I.C.T. ATLAS COMPUTER

SUPERVISOR AND FIXED STORE ROUTINE SPECIFICATIONS

VOLUME 2

ROUTINE 400-599

Both the Atlas computer and the routines described in this document are the result of a collaboration between Manchester University and I.C.T. Limited (formerly Ferranti Limited, Computer Department).

This document is confidential and may not be given or lent to any person other than addressee without the authority of I.C.T. Limited.

ATLAS SUPERVISOR AND FIXED STORE ROUTINES

This document contains annotated programs of the Atlas Fixed Store and Supervisory Routines.

It should be used in conjunction with the Volume which contains an overall description of the routines and their relationships to each other and also the volume containing the routine specifications.

As all the programs are not yet available this document will be corrected and supplemented from time to time.

2.9.63.

SECTION 4 MAGNETIC TAPE ROUTINES

MAGNETIC TAPE ROUTINES

SECTION 5 PERIPHERAL ROUTINES

SECTION 6 THE OPERATING SYSTEM

PERIPHERAL ROUTINES

CONTENTS/3 (VOL. 2)

SECTION 7 MONITOR ROUTINES

SECTION 9 EXTRACODES 1200-1777

1. 9.64

CONTENTS / 4

1.10.63

MAGNETIC TAPE ROUTINES - ABBREVIATIONS

PBA	Present Block Address, read from the magnetic tape V store
EBA	Expected Block Address, used to check the PBA
WBA	Wanted Block Address, used during search orders
SBA	Stop Block Address, used to check if the tape has stopped
LBA	Leading Block Address (always non zero)
TBA	Trailing Block Address (always zero)
CU	Current Order, or order being processed
FO	Following Order, or order waiting to be processed when the CU is completed and the store is ready
DD	Deck Directory, a flip flop register
ED	Error Directory, a flip flop register for tape errors and faults
TCR	Tape Control Register
TQ	Tape Queue
Qm	Location of most recent TQ entry for this channel. m is contained in the DD
Qp	Location next TQ entry to be processed for this channel. p is contained in the CU.

1/10/63

R400: Tape Flip Flops in Subsidiary Store

The following two flip flops are contained in the CU locations:

CU not complete flip flop.

Set when a tape order is placed in a CU location.
Reset when the order has been successfully completed.

Supervisor exit flip flop

Set by the main supervisor.
Reset on successful completion of the order, on entry to R217.

The following two flip flops, in the DD location, are for the purpose of ensuring that the total number of possible long interrupts, which may have to be entered in an SER queue is not more than two entries per channel:

Clear last flip flop

Set when a tape order has been successfully completed, but the tape has not been stopped. Reset by the clear last long interrupt routine.

Prepare next flip flop

Set, if the clear last flip flop is set, when testing whether to enter the prepare next routine. Reset by the clear last long interrupt routine.

The following two flip flops in the DD locations are to test either whether to initiate the next tape order and start the tape after coming back from the Organise store subroutine R414, or whether to initiate the next tape order as soon as the current order has been successfully completed, without stopping the tape.

Tape in use flip flop

Set when the tape is started moving.
Reset when the tape has stopped, after successfully completing a tape order.

Store ready flip flop

Set when the store is ready for the next tape order.
Reset when the next tape order is initiated.

R400: continued

The following two flip flops are for the purpose of ensuring that if a LBA, forwards, or a TBA, reverse, is encountered, then another BA interrupt should occur within 0.1 sec:-

F1: Set at a LBA, forwards, or TBA, reverse
Reset at a TBA, forwards, or LBA, reverse

F2: Set by the clock interrupt, if F1 is set
Reset at a TBA, forwards, or LBA, reverse

Note that if F1 and F2 are both set when a clock interrupt occurs then the transfer is ended and control of the tape passes to the monitor. This, however, does not apply to Orion tape.

Location F3

This location is used to record the clock reading of the time when either a start tape order or a stop tape order is written to the TCR. It is reset as soon as a BA interrupt occurs. If the clock interrupt routine finds that F3 has not been reset after a certain interval of time, it is assumed either that the tape has failed to respond to start or stop order, or that BA interrupts are failing to register.

Tape Counters in Subsidiary Store

CE: This counter is used after slowing down a long search from fast speed to normal speed. It counts consequentively sequenced LBAs, in order to determine when normal BA interrupts are assumed to occur. The counter is also used when aligning the tape to the EBA. It counts the number of changes of direction of tape movement required to align the tape to EBA = PBA.

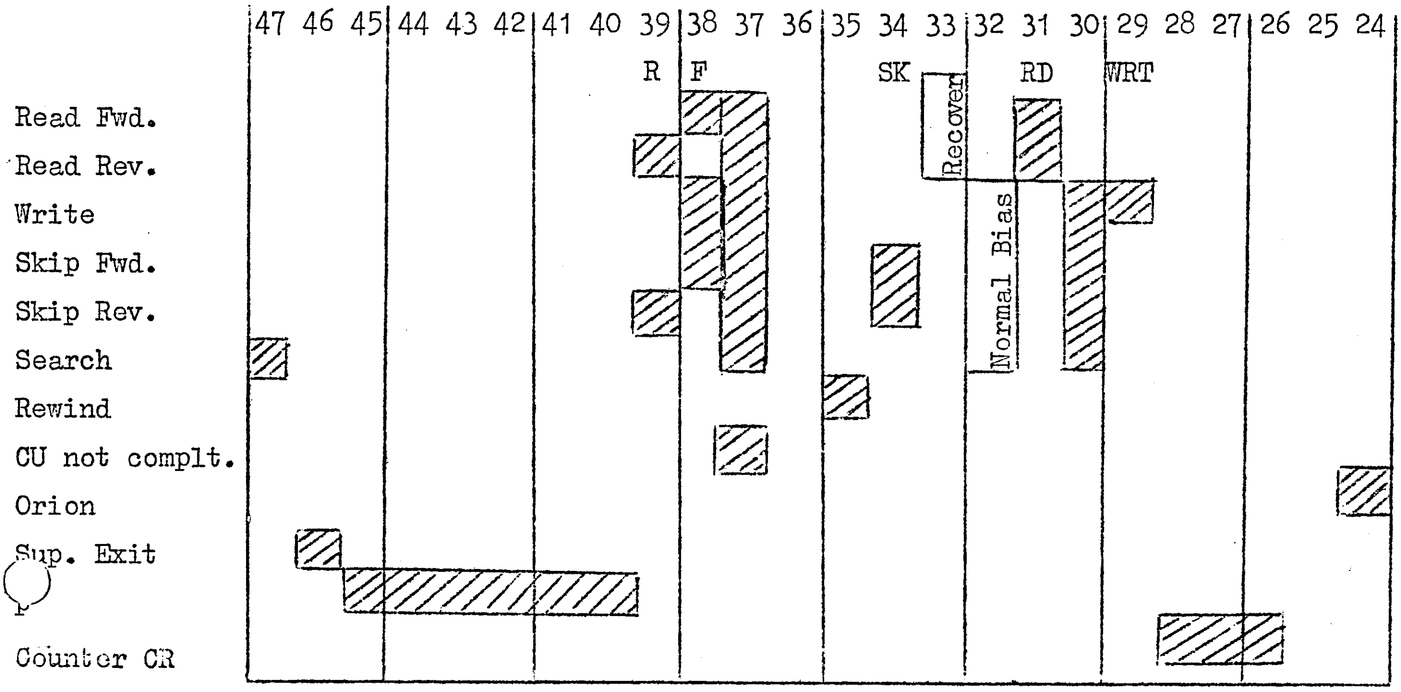
CT: Number of seconds on a fast speed search before it has to be slowed down to normal speed.

CQ: Number of blank spaces in the tape queue. Note that this counter must be preset, probably at 15.4. Otherwise, no tape order can be processed.

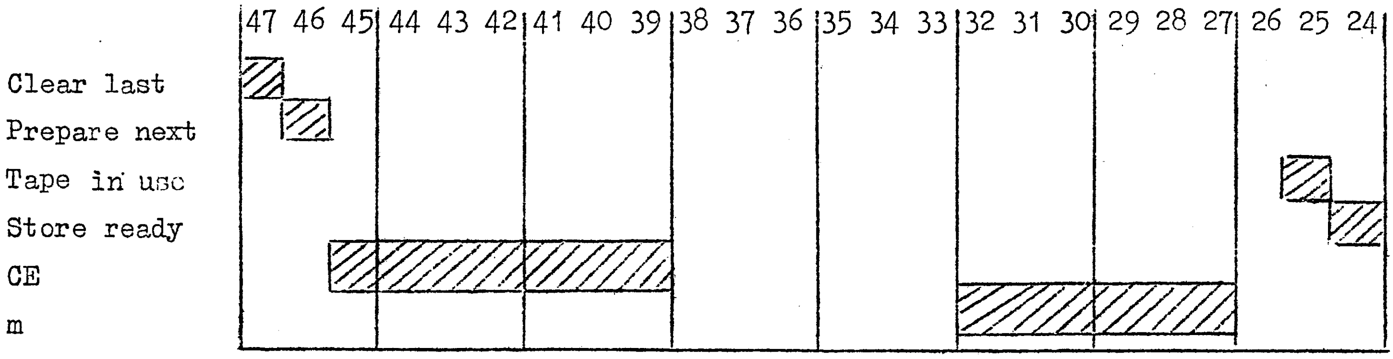
CR: This counter is contained in the CU location. It counts the number of times the current order has been repeated, due to error conditions.

