEN IS IN A LOOP WAITING FOR THE HUMAN TO TELL IT WHAT TO DO NEXT. ALSO THE PROGRAM MUST WAIT IN THE QUEUE BEFORE IT CAN BE INITIATED. THE PAUSE SYSTEM IS USEFUL FOR GETTING SHORT BURSTS OF USER PROGRAM.

IN IMPLEMENTING A TIME SHARING SWAPPING SYSTEM FOR THE SCOPE MONITOR, IT WAS FOUND EASY TO ALLOW ANY USER TO WRITE SUBPROGRAMS OF RELOCATABLE REENTRANT ASSEMBLY CODE WHICH ARE ORGANIZED BY THE SCOPE MONITOR, SWAPPED IN AN OUT AS REQUIRED AND AS SPACE PERMITS, RELOCATED IN CORE AND LINKED TOGETHER DYNAMICALLY IN A SIMPLE WAY. THERE IS ALSO AN AUXILIARY MACRO SYSTEM WHICH ALLOWS THE CONVERSION OF ORDINARY ASSEMBLY CODE INTO THE REQUIRED REENTRANT RELOCATABLE MODULAR FORM. IT TURNS OUT THAT ANY MODULE OF ANY USER CAN CALL ANY MODULE OF ANY OTHER IN AS VIOLENTLY A RECURSIVE WAY AS REQUIRED, AND THAT ONLY ONE COPY OF ANY MODULE IS IN CORE EVEN IF CALLED BY SUBSYSTEMS FROM ALL THREE SCOPES AT ONCE.

TO CONVERT CODE TO MODULAR FORM, ONE USES THE MACROS AND ROUTINES ON USER CR38AB14, FILE 34. THEN ONE BREAKS THE CODE AS FOLLOWS:

LAYOUT		EXAMPLE
FILE 34 PACKAGE		USER CR38AB14; FILE 34/P; INSERT $
NON RELOCATABLE (GLOBAL) SYMBOL DECLARATIONS		LBL T90;
BEGIN		BEGIN
RELOCATABLE (LOCAL) SYMBOL DECLARATIONS		LBL E20;
ENTRY POINT DECLARATIONS		ENPT 1, E1
		ENPT 2, E2
(REENTRANT) CODE	E1	ENT
		PUSH 5;
		CLA 0 1;
		STL 2,50
		TRM E3
		EXIT
OR NONREENTRANT	E2	ENT
		TRM E4
		TRA 1 E1
EXTERNAL IDENTIFICATIONS TO	E3	ISMOD 1, 5, 'AB14;
ENTRY POINTS OF OTHER MODULES	E4	ISMOD 2, 7, 'AD03;
E.G. E3 IS ENTRY POINT 1 OF MODULE 5 OF USER AB14		
END		END
STORE		STORE 6, 'AB14;
STORES THE GENERATED MODULE, E.G., AS MODULE 6 OF USER AB14		

THE PUSH MACRO DECLARES STACK VARIABLES, E.G., PUSH 5 DECLARES 5 VARIABLES AND PUSHES THE STACK. ONE THEN USES THESE VARIABLES WITH THE POINTER IN REGISTER 50.

```
THUS    CLA   2,50   CLEAR AND ADD SECOND
                     STACK VARIABLE.
        STI   3,50   PUT IN THIRD STACK
                     VARIABLE.
```

PUSH MUST IMMEDIATELY FOLLOW THE ACTUAL ENTRY POINT (TO ALLOW THE MARK TO BE STACKED). A REENTRANT ROUTINE DOES NOT EXIT THROUGH ITS MARK BUT THROUGH THE STACKED MARK USING POP N, WHERE N IS THE NUMBER OF STACK VARIABLES IN THE ROUTINE. TO EXIT BACK TO THE SCOPE MONITOR USE EXIT. THE STACKING, INCLUDING STACKING THE MARK INTO 1, 50 (WHICH SHOULD THEREFORE NOT BE USED BUT ALWAYS ALLOWED FOR) AND PUSHING, POPPING, ERROR RECOVERY IS ALL DONE BY THE SCOPE MONITOR.