R400: Digits in Tape Working Store (continued)

Tape Order (Qp, FO, CU)



Deck Directory (DD)



R400: Digits in Tape Working Store (continued)

Error Directory (ED)

	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write Crisis								▨												▨				
Read Crisis								▨										▨						
Parity 3								▨								▨								
Parity 6														▨										
Checksum																								▨
512 Words																							▨	
Write not reset																			▨					
Read not reset																	▨							
LBA Error						▨																		
TBA Error					▨																			
Beg. Tape				▨																				
End. Tape			▨																					
Dir. fault									▨															
Spd. fault							▨																	
Read Bias fault															▨									
Deck Fail Reset																					▨			
Non Equivalence													▨											
Counters										▨	▨	▨												

R401: B.A. Interrupt Entry

Whenever a B.A. interrupt occurs from magnetic tape this routine is entered. It stores the following information from the subsidiary store and the tape V-store into B registers: PBA, EBA, TCR, CU, DD. It also resets location F3 and the BA interrupt LAM. The routine then transfers control to an interrupt routine, which has been preselected by the supervisor or monitor programs. The main B.A. interrupt routines include the following:

1. R402 B.A. interrupt routine
2. R403 Tape stopped interrupt routine
3. R404 Alignment of tape to EBA
4. R405 Calculation of EBA after an off channel search
5. R500 (Exit) if BA interrupts are to be ignored

Registers of fixed store	10
Instructions obeyed	10
B registers used	111-116, 123

R402: B.A. Interrupt Routine

The B.A. Interrupt routine deals with the B.A. interruptsas they occur from a magnetic tape, which is moving under normal operation. It is entered from the B.A. interrupt entry routine R401 at one of two entry locations. One entry corresponds to the LBA, and the other entry corresponds to the TBA

The routine accomplishes the following:

1. Checks the PBA = EBA. If there is any discrepancy, the routine records an error in the error directory, and stops the tape.

2. At a LBA, forwards, or TBA, reverse, it sets flip flop bit F1 to indicate that another BA interrupt is expected within 0.1 sec. At a TBA, forwards or LBA, reverse, it resets flip flop bit F1. It also resets flip flop bit F2, which has been set by the clock interrupt. Note that if both F1 and F2 are set when a clock interrupt occurs, it indicates that a BA interrupt has failed to register. If this should occur, the transfer is ended and control of the tape is passed to the monitor. This, however, does not apply to Orion tape.

3. At a LBA, forwards, or TBA, reverse, the CU is tested. If the CU is a write order, then the "Write" digit is set in the TCR. If the CU is a read order, the TCR is tested to ensure that the "Start Read at next BA" digit is set. If it is set, then the "End Read at Next BA" digit is written to the TCR, otherwise, an error is recorded in the error directory and the transfer is ended, but the tape proceeds to the next B.A. interrupt before further action is taken. If the CU is a search, the WBA is compared with the EBA, and action to proceed, or stop the tape is taken accordingly.

4. At a TBA forwards, or LBA, reverse, the TCR is checked for certain error conditions, such as check sum error, not 512 words in a block, write digit not reset, according to the order that has just been completed. Also the parity 3 and parity 6 errors are examined. Any errors which are indicated are recorded in the error directory, and the tape is stopped.

5. At a TBA, forwards, or LBA, reverse, if the tape order just completed was a read or write, but not on Orion tape read order, and no errors were found, then the corresponding entry in the PAR is altered to prevent it from interfering with any following tape transfer order.

6. At TBA, forwards, or LBA reverse, if the next tape order is ready then it is moved into position to be processed, and the corresponding entry in the PAR is set to enable the page to accept the transfer. Hence, successive tape orders can be processed without stopping the tape. This, however, is not done when the tape either is in monitor control, or is an Orion tape.

R402: continued

7. At a TBA, forwards, or LBA, reverse, either the routine exits to the clear last long interrupt, R412, if the previous tape order was a read or write order and the tape is proceeding to the next tape order without stopping, or the routine exits to main control.

8. At a LBA, forwards, or TBA reverse, either the routine exits to the prepare next long interrupt, R411 if there is space in the SER tape queue, or the routine exits to main control, setting a flip flop bit in the DD to indicate that as soon as there is space in the SER tape queue, the prepare next long interrupt can be entered.

9. If the tape has to be stopped, then the routine writes the clock reading into F3 location, to indicate that a stop B.A. interrupt is expected within a certain period of time.

Entry		R401	BA Interrupt entry
Exit	1.	R411	Prepare next long interrupt
	2.	R412	Clear last long interrupt
	3.		Main control

Fixed store registers 124

Instructions obeyed varies from 15 to 50 in normal operation

B Registers used 111-117

R403: Tape Stopped Interrupt Routine

The tape stopped interrupt routine is entered from the BA interrupt entry R401 at the next BA interrupt after a stop order has been given to the tape. To determine whether the tape has stopped it compares the PBA with the value stored from the previous BA interrupt. If there is no change, it is assumed that the tape has stopped.

If it is assumed that the tape has stopped, the routine then enters the long interrupt start tape routine R413. This does not apply, however, if the tape is endeavouring to align itself to stop before EBA=PBA. In this case, the tape is either restarted again moving in the opposite direction, or the monitor routine is entered, depending on whether the tape has been restarted less than or more than five times, respectively.

If it is indicated that the tape has not stopped, the routine checks to determine if the TCR is set to stop, and informs the operator accordingly. BA interrupts, which may occur after the tape has indicated that it has stopped, are ignored.

Entry:		R401	BA Interrupt entry
Exit:	1.	R419	Start tape long interrupt
	2.	-	Failed to find EBA after 5 attempts
	3.	-	Failed to stop
	4.	-	TCR not set to stop
	5.		Main control

Number of fixed store registers	30
Maximum number of instruction obeyed	16
B register used	111-116, 123

R404: Alignment of Tape to E.B.A.

This B.A. interrupt routine is used in conjunction with the tape stopped interrupt routine by the main store monitor program when it is desired to align the tape to stop just before PBA = EBA, where the EBA remains unchanged.

Successive PBA's are checked at each BA interrupt until one is found equal to the EBA, when the tape is moving in reverse. The tape is then stopped. This search for the required PBA is repeated, if necessary, up to five times before indicating that the required PBA cannot be found.

Entry: R401 BA interrupt entry

Exits: 1. (5/402) of BA interrupt routine

2. Main

Number of fixed store registers	17
Maximum number of instructions obeyed	12

B register used 111-116, 123

R405: Calculation of E.B.A.

This BA interrupt routine is used after the tape has been brought back to normal speed from fast speed towards the end of a long off channel search. It determines when BA interrupts occur normally, and calculates the EBA.

The value of the EBA is taken as the value of a certain number of consecutively sequenced, and checked PBA's. After having determined this value for the EBA, then BA interrupts will normally be processed by routine R402.

Entry: R401 BA interrupt entry

Exit Main control

Number of fixed store registers 24

Maximum number of instructions obeyed 10

B registers 111-116, 123

R406: Deck Failure Interrupt Routine

The deck failure interrupt routine identifies whether the deck failure was caused by a read or write crisis, or by the tape being stopped on the metal backing, or by a mechanism failure. It then deals with the interrupt as follows:

If the interrupt was caused by a read or write crisis, the routine exits to (1/407). Otherwise it writes "End Transfer" to the TCR of the effected channel, resets the timers and tries to reset the interrupt.

If the interrupt cannot be reset, the tape is stopped and disengaged, and the operator is informed. There should be no further BA interrupts, but if any occur, they will be ignored.

If the interrupt has been reset, and it is found that the metal backing digit is set in the TCR, then the routine exits to (1/416), the metal backing long interrupt routine.

If the interrupt has been reset, and the tape is not at the metal backing, then the tape will not be stopped. An error mark will be written to the ED of the channel concerned, and the tape continues moving until the next BA interrupt occurs,before further action is taken.

Entry: from deck failure interrupt

Exits: 1. 1/407 Read and Write Crisis

2. 1/416 Metal backing long interrupt

3. R419 Deck failure cannot be reset

4. Main

Number of registers of fixed store 34

B registers used 111-113, and 123

R407: Parity 3 & 6 Read & Write Crisis Time Interrupt

Parity 3 interrupt occurs if an incorrect parity is detected in the core store page in the directory, when reading from tape, or writing to tape. There is no indication given as to which channel was affected.

The routine, therefore, has to examine the TCRs of all channels to determine which are processing read or write orders. If a channel is processing a read or write order, the transfer is ended, and an indication is made in its respective ED that there has been a parity 3 fault. If a channel is not processing a read or write order, and indication is made to its respective ED to stop the tape from processing further tape orders for the duration of the parity 3 monitor program, which will be called in after all the tapes have been stopped.

The read and write crisis interrupts occur if the word transfer between the tape co-ordinator and the central computer is not met within the crisis time of approximately 13 microseconds. This causes a deck failure interrupt, which transfers control to this routine. It is treated in the same manner as a parity 3 interrupt.

Parity 6 interrupt occurs if an incorrect parity is detected in the tape co-ordinator when writing to tape. The buffer parity fault digit is set in the TCR of the effected channel. The routine examines the TCRs of all channels, and if any channel has the buffer parity fault digit set, its transfer is ended, and an indication is made in its respective ED that there has been a parity 6 fault. The other channels remain uneffected.