STACKED VARIABLES HAVE TO BE USED TO KEEP THE VALUES OF VARIABLES NEEDED DURING RECURSIVE CALLS OR ANY TIME THE CODE MAY HAVE TO WAIT. TO SWAP IN ANOTHER MODULE ONE HAS TO WAIT FOR THE DISC. SO STACKED VARIABLES HAVE TO BE USED FOR ANY VALUES, SET BEFORE ANY TRM, WHICH ARE REFERRED TO AGAIN AFTER THE RETURN

THROUGH THE MARK. THIS IS BECAUSE ANOTHER USER MAY ENTER THE SAME CODE DURING THE WAIT. ONE CAN HAVE MODULES OF REGULAR CODE BUT IT CANNOT CALL ITSELF RECURSIVELY, AND CANNOT BE SHARED BY ANY OTHER SYSTEM. NOTE THAT EACH USER HAS HIS OWN NAMES FOR ALL HIS IDENTIFIERS. HAVING CREATED THE SUBSYSTEM, IT CAN BE LOADED FROM THE PROGRAM STATE. MODULES WILL NORMALLY BE MARKED AS DISPENSIBLE AFTER USE, AND ARE LIKELY TO BE SWAPPED OUT IF THE SPACE IS NEEDED FOR SOMETHING ELSE; HOWEVER, THE USER CAN MARK ANY MODULE AS 'RETAINED' WITH AN INTERRUPT ON THE PROGRAM PAGE. HE CAN 'RELEASE' ALSO. LOADING A MODULE AUTOMATICALLY RETAINS IT, OR ONE CAN SIMPLY ASK TO TRANSFER TO A MODULE WHICH WILL LOAD IT IF NECESSARY, EXECUTE IT AND RELINQUISH IT.

THE ADVANTAGE OF SUBSYSTEMS IS, OF COURSE, THEIR EFFICIENCY -- THEY CAN BE USED ON AN INTERRUPT BASIS WITHOUT SUBMITTING A G-20 PROGRAM. A SUBSYSTEM CAN USE B ROUTINES TO SET UP DISPLAYS, ETC. IN PRINCIPLE, ASSEMBLY CODE AND EVEN OCTAL CODE GENERATED BY A COMPILER CAN BE CONVERTED TO SUBSYSTEM FORM, SPACE PERMITTING. IN ORDER TO HAVE A DATA AREA TO WORK ON, IT IS SUGGESTED THAT SOME MODULES BE RESERVED AS DATA AREAS WITH THE ENTRY POINTS GOING TO DATA ACCESSING FUNCTIONS. SUCH MODULES COULD THEN BE LOADED AND RETAINED IN CORE AND THE CODE MODULES BE PURE PROCEDURES WHICH COULD SWAP IN AND OUT AND MANIPULATE THIS DATA. THEY ARE NOT ACTUALLY SWAPPED OUT, JUST RELEASED TO AVAILABLE SPACE, AND, WHEN NEXT NEEDED, A NEW COPY SWAPPED IN.

FOR PASSING PARAMETERS INDEPENDENTLY OF PARTICULAR DATA AREAS, REGISTERS 52-58 CAN BE USED. THESE ARE SAVED DURING WAITING FOR THE DISC TO SWAP IN THE NEXT MODULE.

THE TEXT EDITOR IS A SEPARATE SUBSYSTEM DEVELOPED BY MIKE COLEMAN, AND THEN ADAPTED TO WORK WITH THE SCOPE MONITOR.