The parity 3 and parity 6 interrupt routines are combined into one routine, but they have different entry points as indicated below. The routine ends by resetting the interrupt in the V store and returning control to main. The BA. interrupt routine will then identify and deal with the fault accordingly.

Entry: 1. Parity 3 interrupts

2. Parity 6 interrupts

3. Deck failure interrupt

Exit: Main

Registers of fixed store	30
Instruction obeyed	76
B registers	111-116, 123

R411: Prepare Next Tape Order

The prep-re next tape order routine is either entered as a long interrupt from the first BA interrupt after initiating a tape order, or it is entered as an extension of the basic order to tape queue routine, R421. Its purpose is to determine if the next tape order for the channel concerned can be updated and then initiated without stopping the tape on completion of the current tape order. This can be done if the tape order does not involve a change of direction, nor is it a search, nor a rewind, nor an Orion tape order, nor a supervisor order. If none of these conditions apply then the order is moved into the FO position and the organise store subroutine, R414, if required, prepares the corresponding page of core store to accept the transfer. If the order is a composite order; that is, more than one block is involved, then only one part of it is moved to the FO position. When the next BA interrupt occurs, if the organise store routine has prepared the corresponding page of core store and if the current tape order has been successfully completed, then the tape order in the FO location can be initiated by the BA interrupt routine without stopping the tape.

Parts of this routine are common with routines R412 and R413

Entry:	R402	BA interrupt routine
	R421	Basic order to tape queue
	R412	Clear last tape order
	R413	Start tape routine
Exit:	R202	Main
Subroutine Used:	R214	Free program
	R414	Organise store blocks

Registers of fixed store: 72

B registers used 100-110, Bt

R412: Clear last tape order

The clear last routine is entered as a long interrupt from the BA interrupt routine after the completion of a read or write order, if the next tape order has been initiated without stopping the tape. The purpose is to remove the lockout and lockdown digits from the page involved during the previous tape transfer.

Entry:	R402	BA interrupt routine
Exit:	R411	Prepare next tape order
	R202	Main

Subroutines Used: R205 unlock store block

Registers of fixed store	14
B registers used	100-110

R413: Start Tape Routine

The start tape routine, R413, is a preselected routine entered either as a long interrupt from the tape stopped interrupt routine, R403, or as an extension of the placing of a tape extracode into the tapes queue routine, R421. There are three parts to this routine: the clearing up of the previous tape order, the updating of the next tape order, and the initiation of a tape order by writing to the TCR.

The part concerned with the clearing up of the previous tape order begins by testing the ED and if there are any errors recorded in it, the routine exits to R419.

Next, the current order just completed is examined, and if it has not been marked as having been completed successfully, then it will be initiated again.

If it has been marked as complete, and it was a read or write order, then the lockout and lockdown digits from the PAR of the pages involved in the transfer will be removed before proceeding. Also, if the order was marked as a supervisor order, then subroutine R217 will be entered before proceeding to update the next tape order.

The updating part of this routine begins by examining if there is a tape order waiting to be processed in the FO location. If there is, then the order is either moved to the CU location and initiated, if the core store pages concerned are ready for the transfer, or the routine will exit to main control, and the order will be initiated when the store is ready.

If there is no tape order waiting to be initiated, the next order in the tape queue for the channel concerned is extracted and either moved to the FO location, if it is a read or write order, where it will wait until the corresponding pages of core store are prepared, or moved directly to the CU location, if it is a search or skip or rewind order, and initiated.

If there are no further entries in the tape queue for the channel concerned, the routine exits to main control.

If the tape queue should be marked as full, and the updated entry has left a space in the queue, then the tape queue full indication will be removed, and those programs which have been held up because of this will be freed.

The initiating part of the routine writes the necessary digits to the TCR, corresponding to the CU, and then reads them back again to ensure that they have been written correctly, before starting the tape. Also on starting the tape, the clock reading is written to location F3.

R413: continued

Except in the case of search and rewind orders, all tape orders are initiated by extracting the corresponding digits from the CU location and writing them directly to the TCR.

In the case of a search order, the direction in which the tape has to move is determined from a comparison of the WBA with the EBA, and its initial speed of motion is determined from the distance between the WBA and the EBA. If this distance is greater than 200 blocks, the tape is started at fast speed, if not, it is started at normal speed. The length of time, which a tape is to remain on fast speed during a search is calculated, and stored to the nearest second in counter CT, allowing one second for every 23 blocks beyond 200 blocks.

In the case of a rewind order, the following digits are written directly to the TCR: end transfer, end read, normal read bias, fast speed, start, reverse, disengage.

Entry:	R403	Tape stopped interrupt
	R421	Basic instruction to tape queue
Exit:	Main control	
	Three times repeat monitor R419	
	TCR read back incorrect before start, machine monitor	
	Write permit not present on a write order, program monitor	
Subroutine Used:	R217	Enter SEC
	R214	Free Program
	R205	Unlock store block
	R411	Prepare next tape order

Registers of fixed store 118

B register used: 100-110

R414: Organise store blocks for tape

Purpose: An SER to call blocks to core store for transfers to or from magnetic tapes and to set the page address registers for a tape transfer.

Register of fixed store: 62

Instructions obeyed: Supervisor block : 33 + entry to (1/318) + entry to R312
Program block : 45-56 + entry to (2/318) + entry to R312

Multiple program blocks : Around n times one program block for n blocks

Parameters used: (1) to (19)

Cross references:

(2)	=	(5/201)	SER re-entry
(3)	=	(7/201)	SER base
(4)	=	(52/400)	Following tape order (type spec)
(5)	=	0.4(52/400)	Following tape order (store spec)
(6)	=	(53/400)	Deck directory
(7)	=	(35/203)	Block location table
(8)	=	(2/203)	Block directory
(9)	=	(1/318)	Call to cores
(10)	=	(2/318)	Call to cores
(11)	=	(36/314)	Find directory entry
(12)	=	(32/314)	Lose sector
(13)	=	(1/312)	Set PAR
(14)	=	(1/218)	Step directory reference
(19)	=	(15/411)	Return address

Connections with other routines

Entered at (1) from tape routine with B100 = channel no. (digits 5-3) rest zero
Exit to (19/414) with B100 = Channel no.
Block (s) locked down in core store. Page address registers set for channel, forward or reverse, if tape stopped.
Otherwise, digit 14 inverted.
Page number of last block transferred in digits 23-3 of (52/400), digits 2-0 unaltered

Subroutines:

a) "Call to cores" (i) Entered at (1/318) if supervisor block with
B109 = block label (22-12) digits 23, 10, 9, 0 = 1
(lock down, operand, no timer)
B110 = return address.

(ii) Entered at (2/318) if block of object program with
B108 = Directory location relative to start of program, p 12-2. Program number p20-14
Digits 22,0 = 1 (lock down operand)
Rest zero
B110 = Return address
Block location table = BD position relative to start of BD

Exit to resume at re-entry address with B100 preserved if block on drum or drum queue full
Exit to return address when in core store with B109 = Page no., B100-104 preserved.

b) Find directory entry:
Entered at (36/314) with
B102 = Prog. no. (20-14); B.D. entry relative to program start (12-2)
B110 = Return address
Exit to return address with B107 = Program number (8-2)
B108 = Directory entry relative to start of BD

c) Lose sector:
Entered at (32/314) with
B100 = sector number, digits 11-1
B110 = return address
Exit to return address with B109, B101-104 unaltered

d) Set page address register:
Entered at (1/312) with
B108 = New contents of PAR, digits 23-12
B109 = Page no. digits 10-3
B110 = Return address
Exit to return address with B109 unaltered

e) Step directory reference:
Entered at (1/218) to step back at 1.1 (1/218) to step forwards with B107 = Prog. no. (20-14)
BD entry relative to start of program area (12-2)
Remainder irrelevant
B110 = Return address
Return via R203 with
B108 = Prog. no. (20-14)
BD entry of next block relative to start of program area (12-2)
p23 = 1
Rest zero
B105-109 altered.

Temporary working space: B101 - B110, B_t

Notes:

1. On entry to this routine, the "following" entry in the tape queue for the channel holds either.

   b (p22-12), rest zero if a supervisor block. Otherwise program no. (p21-15). Block directory location relative to start of area for program (p13-3), No. of blocks less 1 (p2-0) (always zero for Atlas) p23 = 1.

   On exit the contents are changed to page number (p23-3), p2-0 unaltered. No other tape directory is altered by this routine.

2. If the block is required for a read transfer from Atlas tape and is on the drum, it is "lost" before being called to core store, thus avoiding one drum transfer. This does not apply for an Orion transfer, which may not use the whole block.

3. If the block is a supervisor block, the "dont change timer" digit is set in the Page Directory, irrespective of the block label.

4. The P.A.R. are set as follows for channel n

Atlas	tape,	forward,	tape stopped	:	*77n0	(Orion *77n1 etc)
"	"	"	tape moving	:	*77n4	(" *77n5 *77n6)
"	"	backward,	tape stopped	:	*77n7	(" *77n6 *77n5)
"	"	"	tape moving	:	*77n3	(" *77n2 *77n1)

   P.A.R. contents are also set to these values.
   Digit 1 of the deck directory reads 0 if the tape is stopped, 1 if moving.