TO DEBUG A SUBSYSTEM, ONE SHOULD FIRST GET IT WORKING AS COMPLETELY AS POSSIBLE BY RUNS IN LOWER CORE WITH LINEPRINTER OUTPUT. THEN ONE CAN RUN IT IN THE H-MODULE BY RUNNING A WAITING PROGRAM IN LOWER CORE, SO THAT YOU CAN ONLY CLOBBER YOURSELF. THE WAITING PROGRAM IS BEST WRITTEN IN UPDATE AND CAN THEN GIVE A DUMP OF THE H-MODULE AND RELOAD A FRESH COPY OF THE SCOPE MONITOR AT THE TERMINATION OF THE RUN. IN THIS WAY, ONE CAN DEBUG A SYSTEM IN 3 MINUTE BURSTS WITHOUT ENDANGERING INNOCENT USERS AND WITHOUT BOTHERING THE OPERATORS TO DO DUMPS. WHEN THE USER SUBSYSTEM IS SUPPOSEDLY DEBUGGED, IT CAN BE RUN ANY TIME WITH NORMAL USER PROGRAMS IN LOWER CORE, BUT IT MUST FIRST PASS AN ACCEPTANCE TEST. THE ACCEPTANCE TEST PROGRAM CAN BE OBTAINED FROM A. H. BOND.

10. GRASP

'GRASP' IS A GRAPHICAL SYSTEM,AKIN TO 'SKETCHPAD', DEVELOPED BY GENE THOMAS ON THE G-21, AND DESCRIBED BY HIM AT THE ACM CONFERENCE 1967. IT WORKS WITH OUR SCOPES AND IS WRITTEN IN ALGOL 20, SO IT SHOULD BE EASILY TRANSFERABLE TO THE 360. IT IS CURRENTLY BEING MAINTAINED AND EXTENDED BY RON BUSHYAGER.

GRASP (GRAPHIC SERVICE PROGRAM) IS A GENERAL GRAPHIC MODEL BUILDING SYSTEM. IT IS USED IN TWO WAYS:

1. IT PROCESSES AN INPUT STREAM OF CARDS IN A SIMPLE LANGUAGE,WHOSE FORMAT IS SIMILAR TO A SEQUENCE OF ALGOL PROCEDURE CALLS. THIS ALLOWS THE USER TO DEFINE AND NAME GRAPHICAL ELEMENTS LIKE POINTS,LINES,ETC.,BUILD NAMED CONFIGURATIONS FROM THESE ELEMENTS AND DUPLICATE INSTANCES OF THESE CONFIGURATIONS AT DIFFERENT LOCATIONS AND ORIENTATIONS IN THE [3] MODEL SPACE. THE STRUCTURE OF THE MODEL IS NESTED, SO ONE HAS CONFIGURATIONS AT VARIOUS LEVELS. ONE CAN INPUT CARDS FROM THE NORMAL INPUT STREAM OR FROM THE SCOPE FACE AND ONE CAN OUTPUT A [2] DISPLAY DERIVED FROM THE MODEL,ON THE LINE-PRINTER OR THE SCOPE FACE. THE [2] DISPLAY IS COMPLETELY SPECIFIED BY THE USER AS TO ITS SCALE, REGION OF INTERSET WITHIN THE MODEL, VIEWPOINT IN THE MODEL SPACE AND ORTHOGRAPHIC OR STEREOGRAPHIC PROJECTION. THE MODEL CAN ALSO BE CHANGED IN VARIOUS SIMPLE WAYS - PARTS OF IT CAN BE ROTATED, MOVED OR DELETED , AND THE MODEL CAN BE SAVED ON AN AUXILIARY AND FILE.

2. THE USER CAN USE PART OF THE GRASP SYSTEM AS AN OUTER BLOCK TO AN ALGOL PROGRAM, WHICH CONTAINS PROCEDURE CALLS TO GRASP PROCEDURES, SIMILAR TO THE LANGUAGE, BUT EMBEDDED IN ANY ALGOL CONSTRUCTIONS. THIS PROGRAM WOULD CONSTITUTE AN 'APPLICATION PROGRAM' AND, IN THE GRASP LANGUAGE, ONE CAN INSTRUCT THE APPLICATION PROGRAM TO BE CALLED FROM ITS AND FILE AND APPLIED TO THE MODEL.

GRASP DOES NOT HAVE CONSTRAINT SATISFACTION FEATURES BUILT IN. THERE IS QUITE A GOOD AND COMPREHENSIVE USER MANUAL AVAILABLE.

11. HOW THE SCOPE MONITOR WORKS.

A. RELATIONSHIP OF THE SCOPES TO THE G-21 AND THE MAIN MONITOR

THIS SECTION CAN BE SKIPPED.