5. The current SER base is set to 0.1 by this routine, and the re-entry address is altered.

R416: Metal Backing Routine

The metal backing routine is a long interrupt routine entered from the deck failure routine, R406, if the interrupt was caused by the tape stopping on the metal backing at the beginning or at the end of the tape.

If the tape, which has caused this interrupt is on channel 7 and is being addressed, then the routine exits to the addressing routine.

If the tape is in the process of a search order, and this interrupt occurs, then the deck timer is set for two seconds, and the routine exits to main. The tape will be restarted again moving in the opposite direction, from one to two seconds later, by the one second clock interrupt routine. The first expected BA interrupt will be preselected as a LBA.

If neither of the above cases has caused this routine to be entered, then it is assumed that there has been a mechanism failure, and hence the routine will disengage the tape and inform the operator accordingly.

Entry: Deck failure interrupt routine R406

Exits: 1. Addressing routine

2. Main control

3. Mechanism failure

Registers of store: 15

B registers used: 100-102

R419: Tape Error Repeat Routine

Whenever the start tape routine, R413, is entered, the ED of the channel concerned is examined. If there are any errors or faults recorded in it, control is transferred to the tape error repeat routine. This is a main store monitor routine. It examines the digits, which are set in the ED, one at a time, to identify the type of error or fault which has occured, and then either exits to a monitor routine, or initiates a procedure which will repeat the current order to see if the error can be cleared.

The deck faults which cause this routine to exit directly to the machine monitor R400, include the following:

1. Write digit not reset in the TCR after a write transfer (F7)

2. Read next BA not set at the first BA interrupt at the beginning of a read transfer, or not reset at the end of the transfer (F8)

3. Failure to clear a tape error after several repeats of the current order (F9).

The program faults which cause this routine to exit directly to the program monitor include the following:

1. Beginning of tape

   If the tape has been stopped because the program has referred to block 0, then this routine will space the tape forward to stop between the TBA of block 0, and the LBA of block 1, before entering the program monitor. However, if the supervisor wishes to refer to block 0, it can do so by setting EBA≯1 at this time.

2. End of tape

   If the tape has been stopped because the program has referred to block 5,000 then this routine will space the tape backwards to stop just before the LBA of block 5,000 before entering the program monitor. However, if the CU is a search order, the routine will not enter the monitor, but instead the WBA will be compared with the last EBA, to determine if the block is on the tape. If not it indicates that it is a short tape.

3. Thirteenth bit set in EBA

   This bit may have been set by the supervisor in order to stop the tape at the next LBA. In this case, the EBA will be made equal to the PBA, and the tape restarted.

R419: continued

The deck errors which cause the current order to be repeated include the following, which are tested in this sequence:

1. LBA error

2. TBA error

3. Checksum failure

4. Not 512 words transferred

5. Deck failure has occured and was reset immediately

6. Parity 6

The repeat process is accomplished by altering the EBA so that the tape can be realigned to stop just before the block where the error had occured, by the alignment of tape EBA routine, R404. The ED is cleared, and the tape restarted. When the tape has been realigned, the start tape routine, R413, will automatically re-initiate the order.

A three bit counter in the CU location determines how many times the CU has been repeated. Each time it is repeated, this counter is incremented. If it should reach a certain maximum, say 7 (it can be altered to any number up to 7), then the routine exits to the machine monitor R400, to indicate that the error has not cleared itself.

Deck errors which occur in block 0 are not repeated, but instead the routine exits directly to the monitor.

Other tape errors recorded in the ED, which cause this routine to exit to special routines, include the parity 3 error and the read and write crisis errors.

Entry: R413 Start Tape

Exit: Machine monitor

Program monitor

Main control

Registers of store: 96

B registers used: 100-105, 109

R421: Basic Order to Tape Queue

This routine finds the deck number, allocates and locks out the associated store blocks if necessary, of a basic tape order and then places the order in the tape queue linking it with the previous order for the same channel. If the tape queue should be full when the order is given, then the program affected will be halted.

The basic order to tape queue routine is entered directly from the extracode vector of basic tape orders. There are nine of these orders: search, read forward Atlas tape, read forward Orion tape, read reverse Atlas tape, read reverse Orion tape, write Atlas tape, skip forwards, skip reverse, and rewind. These orders can be simple or composite. A simple order involves only one block from tape. A composite order involves up to eight blocks from tape.

If the order should be a search, a test is made to check if the WBA is on the tape. If not, the program is monitored. Also, if the tape is in variable length mode, the variable length operations are ended before starting the search.

The order is then placed in the tape queue together with its block directory entry or WBA, according to the type of order involved, linked to the previous order for the same channel.

The routine exits either to prepare next long interrupt, or to start tape long interrupt, depending on whether the tape should be marked in the DD as in use or not, respectively.