THE G-21 HAS SEVERAL 8K MEMORY MODULES ON A BUSS, AND, IN ADDITION, THE H-MODULE, WHICH CAN BE SWITCHED IN AS REQUIRED TO REPLACE THE G-MODULE. THE CORE LOCATIONS OF THE H-MODULE ARE /160,000 TO /177,777, AND A PROGRAM WILL COMMUNICATE WITH THIS CORE WHEN THE H-MODULE IS SWITCHED IN; OTHERWISE, IT WILL COMMUNICATE WITH THE G-MODULE. SWITCHING IS DONE BY SETTING $13 IN THE CE REGISTER. IT CAN ONLY BE SWITCHED IF THE OPERATOR HAS SET THE MODULE SWITCHES. TO SEE IF IT IS SWITCHABLE WE MUST READ THE STATUS REGISTER SR (REGISTER 5) AND LOOK AT $4. THE NORMAL PE IMAGE PROTECTS THE H-MODULE, AND SO WE CAN RESET THAT TO /70 OR /13 ACCORDING TO WHETHER THE CORE IS INVERTED OR NOT. THE CORE IS INVERTED IF THE (ABC) BUTTON HAS BEEN SET BY THE OPERATOR, AND THIS CAN BE READ BY LOOKING AT $1 OF SR, IF $1 IS SET, THE USE /70. THE MAIN MONITOR IS CONTINUALLY SERVICING INTERRUPTS FROM TELETYPES, ETC., AND WHEN IT DOES SO IT STACKS THE ACCUMULATOR AND THE NC REGISTER ONLY. THE MAIN MONITOR INTERRUPT STACK IS 4 DEEP AND CIRCULAR. WHEN IT RESTORES CONTROL TO THE PROGRAM, IT RESTORES THE ACCUMULATOR AND TRANSFERS TO (NC) AND IT RESETS PE AND CE TO A STANDARD PATTERN, NOT TO THE PATTERNS IN OPERATION WHEN THE INTERRUPT OCCURRED. IT RESETS THEM FROM THE PE AND CE IMAGES, WHICH ARE (169+1) AND (133+5), RESPECTIVELY. HENCE, WE MUST EITHER TURN CONTROL OFF OR RESET THESE EVERY TIME AFTER CONTROL HAS BEEN ON. THE SCOPE INTERRUPT BUTTONS SET $13 IN IR AND THE MAIN MONITOR SENDS CONTROL TO THE SCOPE MONITOR. THE SCOPE INTERRUPT BUTTONS, AT THE SAME TIME, SET THE INTERRUPT WORDS IN THE H-MODULE, AND THE SCOPE MONITOR READS THESE.

FIGURE 8 SHOWS CODE TO SWITCH IN AND OUT THE H-MODULE. THE CLOCK INTERRUPT TO THE SCOPE MONITOR CAN BE EASILY PATCHED AND FOR SPECIAL EFFECT. THE SCOPE MONITOR CANNOT USE ANY OF THE USUAL FACILITIES OF THE MAIN MONITOR, LIKE I ROUTINES, AS THESE MAY BE IN USE BY THE LOWER CORE PROGRAM. COMMUNICATION WITH THE DISC IS EFFECTED BY USING THE TELETYPE DISC ROUTINE, AND THE SCOPE MONITOR ONLY ENTERS THEM IF THEY ARE FREE AND WAITS OTHERWISE. THE SCOPE FILES ARE IN A SPECIALLY RESERVED PORTION OF DISC, CONSTITUTING RA TYPE 25. THE BLOCKS ARE OF LENGTH 160₈, HALF THE USUAL BLOCK LENGTH. DISC SPACE IS HANDLED IN GLOBS ON AN AVAILABLE SPACE LIST BY THE SCOPE MONITOR.

TO SWITCH IN THE MM-12

```
         ERA  0    ,SR;
         IEZ  0   $4;
         TRA      L1;
         EXR  0   /77776,CE;
         ERA  0   /77777,CE;
         UNL  0   $13;
         OAD  0   0;
         LDR  0    ,CE;
         ERA  0    ,SR;
         IEZ  0   $1;
         OCA  0   /70-/13;
         OCA  0   /13;
         LDR  0    ,PE;
         TRA      L2;
L1       ERROR EXIT          H MODULE NOT SWITCHABLE
L2       NORMAL EXIT         H MODULE SWITCHED IN,CONTROL IS OFF
```

TO RESTORE NORMAL USER SETTINGS

```
         LDR      |69+1,PE;
         LDR      |35+5,CE;
```

FIGURE 8.

B. THE H-MODULE

THE LAYOUT OF THE H-MODULE IS SHOWN IN FIGURE 9; THE ACTUAL PATH OF THE SCANNER IS AS SHOWN IN THE SMALL FIGURE. IN ORDER NOT TO UPSET THE SCANNER, IT IS DIVERTED MOMENTARILY TO A SMALL LOOP, LOCATIONS 0 AND 1 OF THE H-MODULE, ON ANY REARRANGEMENT OF THE DISPLAY AREA.

THE SYSTEM MESSAGES SIT IN THE H-MODULE AND ARE MADE VISIBLE ON A GIVEN SCOPE BY SETTING THE LOWER BITS OF THE DELIMIT. SYSTEM MESSAGES DISPLAY IN ALTERNATE MODE AND ON ALL PAGES.

C. PROCESSING OF INTERRUPTS, WAITING, REENTRANT CODE

THE SCOPE MONITOR IS LAID OUT AS AN INTERRUPT CLASSIFICATION PART AND THEN TABLES OF ENTRY POINTS FOR THE MEANINGS OF INTERRUPTS IN EACH STATE. TO EXECUTE A TASK THE APPROPRIATE ENTRY POINT IS ENTERED IN THE PART OF THE CODE WHICH IS REENTRANT. WHEN THE CONTROL REACHES A POINT WHERE IT HAS TO WAIT FOR THE DISC OR FOR THE HUMAN TO TYPE IN SOMETHING ON THE SCOPE, IT MERELY SETS UP A REQUEST AND RETURNS TO THE ISR, LEAVING ALL THE LOCAL VARIABLES AND MARKS, FOR THE ROUTINES SO FAR PASSED THROUGH, IN THE STACK (THERE IS ONE STACK FOR EACH SCOPE). WHEN THE OPERATION IS COMPLETE, IT CARRIES ON WHERE IT LEFT OFF.

/160000	USED BY SCANNER		
10	STATE WORD		
11	INTERRUPT WORD	SCOPE 1	NORMAL
12	POSITION WORD		
20	''	SCOPE 2	CONSOLE
30	''	SCOPE 3	GROUPS
110	''		ALTERNATE
120	''		CONSOLE
130	''		GROUPS
150	INTERRUPT ENTRY POINT		
151	TRA ISR		
152	CLOBBER WORD JR01		
154	USER ENTRY POINT		
155	TRA USER INTERFACE		
160	SYMBOL TABLE AND SYSTEM VARIABLES IN FIXED LOCATIONS		
180-1700	TABLES AND ALL DATA USED BY THE SCOPE MONITOR		
1700-3100	SYSTEM DISPLAY PAGES		
3100-4400	ISR		
4400-10000	ROUTINES FOR CARRYING OUT OPERATIONS		
170000-177777	DISPLAY AREA		
177776	DELIMIT TO PROTECT		
177777	CYCLE TO DISPLAY PAGES		

FIGURE 9.

THE COMPLETION OF AN OPERATION IS EITHER TRIGGERED BY AN INTERRUPT LIKE THE COMPARE INTERRUPT ON THE RETURN CHARACTER, OR, IN THE CASE OF DISC TRANSFERS, THE SCOPE MONITOR KEEPS LOOKING TO SEE IF IT CAN COMPLETE THE OPERATION, IN THIS CASE TO ENTER THE MAIN MONITOR DISC ROUTINES.

THIS TIME SHARING, INTERRUPT PROCESSING, MECHANISM WAS DESIGNED AND IMPLEMENTED BY JERRY RIGHTNOUR.