Entry:	Transfer vector of basic tape extracodes	
Exit:	R411	Prepare next tape order
	R413	Start tape routine
	R213	Tape queue full
Subroutine Used	R221	Find deck number
	R203	Store location and lock out
Register of fixed store	100	
B register used	100-110, Bt, 91, 92, 96, 97	

```
      R450                                              [EXTRACODE AND INTERRUPT ROUTINES

      (0) = *3560

17)   121,    100,    0,      0.7(107)
      113,    100,    0,      0.4(23)                   [Set link when engaged
      121,    108,    0,      4(0)                      [Set link
      101,    110,    0,      (5/203)                   [Programme No.
      113,    110,    0,      1.4(27)                   [Store program No.
      121,    126,    0,      (17/204)                  [Halt main program
      121,    100,    0,      0.1
      113,    100,    0,      (7/201)                   [Set Current SER Base
      121,    100,    0,      4(0)
      121,    101,    0,      (23)                      [Return from print routine
      113,    101,    0,      5.4(9)
      121,    126,    0,      2(62)                     [Print 'engage deck 7'

      *45564741       /       *47450144                 [engage d
      *454353O1       /       *27                       [eck 7

23)   121,    109,    0,      0
      121,    100,    0,      0.2
      114,    100,    0,      7(58/400)                 [Set Deck Allocation Directory
      121,    100,    0,      7
      121,    110,    0,      (1/202)                   [Immediate link to prog scan
      121,    126,    0,      (1/216)                   [Wait for deck engaged
27)   121,    109,    0,      64.3
      121,    108,    0,      0                         [Programme No.
      147,    109,    108,    (9/204)
      113,    109,    108,    (9/204)
      113,    109,    0,      (6/202)
      121,    100,    0,      (40/451)
      113,    100,    108,    20(7/204)
      121,    126,    0,      (1/202)                   [Exit to program scan
25)   121,    100,    0,      2.7(60)                   [Set to print 'modify deck'
      101,    101,    0,      21*6003                   [Pick up TACR
      127,    101,    0,      256
      214,    126,    101,    5(0)                      [Jump if deck not modified
      121,    100,    0,      6.4(60)                   [Set to print 'permit write'
      101,    101,    0,      15*6003                   [Pick up TCR
      127,    101,    0,      1
      215,    126,    101,    (24)                      [Jump if writing permitted
      121,    101,    0,      3(0)
      113,    101,    0,      5.4(9)                    [Print
      121,    126,    0,      2(62)
      121,    100,    0,      *00004
      113,    100,    0,      15*6003                   [Disengage deck
      121,    109,    0,      0.7(25)
      121,    126,    0,      1(23)
```

```
24)     121,    109,    0,      *35612001
        121,    110,    0,      2(0)
        121,    126,    0,      (1/318)         [Lock down interrupt routines
        121,    100,    0,      (1/202)
        113,    100,    0,      5.4(9)          [Set return from print routine
        121,    100,    0,      (3)
        113,    100,    0,      (102)           [Set link to tape SER's
        121,    100,    0,      0
        211,    126,    100,    (4)             [Enter addressing
        165,    101,    100,    0.2
        215,    101,    101,    -2              [Allow for block 0
        122,    101,    0,      50
        121,    100,    0,      0               [Pick up modifier
        121,    126,    0,      (100)           [Enter re-addressing
84)     121,    104,    0,      3(0)
        121,    109,    0,      *3561
        121,    126,    0,      (1/329)         [Remove lockdown
        101,    100,    0,      7.4(24)
        214,    126,    100,    (107)           [Jump if in fault mode
        101,    101,    0,      2(90)           [Pick up last address
        124,    101,    0,      0.1
        121,    106,    0,      2(0)
        121,    126,    0,      3(73)           [Convert to decimal
        113,    103,    0,      2(60)
        121,    100,    0,      4(0)
        113,    100,    0,      5.4(9)
        121,    100,    0,      (60)
        121,    126,    0,      2(62)           [Print no. of blocks
107)    101,    100,    0,      21*6003         [Pick up TACR
        127,    100,    0,      256
        214,    126,    100,    (27)            [Jump if deck not modified
        121,    100,    0,      2.5(60)
        121,    101,    0,      3(0)
        113,    101,    0,      5.4(9)
        121,    126,    0,      2(62)           [Print 'unmodify deck'
        121,    100,    0,      *0000412
        113,    100,    0,      15*6003         [Disengage deck
        121,    109,    0,      0.7(107)
        121,    126,    0,      1(23)           [Wait for engage

88)     *51444556       /       *64514651       [identifi
        *45620101       /       0               [er
        0               /       *01010101
        0               /       0
        *00445163       /       *55576556       [dismoun
        *64016441       /       *6045           [t tape

89)     121,    100,    0,      11.6(60)        [Monitor - 'short blocks'
        121,    126,    0,      1(88)
63)     121,    100,    0,      9.6(60)         [Monitor -'too many faults'
        121,    126,    0,      1.1(88)
```

```
88)     121,    100,    0,      9.6(60)         [Monitor - 'too many faults''
        121,    101,    0,      *0052452        [Set fast wind disengage
        210,    101,    126,    *0000452        [Change to disengage if on leader
        113,    101,    0,      15*6003
        113,    0,      0,      7.4(24)         [Set routine in 'fault mode'
        121,    101,    0,      (84)
        113,    101,    0,      5.4(9)
        121,    126,    0,      2(62)           [Print
4)      121,    100,    0,      0
        113,    100,    0,      0.4(46)         [Store length for pass 1

                                                [START PASS 1
        121,    100,    0,      60
        113,    100,    0,      (90)            [Set timer
        121,    100,    0,      (10)
        113,    100,    0,      (92)            [Set BA link
        121,    100,    0,      50
        113,    100,    0,      511.4(3)
        113,    0,      0,      1.4(90)
        121,    100,    0,      2
        113,    100,    0,      15*6003         [Reset write
        121,    101,    0,      0.1
        113,    101,    0,      0.4(90)         [Set BM count
        121,    100,    0,      (70)
        113,    100,    0,      (91)            [Set end of tape link
        121,    100,    0,      511.4(3)
        122,    100,    0,      (99)
        113,    100,    0,      1(90)           [Set error count
        113,    101,    0,      3*6             [Inhibit Interrupts
        121,    100,    0,      *00002625
        113,    100,    0,      21*6003         [Set TACR
        121,    100,    0,      *0026054
        113,    100,    0,      15*6003         [Start deck
        113,    0,      0,      3*6             [Permit interrupts
        121,    100,    0,      *01210045
62)     113,    100,    0,      15.4(60)
        121,    100,    0,      13.3(60)
        113,    100,    0,      7.4(0)
        113,    126,    0,      (5/201)
        121,    101,    0,      (8)
        121,    110,    0,      2(0)
        121,    126,    0,      (1/220)         [Reserve teleprinter
        113,    126,    0,      (5/201)
        121,    108,    0,      0.1
        121,    109,    0,      0
        121,    110,    0,      2(0)
        121,    126,    0,      (1/240)         [Print text
        113,    126,    0,      (5/201)
        121,    109,    0,      2.1
        121,    110,    0,      2(0)
9)      121,    126,    0,      (3/240)         [New line
        121,    101,    0,      (8)
        121,    110,    0,      2(0)
        121,    126,    0,      (2/220)         [Free teleprinter
        113,    126,    0,      (5/201)
        121,    126,    0,      (1/202)
```

```
                                                              [450/4

                                                              [END OF PASS 1
70)     101,    100,    0,      0.4(90)
        101,    102,    0,      1(90)
        127,    100,    0,      -1
        214,    126,    100,    (71)
        165,    101,    102,    0.4
        214,    126,    101,    4(0)
        122,    102,    0,      0.4                           [Adjust if odd LBM at end
        113,    100,    102,    (99)
        121,    126,    0,      (71)
        114,    100,    102,    (99)
71)     121,    101,    0,      12
        102,    101,    102,    (99)
        217,    126,    101,    5(0)
        120,    101,    0,      12
        124,    102,    0,      0.4                           [Set to erase last few feet
        114,    101,    102,    (99)
        121,    126,    0,      (71)
        165,    103,    102,    0.4
        214,    126,    103,    (72)
        121,    103,    0,      12
        120,    101,    0,      0
        164,    103,    101,    1
        127,    101,    0,      -2
        113,    101,    102,    (99)
        122,    102,    0,      0.4
        113,    103,    102,    (99)
72)     113,    102,    0,      1(90)
        121,    100,    0,      0.4
        121,    103,    101,    0
        101,    101,    100,    3(90)
        165,    104,    101,    *7417036
        163,    104,    0,      0
        163,    104,    0,      0                             |Time in 8- bit chs.
        122,    101,    104,    0
        211,    126,    126,    -1.7(0)
        202,    126,    100,    -7(0)
        122,    103,    101,    0                             [Time elapsed in 8-bit
        165,    101,    103,    *001403
        165,    104,    101,    *000401
        163,    104,    0,      0
        163,    104,    0,      0
        163,    104,    0,      0
        163,    104,    104,    0