D. INTERACTION WITH THE USER PROGRAM

THE B ROUTINES ARE JUST A PART OF THE SCOPE MONITOR WHICH IS EXECUTED BY THE NEW PROGRAM. FIGURE 10 IS THE INTERFACE ROUTINE. ONE CALLS A B ROUTINE BY PUTTING THE NUMBER OF THE B ROUTINE IN THE ACCUMULATOR AND THE SUCCESSIVE ARGUMENTS IN REGISTERS 52-56 AND DOING A TRM TO 10. THIS BLOCK OF CODE IS INCLUDED IN THE B PROCEDURE IN ALGOL.

USER INTERRUPTS ARE HANDLED DIFFERENTLY FROM INTERNAL INTERRUPTS. THEY ARE CLASSIFIED IN THE ISR, BUT CONTROL IS NOT TRANSFERRED TO THE USER PROGRAM UNTIL AFTER ALL THE SWITCHES AND MAIN MONITOR REGISTERS HAVE BEEN RESTORED JUST BEFORE CONTROL WOULD BE TRANSFERRED BACK TO MAIN MONITOR. AT THIS POINT, THE SCOPE MONITOR EXECUTES ANY USER INTERRUPTS BY TRANSFERRING WITH CONTROL ON TO THE USER ENTRY POINT IN LOWER CORE. ACTUALLY,IT STORES ITS OWN MARK IN THE USER ENTRY POINT AND DOES A TRE TO ENTRY POINT +1. THUS IF THE FIRST INSTRUCTION TURNS CONTROL OFF,ONE CAN MAINTAIN CONTROL OFF IN AN INTERACTIVE PROGRAM.

```
I0      ENT                 USER INTERFACE ROUTINE
        EXR  0  /77776,CE;  CONTROL OFF
        STI     L20;        SAVE PARAMETER
        ERA  0   ,SR;       READ STATUS REGISTER
        IEZ  0  $4;         IS THE MM-12 SWITCHABLE
        TRA     L2;         NO EXIT
        LDR  0  /20302,CE;  SWITCH TO THE MM-12
        CAL     /160152;    GET THE CLOBBER WORD
        IUO     L10;        IS IT INTACT
        TRA     L2;         NO EXIT
        LDR  0   ,PE;       RESET MEMORY PROTECT
        CLA     L20;        REFETCH THE PARAMETER
        TRM     /160154;    ENTER THE SCOPE MONITOR
L0      LDR     |69+1,PE;   RESTORE MEMORY PROTECT
        LDR     |33+5,CE;   RESTORE CE REGISTER
        TRE  1  I0;         EXIT
L2      CLS  0  1;          SET EXIT SWITCH TO ERROR CONDITION
        TRA     L0;         EXIT
L10     ALF     1JR01;      CLOBBER WORD
L20     LWD      ;          TEMP
        LBL     L;
```

FIGURE 10.

E. THE TRANSIENT VERSION

IN THE TRANSIENT VERSION UNDER DEVELOPEMENT,ONLY THE ISR AND TABLES WILL BE RESIDENT,OCCUPYING ABOUT 1500. WORDS. THE OTHER ROUTINES AND THE SYSTEM MESSAGES ARE SWAPPED IN AS REQUIRED AND ALL MODULES,WHETHER THEY BE SYSTEM CODE,USER CODE, SYSTEM MESSAGES OR USER DISPLAYS,ARE TREATED EQUIVALENTLY IN THE SAME AVAILABLE SPACE. THE SCHEDULING IS SUCH THAT MODULES ARE KEPT IN CORE AS LONG AS POSSIBLE,TO MINIMISE UNNECESSARY SWAPPING. THUS , A USE OF SEVERAL RELATED FACILITIES SHOULD INVOLVE NO SWAPPING. IN THIS WAY, FOR LIGHT USE THE TRANSIENT VERSION SHOULD RUN AS FAST AS THE RESIDENT VERSION, AND FOR HEAVY USE, EITHER CODE OR DISPLAY AREA, THE TRANSIENT VERSION WILL BE ABLE TO CARRY OUT OPERATIONS IMPOSSIBLE FOR THE RESIDENT VERSION ,BUT WITH LESS EFFICIENCY AND SLOWER RESPONSE.