        122,    103,    101,    0
        164,    104,    103,    *00177777
        165,    101,    104,    *001774
        122,    104,    101,    0
        163,    101,    0,      0                             [Time elapsed in secs
        163,    101,    0,      0
        124,    104,    101,    0
        124,    101,    101,    0
        125,    104,    0,      0
        163,    104,    101,    0                             |Max. no. of BM's


                                                              [10.6.64
        121,    100,    102,    (99)
        122,    100,    0,      511(3)
        121,    101,    0,      0
        121,    102,    0,      0
        104,    101,    100,    511.4(3)
        104,    102,    100,    511(3)
        201,    126,    100,    -2(0)
        124,    102,    101,    0
        163,    101,    0,      26
        163,    101,    0,      0
        163,    101,    0,      0
        163,    101,    0,      0
        113,    101,    0,      2(90)          [Store last address
        163,    102,    0,      25
        163,    102,    0,      0
        163,    102,    0,      0
        122,    102,    104,    0
        216,    126,    102,    (89)           |Jump if short blocks
78)     121,    100,    0,      (20)
        113,    100,    0,      (92)
        121,    100,    0,      (76)
        113,    100,    0,      (91)
        101,    100,    0,      1(90)
        127,    100,    0,      0.4
        215,    100,    100,    *777777
        124,    100,    0,      *000006
        113,    100,    0,      21*6003
        121,    100,    0,      *0012002
        113,    100,    0,      15*6003        [Start deck
73)     101,    101,    0,      1.4(90)        [No. of bad blocks
        214,    126,    101,    (75)
        121,    106,    0,      (69)
        121,    102,    0,      1
        121,    103,    0,      0
        121,    104,    0,      0.1
        121,    105,    0,      -0.1
        102,    101,    102,    (74)
        124,    105,    0,      0.1
        216,    126,    101,    -2(0)
        104,    101,    102,    (74)
        125,    103,    105,    0              [Binary to decimal
        215,    104,    105,    2
        124,    103,    104,    0
        202,    126,    102,    -8(0)
        121,    105,    101,    0
        211,    126,    126,    -4.7(0)
        121,    126,    106,    0

74)     n10             /       n100
        n1000           /       0
```

```
69)   113,   103,   0,     5(60)
      121,   100,   0,     4(0)
      113,   100,   0,     5.4(9)
      121,   100,   0,     5(60)
      121,   126,   0,     2(62)                [Print n faults
      121,   100,   0,     (1/202)
      113,   100,   0,     5.4(9)
75)   121,   100,   0,     *01220045
      121,   126,   0,     (62)

                                               [END OF PASS 2

76)   101,   100,   0,     511.4(3)
      104,   100,   0,     511(3)
      122,   100,   0,     50
      215,   126,   100,   (4)                  [Return to pass 1 if BM missed
81)   121,   101,   0,     40
      113,   101,   0,     (90)                 [Set timer
      113,   0,     0,     2.4(90)              [Set EBA = 0
      121,   102,   0,     (30)
      113,   102,   0,     (92)                 [Set BA link
      121,   101,   0,     50
      113,   101,   0,     511.4(3)
      113,   0,     0,     1.4(90)
      113,   0,     0,     0.4(90)              [Set BM count
      121,   100,   0,     (77)
      113,   100,   0,     (91)                 [Set end of tape link
      121,   100,   0,     511.4(3)
      122,   100,   0,     (99)
      113,   100,   0,     1(90)                [Set error count
      121,   100,   0,     0.1
      113,   100,   0,     3*6                  [Inhibit interrupts
      121,   100,   0,     *00002525
      113,   100,   0,     21*6003              [Set TACR
      121,   100,   0,     *0026004
      113,   100,   0,     15*6003              [Start deck
      113,   0,     0,     3*6                  [Permit interrupts
      121,   100,   0,     *01230045
      121,   126,   0,     (62)

                                               [END OF PASS 3
77)   101,   100,   0,     2.4(90)
      122,   100,   0,     0.1
      101,   101,   0,     (92)
      210,   126,   101,   (79)                 [Jump if last interrupt not TBA
      106,   100,   0,     2(90)
      215,   126,   100,   (79)                 [Jump if EBA out of step
      121,   100,   0,     511.4(3)
      122,   100,   0,     (99)
      101,   101,   0,     1(90)
      101,   102,   0,     0.4(90)
      214,   126,   102,   8(0)
      165,   103,   101,   0.4
      214,   126,   103,   5(0)
      122,   101,   0,     0.4
      113,   101,   0,     1(90)
      113,   102,   101,   (99)
```

```
        121,   126,   0,     2(0)
        114,   102,   101,   (99)
        101,   102,   0,     1.4(90)
        110,   102,   0,     2(90)            [Adjust last address
        126,   101,   100,   0
        215,   126,   101,   (78)             [Jump if errors in pass 3
        121,   100,   0,     (40)
        113,   100,   0,     (92)
        121,   100,   0,     511.4(1)
        122,   100,   0,     (83)
        113,   100,   0,     1(90)
        121,   100,   0,     (80)
        113,   100,   0,     (91)
        121,   100,   0,     *00000525
        113,   100,   0,     21*6003
        121,   100,   0,     *0012012
        113,   100,   0,     15*6003          [Start deck
        121,   100,   0,     *01240045
        121,   126,   0,     (62)
79)     121,   100,   0,     2048*4
        113,   100,   0,     (92)
        121,   100,   0,     (4)
        113,   100,   0,     (91)
        121,   100,   0,     *00000525
        113,   100,   0,     21*6003
        121,   100,   0,     *0052012
        113,   100,   0,     15*6003          [Fast rewind
        121,   100,   0,     *01250045
        121,   126,   0,     (62)

                                               [END OF PASS 4
80)     101,   100,   0,     2.4(90)
        215,   126,   100,   (81)             [Return to pass 3 if EBA ≠ 0
        101,   100,   0,     (92)
        126,   100,   0,     (40)
        215,   126,   100,   (81)             [Return to pass 3 if expecting LBA
        101,   100,   0,     21*6003
        127,   100,   0,     0.2
        215,   126,   100,   (81)             [Return to pass 3 if addr fault set
        101,   100,   0,     1(90)
        122,   100,   0,     511.4(1)
        124,   100,   0,     (83)
        214,   126,   100,   (84)             [Exit if no errors
        123,   101,   100,   0
        163,   101,   0,     0
        163,   101,   0,     0
        110,   101,   0,     2(90)            [Adjust last address
        124,   100,   0,     0.4
        113,   101,   0,     1.4(90)          [Store bad block count
        121,   101,   100,   0
        121,   102,   100,   511.4(1)
        122,   102,   0,     (83)
        101,   103,   102,   (99)
        113,   103,   102,   (83)
        124,   102,   0,     0.4              [Copy list
        200,   126,   101,   -3(0)
        121,   101,   0,     -50
```

```
100)    113,    101,    0,      0.4(101)
        121,    101,    0,      511(3)
        122,    101,    0,      (99)
        121,    102,    0,      -n26
        113,    102,    100,    511(1)
        121,    103,    0,      *4
82)     101,    102,    100,    511.4(1)
        122,    102,    0,      0.1
        102,    102,    100,    511(1)
        124,    103,    0,      2
        214,    126,    102,    10(0)           [Jump if two consecutive errors
        125,    102,    0,      0
        163,    102,    0,      0
        163,    102,    0,      0
        113,    102,    101,    0.4(99)
        203,    126,    101,    2(0)
        121,    126,    0,      (63)            [Jump if too many faults
        217,    126,    103,    2(0)
        113,    103,    101,    2(99)
        121,    103,    0,      0
        200,    126,    100,    (82)
        124,    103,    0,      2
        113,    103,    101,    1(99)
        124,    101,    0,      0.4
        113,    101,    0,      1(90)
        113,    0,      101,    (99)
        122,    101,    0,      511(3)
        124,    101,    0,      (99)
101)    121,    100,    0,      -50
        104,    100,    101,    511.4(3)
        200,    126,    101,    -1(0)
        113,    100,    0,      0.4(90)         [Set count for search
        121,    100,    0,      (50)
        113,    100,    0,      (92)
        121,    100,    0,      (79)
        113,    100,    0,      (91)
        121,    100,    0,      *00000525
        113,    100,    0,      21*6003
        121,    100,    0,      *0026052
        113,    100,    0,      15*6003         [Start deck
        121,    100,    0,      *01260045
        121,    126,    0,      (62)
```

```
60)     *56573701      /      *57460142      [No. of b (60)
        *54574353      /      *63013601      [locks -
        *14141414      /      *00655655      [???? unm  2.5(60) 2.7(60)
        *57445146      /      *71014350      [odify ch
        *41565645      /      *54012700      [annel 7
        *14141414      /      *01464165      [???? fau 5(60)
        *54646300      /      *60456255      [lts perm 6.4(60)
        *51640167      /      *62516445      [it write
        *01575601      /      *43504156      [on chan
        *56455401      /      *27006457      [nel 7 to 9.6(60)
        *57015541      /      *56710146      [o many f
        *41655464      /      *63006350      [aults sh 11.6(60)
        *57626401      /      *42545743      [ort bloc
        *53630041      /      *44446245      [ks addre 13.3(60)
        *63635156      /      *47013601      [ssing -
        *60416363      /      *01140045      [pass ? e 15.7(60)
        *62625762      /      *01515601      [rror in
        *44416441      /      *00000000      [data

        (83)           =      (0)

        (0)            =      *3561
```

[10.6.64

```
                                                [450/10

                                                [PASS 1

3)    121,   125,   0,    (93)                  [Entry for BA interrupt

      (92)          =     -0.4(0)

106)  113,   126,   0,    4.4(1/416)            [Initial entry
      121,   102,   0,    (3)                   [End of tape entry
      113,   102,   0,    (102)                 [Reconnect to tape extracodes
      121,   109,   0,    4(0)
      113,   109,   0,    (5/201)               [set re-entry
      121,   109,   0,    3.0
      121,   126,   0,    (1/213)               [Wait for 1 second
      121,   109,   0,    (0)                   [Link to addressing SER
      121,   126,   0,    -4(0)                 [Wait for 1 second

      (91)          =     -1.4(0)

10)   101,   111,   0,    (90)                  [LAI TIMER
      203,   125,   111,  10(0)
      121,   111,   0,    (11)
      113,   111,   0,    (92)                  [Set BA link
      121,   111,   0,    *0000103              [Permit count, do not write 1's
      113,   111,   0,    21*6003               [Address tape
      121,   111,   0,    n8191
      113,   111,   0,    7*6003                [Set PBAR 7
      101,   111,   0,    (6/229)
      113,   111,   0,    3(90)                 [Note starting time
      121,   125,   0,    2048*4
      121,   112,   0,    *00002
      113,   112,   0,    21*6003               [Set LAI
      113,   111,   0,    (90)                  [Store count
      121,   125,   0,    2048*4

90)   0      /      0                           [Timer / BM count
      0      /      0                           [Error count / No. of bad blocks
      0      /      0                           [Last address / EBA
      0      /      0                           [Starting time / finishing time

11)   101,   111,   0,    0.4(90)               [BA INTERRUPT
      101,   112,   0,    (6/229)
      113,   112,   0,    3.4(90)
      101,   112,   0,    21*6003
      165,   113,   112,  *001                  [Select BM absent indicator
      101,   114,   0,    1(90)                 [Modifier
      215,   125,   113,  (15)                  [Jump if BM absent
      124,   111,   0,    1                     [Add 1 to BM count
      101,   113,   0,    7*6003
      126,   113,   0,    n8191
      215,   125,   113,  (12)                  [Jump if address ≠ 8191
      210,   125,   111,  3(0)                  [Jump if LBA
      165,   113,   112,  0.2
      215,   125,   113,  (14)                  [Jump if ADDRESS FAULT


                                                [10.6.64
      165,    113,    112,    *0002
      215,    125,    113,    2(12)          [Jump if RM absent
      210,    125,    111,    (13)           [Jump if LBA
      165,    115,    114,    0.4
      214,    125,    115,    5(14)          [Jump if switch set
      114,    111,    114,    (99)           [Add BM count to list
      121,    111,    0,      0              [Reset BM count
13)   126,    111,    0,      0.1            [Change BA indicator
      113,    111,    0,      0.4(90)        [Store BM count
      210,    125,    111,    2048*4         [Jump if TBA
      121,    111,    0,      *0000004
      113,    111,    0,      21*6003        [Write 1's
      121,    125,    0,      2048*4
12)   121,    113,    0,      n8191
      113,    113,    0,      7*6003
      211,    125,    111,    (14)           [Jump if TBA
      121,    113,    0,      0.2
      113,    113,    0,      21*6003        [Set Address Fault
      121,    125,    0,      (13)
14)   121,    113,    0,      0.1
      113,    113,    0,      21*6003        [Reset Address Fault
      114,    113,    0,      1.4(90)        [Add 1 to bad block count
      165,    115,    114,    0.4
      214,    125,    115,    -2(13)         [Jump if switch set
      202,    125,    114,    2(0)
      121,    125,    0,      (98)           [Jump if too many faults
      113,    114,    0,      1(90)
      113,    111,    114,    (99)           [BM count to list
      121,    125,    0,      -1(13)
15)   210,    125,    111,    3(12)          [Jump if LBA
      121,    113,    0,      0.1
      114,    113,    0,      1.4(90)        [Add 1 to bad block count
      121,    125,    0,      3(12)

                                             [PASS 2

20)   101,    111,    0,      1(90)          [BA INTERRUPT
      101,    112,    111,    (99)
      203,    125,    112,    8(0)           [Jump if no change to Write Ref.
      124,    111,    0,      0.4
      113,    111,    0,      1(90)
      165,    113,    111,    0.4
      121,    114,    0,      *000002
      215,    114,    113,    *000001
      113,    114,    0,      21*6003        [Set Write Ref.
      121,    125,    0,      1(20)
      113,    112,    111,    (99)
      121,    125,    0,      2048*4
```

```
                                                [PASS 3

30)   101,   111,   0,     (90)                 [LAI TIMER
      203,   125,   111,   8(0)
      121,   111,   0,     (31)
      113,   111,   0,     (92)
      121,   111,   0,     *0000101
      113,   111,   0,     21*6003              [Permit Count, Address Tape
      121,   111,   0,     n8189
      113,   111,   0,     7*6003               [Set up LBA for Block 0
      121,   125,   0,     2048*4
      121,   112,   0,     *00002
      113,   112,   0,     21*6003              [Set LAI
      113,   111,   0,     (90)                 [Store Count
      121,   125,   0,     2048*4
31)   101,   111,   0,     0.4(90)              [LBA INTERRUPT
      101,   112,   0,     21*6003
      165,   113,   112,   *001                 [Select BM absent indicator
      215,   125,   113,   (32)                 [Jump if BM absent
      124,   111,   0,     1                    [Add 1 to BM count
      101,   113,   0,     2.4(90)
      121,   114,   113,   -n5000
      214,   113,   114,   n8190                [WBA = 8190 if EBA = 5000
      214,   113,   113,   n8189                [WBA = 8189 if EBA = 0
      124,   114,   0,     n5000
      102,   114,   0,     2(90)
      214,   113,   114,   n8191                [WBA = 8191 if EBA = Last
      102,   113,   0,     7*6003
32)   215,   113,   113,   0.2                  [SET Addr. Fault if addr wrong
      113,   111,   0,     0.4(90)
      113,   0,     0,     7*6003               [Set TBA
      124,   113,   0,     0.4                  [Reset Address Tape
      113,   113,   0,     21*6003
      121,   111,   0,     0.1(33)
      113,   111,   0,     (92)                 [Set TBA Link
      121,   125,   0,     2048*4
33)   101,   111,   0,     0.4(90)              [TBA INTERRUPT
      101,   112,   0,     21*6003
      165,   113,   112,   *001                 [Select BM absent indicator
      101,   114,   0,     1(90)                [Modifier
      121,   116,   0,     *00001011            [Permit Count, ADDR. TAPE
      215,   125,   113,   (36)                 [Jump if BM absent
      124,   111,   0,     1                    [Add 1 to BM count
      101,   113,   0,     7*6003
      215,   125,   113,   (34)                 [Jump if address wrong
      165,   113,   112,   0.2
      215,   125,   113,   (34)                 [Jump if Address Fault
      165,   115,   114,   0.4
      214,   125,   115,   4(34)                [Jump if switch set
35)   114,   111,   114,   (99)                 [Add BM count to list
      113,   0,     0,     0.4(90)              [Reset BM count
      121,   125,   0,     3(36)
```

```
34)     121,    115,    0,      0.1
        114,    115,    0,      1.4(90)
        165,    115,    114,    0.4             [Add 1 to bad block count
        214,    125,    115,    (35)            [Jump if switch set
        202,    125,    114,    2(0)
        121,    125,    0,      (98)            [Jump if too many faults
        113,    114,    0,      1(90)
        113,    111,    114,    (99)            [BM count to list
        121,    125,    0,      1(35)
36)     124,    116,    0,      0.1             [Set Address Fault
        121,    111,    0,      0.1
        114,    111,    0,      1.4(90)         [Add 1 to bad block count
        113,    116,    0,      21*6003         [Set TACR
        101,    111,    0,      2.4(90)
        124,    111,    0,      0.1             [Add 1 to EBA
        121,    112,    111,    -n5000
        113,    111,    0,      2.4(90)
        214,    111,    112,    n8190           [WBA = 8190 if EBA = 5000
        124,    112,    0,      n5000
        102,    112,    0,      2(90)
        214,    111,    112,    n8191           [If last address, change to 8191
        113,    111,    0,      7*6003          [Set LBA
        121,    111,    0,      (31)
        113,    111,    0,      (92)
        121,    125,    0,      2048*4

                                                [PASS 4

40)     101,    111,    0,      21*6003         [TBA INTERRUPT
        127,    111,    0,      0.2
        214,    125,    111,    3(0)
        121,    111,    0,      (81)            [Set to re-enter pass 3 if
        113,    111,    0,      (91)            [Address Fault set
        121,    111,    0,      0.2
        113,    111,    0,      21*6003         [Set Address Fault
        121,    111,    0,      *0000022
        113,    111,    0,      15*6003         [Set Write, End Transfer
        101,    111,    0,      7*6003
        215,    111,    111,    0.1             [Set link odd if error
        124,    111,    0,      (41)
        113,    111,    0,      (92)            [Set LBA link
        121,    111,    0,      0.1
        110,    111,    0,      2.4(90)         [Subtract 1 from EBA
        121,    125,    0,      2048*4
41)     101,    111,    0,      2.4(90)         [LBA INTERRUPT
        121,    112,    111,    -n5000
        210,    125,    125,    8(0)            [Jump if error in TBA
        121,    113,    111,    0               [EBA
        214,    111,    111,    n8189           [WBA = 8189 if EBA = 0
        106,    113,    0,      2(90)
        214,    111,    112,    n8190           [WBA = 8190 if EBA = 5000
        214,    111,    113,    n8191           [WBA = 8191 if EBA = Last
        106,    111,    0,      7*6003
        214,    125,    111,    6(0)            [Jump if no error in block
```

```
       101,   111,   0,     1(90)
       124,   112,   0,     n5000
       113,   112,   111,   (99)          [Error -> list
       202,   0,     111,   0
       113,   111,   0,     1(90)         [Store bad addresses in list
       121,   111,   0,     (40)
       113,   111,   0,     (92)          [Set TBA link
       121,   125,   0,     2048*4

                                         [PASS 6

50)    101,   111,   0,     0.4(90)       [BA INTERRUPT
       122,   111,   0,     1
       113,   111,   0,     0.4(90)
       215,   125,   111,   2048*4        [Test for end of search
       121,   111,   0,     *0001
       113,   111,   0,     15*6003       [Stop deck
       121,   111,   0,     (51)
       113,   111,   0,     (92)
       121,   125,   0,     2048*4
51)    121,   112,   0,     (78)          [End of Search
       121,   125,   0,     (3/201)       [Switch to E

                                         [MISCELLANEOUS BA INTERRUPTS

98)    121,   111,   0,     *0001         [FAULT STOP - PASSES 1 AND 3
       113,   111,   0,     15*6003       [Stop deck
       121,   111,   0,     (52)
       113,   111,   0,     (92)
       121,   125,   0,     2048*4
52)    121,   112,   0,     (88)
       121,   125,   0,     (3/201)       [Switch to E
46)    121,   111,   0,     0             [SHORT TAPE STOP - PASS 1
       202,   125,   111,   6(0)          [Jump if not specified length
       121,   111,   0,     *0001
       113,   111,   0,     15*6003       [Stop deck
       121,   111,   0,     5(0)
       113,   111,   0,     (92)          [Set link for stop interrupt
       121,   125,   0,     2048*4
       113,   111,   0,     0.4(46)
       121,   125,   0,     2048*4
       121,   112,   0,     (70)
       121,   125,   0,     (3/201)       [Switch to E
48)    217,   125,   112,   2048*4        [SHORT TAPE STOP - PASS 3
       214,   125,   112,   2048*4        [Jump if not last block
       121,   111,   0,     *0001
       113,   111,   0,     15*6003       [Stop deck
       121,   111,   0,     3(0)
       113,   111,   0,     (92).         [Set link for stop interrupt
       121,   125,   0,     2048*4
       121,   112,   0,     (77)
       121,   125,   0,     (3/201)       [Switch to E

       (99)          =      (0)

       W
```

```
      R451                                              [MAIN PROGRAMME BLOCK

      (0) = *0001

1)    121,     1,      127,    0                        [Entry (1) addr: 0.1(1) readdr
      1132,    0,      0,      -9(14)                   [Set trap
      121,     2,      0,      (83/450)
      122,     2,      0,      511.4(1/450)             [Set modifier for input data
      121,     3,      0,      -0.1                     [Set for decimal conversion
2)    121,     8,      0,      0                        [Set for phase 1
      1054,    5,      0,      (18)                     [Read next character
      101,     6,      5,      (3)
      121,     7,      5,      0.1
      127,     7,      0,      0.3
      214,     127,    7,      4(0)
      125,     6,      0,      0
      122,     7,      0,      0.1                      [Look up character in directory
      215,     127,    7,      -2(0)                    [of permissible characters
      127,     6,      0,      7.4
      214,     127,    6,      (16)                     [Monitor if illegal character
      124,     6,      8,      -0.4                     [Select operation
      101,     127,    6,      (4)                      [Enter routine

3)    *000404          /       *1014202                 [Directory of permissible characters
      0                /       0
      *24242424        /       *24242424
      *2424            /       0
      0                /       0
      0                /       0
      0                /       0
      0                /       3

4)    1(2)             /       (5)                      [Phase 1
      (16)             /       1(2)
      (6)              /       (16)
      1(2)             /       (16)                     [Phase 2
      (2)              /       1(2)
      (16)             /       1(2)
      (7)              /       (8)                      [Phase 3
      (16)             /       (16)
      (9)              /       (16)
      (16)             /       (16)                     [Phase 4
      (16)             /       (16)
      (16)             /       (10)
      (11)             /       (16)                     [Phase 5
      2(6)             /       (16)
      (16)             /       1(2)
      (7)              /       (12)                     [Phase 6
      (16)             /       (16)                     [Phase 7
      (16)             /       (16)
      (16)             /       (13)
      (11)             /       (16)                     [Phase 8
      (15)             /       (16)
      (16)             /       1(2)
```

```
5)    121,   8,     0,    3                   [Set for phase 2
      121,   127,   0,    1(2)
6)    121,   4,     5,    -2                  [1st digit -> accumulator
      121,   9,     0,    7                   [Set digit count
      121,   8,     0,    6                   [Set for phase 3
      121,   127,   0,    1(2)
7)    113,   4,     2,    511.4(1)            [Store accumulator
      200,   127,   2,    2(0)                [Increase modifier
      121,   127,   0,    (17)                [Monitor if store full
      121,   3,     0,    0                   [Set for octal conversion
      121,   127,   0,    (2)                 [Go to phase 1
8)    121,   8,     0,    9                   [Set for phase 4
      121,   127,   0,    1(2)
9)    124,   4,     4,    0                   [Form 2a
      121,   10,    4,    0                   [Store 2a
      124,   4,     4,    0                   [Form 4a
      124,   4,     4,    0                   [Form 8a
      127,   10,    3,    0                   [Delete 2a if octal conversion
      124,   4,     10,   0                   [Complete multiplication by radix
      124,   4,     5,    -2                  [Add in next digit
      203,   127,   9,    1(2)                [Jump if less than 8 digits read
15)   121,   8,     0,    15                  [Set for phase 6
      121,   127,   0,    1(2)
10)   121,   8,     0,    12                  [Set for phase 5
      121,   127,   0,    1(2)
11)   113,   4,     2,    511.4(1)            [Store accumulator
      200,   127,   2,    2(0)                [Increase modifier
      121,   127,   0,    (17)                [Monitor if store full
      121,   3,     0,    0                   [Set for octal conversion
      121,   127,   0,    (5)                 [Go to phase 2
12)   121,   8,     0,    16                  [Set for phase 7
      121,   127,   0,    1(2)
13)   121,   8,     0,    19                  [Set for phase 8
      121,   127,   0,    1(2)

14)   (19)          /     0                   [Trap vector

16)   121,   97,    0,    0.1
      121,   91,    0,    3(0)                [Monitor - error in data
      1157,  0,     0,    1(1/247)            [Switch to E
      1117,  0,     0,    0                   [End programme
      121,   100,   0,    15.7(60/450)
      121,   126,   0,    (31)                [Print
17)   121,   97,    0,    0.1
      121,   91,    0,    3(0)                [Monitor - too many faults
      1157,  0,     0,    1(1/247)            [Switch to E
      1117,  0,     0,    0                   [End programme
      121,   100,   0,    9.6(60/450)
      121,   126,   0,    (31)                [Print
18)   122,   8,     0,    0.2                 [End of record
      217,   127,   8,    (2)                 [Ignore if phases 1 or 2
      121,   127,   0,    (7)
```