BCPL Syntax in Backus Normal Form

```
<cap> ::= A|B|...Z
<small> ::= a|b|...z
<digit> ::= 0|1|...9
<octd> ::= 0|1|...7

<n1> ::= <cap>|<small>|<digit>
<n2> ::= <n1>|<n2><n1>
<name> ::= <small>|<cap><n2>

<string> ::= '...'
<stringconstant> ::= "..."

<ol> ::= <octd>|<ol><octd>
<octn> ::= #8<ol>
<decn> ::= <digit>|<decn><digit>
<number> ::= <decn>|<octn>

<2-op> ::= <-|->|+|-|#+|#-|!
<3-op> ::= *|/|#*|#/|%
<4-op> ::= +|-|#+|#-
<5-op> ::= lshift|rshift
<6-op> ::= =|!=|<|<=|>|>=|!<|!>|#=|#!=|#<|#<=|#>|#>=|#!<|#!>
<7-op> ::= &
<8-op> ::= |
<9-op> ::= eqv|neqv

<primary-E> ::= <string>|<string-constant>|<number>|true|false|(E)|
            valof<block>|<primary-E>(<E-list>)|<primary-E>[<E>]|
            <name>|<string>|<stringconstant>|<number>
<2-E> ::= <primary-E>|<2-op><2-E>
<3-E> ::= <2-E>|<2-E><3-op><3-E>
<4-E> ::= <3-E>|<3-E><4-op><4-E>
<5-E> ::= <4-E>|<4-E><5-op><5-E>
<6-E> ::= <5-E>|<5-E><6-op><6-E>
<7-E> ::= <6-E>|<6-E><7-op><7-E>
<8-E> ::= <7-E>|<7-E><8-op><8-E>
<9-E> ::= <8-E>|<8-E><9-op><9-E>
<E> ::= <9-E>|<9-E>-*<E>,<E>

<Ell> ::= <E>|<Ell>,<E>
<E-list> ::= <null>|<Ell>

<nll> ::= <name>|<nll>,<name>
<D1> ::= <nll>=<Ell>
<D2> ::= <name>(<namelist>)be<block>
<D3> ::= <name>(<namelist>)=<E>
<D4> ::= <name>=vec<constant>
<manifest1> ::= <name>=<constant>
<manifest2> ::= <manifest1>|<manifest1>;<manifest2>
<manifest> ::= <null>|<manifest2>
<global1> ::= <name>:<constant>
<global2> ::= <global1>|<global1>;<global2>
```

```
<global> ::= <null>|<global2>
<D5> ::= <D1>|<D2>|<D3>|<D4>|<manifest>|<global>
<D> ::= <D5>|<D5>also<D>

<C1> ::= <E11>:=<E11>
<C2> ::= <F>(<E-list>)
<C3> ::= goto<E>
<C3> ::= break
<C4> ::= return
<C5> ::= finish
<C6> ::= resultis<E>
<C7> ::= switchon<E>into<block>
<C8> ::= <block>
<C9> ::= <C1>|<C2>|<C3>|<C4>|<C5>|<C6>|<C7>|<C8>
<C10> ::= if<E>then<C>
<C11> ::= test<E>then<C>else<C>
<C12> ::= unless<E>do<C>
<C13> ::= while<E>do<C>
<C14> ::= until<E>do<C>
<C15> ::= for<name>=<E>to<E>do<C>
<C16> ::= <C9>repeat
<C17> ::= <C9>repeatwhile<E>
<C18> ::= <C9>repeatuntil<E>
<C19> ::= <C9>|<C10>|<C11>|<C12>|<C13>|<C14>|<C15>|
          <C16>|<C17>|<C18>
<L1> ::= <name>:
<L2> ::= case<constant>:
<L3> ::= default:
<L> ::= <L1>|<L2>|<L3>
<C> ::= <C19>|<L><C>
<Clist> ::= <null>|;<C><Clist>
<Dlist> ::= <null>|;<D><Dlist>
<body> ::= <D><Dlist><Clist>|<C><Clist>
<block> ::= $(<body>$)
```