[10.6.64

```
19)   1132,   0,     0,    *4              [Trap for end of input
      113,    1,     0,    (22)            [Set routine in 'no fault' mode
      122,    2,     0,    (83/450)
      124,    2,     0,    509.4(1/450)    [No. of items -4
      121,    3,     2,    1
      211,    127,   1,    2(0)            [Jump if addressing
      123,    3,     2,    0.4
      216,    127,   3,    (16)            [Monitor if wrong no. of items
      211,    127,   1,    (26)            [Jump if addressing
      121,    3,     2,    0               [No. of items -1
      121,    4,     2,    0               [Set scan upper limit
      121,    5,     4,    0               [Set No. of items to be scanned
      121,    4,     0,    0               [Clear interchange marker
      121,    6,     3,    0.1             [Set modifier
20)   202,    127,   5,    4(0)            [Scan list
      202,    127,   4,    -4(0)           [Count through scans
      113,    2,     0,    0.4(22)         [Store No. of items in list
      121,    127,   0,    (21)            [Exit from sort
      101,    8,     6,    1(50)           [Pick up element Ak -1
      121,    7,     8,    0               [Copy
      102,    7,     6,    1.4(50)         [Subtract element Ak
      217,    127,   7,    7(0)            [Jump if Ak -1 < Ak
      214,    127,   7,    7(0)            [Jump if Ak -1 = Ak
      113,    8,     6,    1.4(50)         [Replace Ak by Ak-1
      122,    8,     7,    0
      121,    4,     3,    0.4
      113,    8,     6,    1(50)           [Replace Ak-1 by Ak
      122,    4,     6,    -0.1            [Set interchange marker
      202,    127,   6,    (20)            [Cycle scan
      121,    7,     8,    *6
      214,    127,   7,    -2(0)           [Jump if Ak-1 is a dummy
      121,    8,     0,    *2
      113,    8,     6,    1.4(50)         [Replace Ak by dummy
      122,    2,     0,    0.4             [Reduce No. items counter
      121,    4,     3,    0.4
      122,    4,     6,    -0.1            [Set interchange marker
      202,    127,   6,    (20)            [Scan Cycle
21)   101,    3,     0,    (50)
      102,    3,     2,    1.4(50)
      217,    127,   3,    (16)            [Monitor if last fault out of range
      121,    97,    0,    0.1
      121,    91,    0,    (41)
      1157,   0,     0,    1(1/247)        [Switch to E
      101,    3,     0,    1.4(50)
      214,    127,   3,    (26)            [Jump if Block 0 in error list
      1002,   7,     0,    *0002           [Read next block
      121,    3,     0,    0.4
      101,    4,     3,    0.4(50)
      106,    4,     3,    1*0002
      215,    127,   4,    (16)            [Monitor if identifier incorrect
      202,    127,   3,    -3(0)
      124,    1,     0,    0.2             [Mark B1 if Block 0 read
```

```
26)   113,   1,    0,    (22)
      113,   2,    0,    5.4(23)
      121,   97,   0,    0.1
      121,   91,   0,    (23)
      1157,  0,    0,    1(1/247)        [Switch to E
      101,   2,    0,    (22)
      214,   127,  2,    (24)            [Jump if program monitored
      111,   127,  0,    *0002
34)   121,   8,    0,    0
      121,   9,    8,    -n5000
      216,   8,    9,    n5000           [Set upper limit of 5000 blocks
      124,   8,    8,    0
      124,   8,    8,    -0.4
      113,   8,    0,    0.4*0002        [Set No. of blocks
      1121,  2,    0,    0
      113,   2,    0,    1*0002          [Set date
      1120,  3,    0,    0
      113,   3,    0,    1.4*0002        [Set time
      121,   4,    0,    *45703663
      113,   4,    0,    2*0002
      121,   4,    0,    *71636445
      113,   4,    0,    2.4*0002
      121,   4,    0,    *55
      113,   4,    0,    3*0002          [Set name = EX - SYSTEM
      1004,  7,    0,    *0002           [Write Block 0
      113,   2,    0,    1.4(24)         [Dump date
      113,   3,    0,    2.4(24)         [Dump time
      122,   8,    0,    0.4             [Set count
      1004,  7,    0,    *0007
      202,   127,  8,    -1(0)           [Clear tape
      1017,  7,    0,    0               [Free tape
      121,   97,   0,    0.1
      121,   91,   0,    3(0)
      1157,  0,    0,    1(1/247)
34)   1117,  0,    0,    0               [End of programme
      121,   100,  0,    0
      121,   101,  0,    0
      121,   107,  0,    1
      121,   105,  0,    0
      121,   106,  0,    1
      121,   102,  0,    1
      121,   103,  0,    0
      125,   100,  0,    0
      121,   104,  100,  0
      163,   100,  0,    0
      163,   100,  0,    0
      163,   100,  0,    0               [Convert from octal to 6 - bit
      125,   103,  0,    2
      164,   103,  100,  0.7
      121,   100,  104,  0
      211,   126,  126,  -2.7(0)
      203,   126,  102,  -9(0)
```

```
        113,    103,    105,    1.4(28/450)
        124,    105,    0,      0.4
        203,    126,    106,    -14(0)
        124,    105,    0,      0.4
        121,    100,    101,    0
        203,    126,    107,    -18(0)
        121,    100,    0,      (28/450)
31)     121,    101,    0,      (1/202)
        113,    101,    0,      5.4(9/450)         [Set return from print routine
        121,    126,    0,      2(62/450)          [Print identifiers

22)     0               /       0

23)     121,    101,    0,      2048*4             [Pick up jump address for pass 1
        121,    102,    0,      2048*4             [Pick up jump address for pass 3
        101,    100,    0,      (22)
        113,    100,    0,      7.4(24/450)        [Store key
        210,    126,    100,    10(0)              [Jump if re-addressing
        121,    103,    0,      0
        124,    103,    0,      2
        214,    126,    103,    7(0)               [Jump if length not specified
        101,    102,    0,      (50)
        124,    102,    102,    0
        124,    102,    102,    0
        113,    102,    0,      0.4(4/450)         [Store length for pass 1
        121,    101,    0,      (46/450)           [Pick up jump address for pass 1
        121,    102,    0,      (48/450)           [Pick up jump address for pass 3
        113,    101,    0,      2.4(13/450)        [Set jump in pass 1
        113,    102,    0,      15.4(36/450)       [Set jump in pass 3
        211,    126,    100,    (33)               [Jump if addressing
        101,    101,    0,      0.4(22)            [Set counter
        121,    103,    0,      0.4
        101,    102,    101,    1.4(50)
        113,    102,    103,    511(1/450)         [Move error list into correct
        122,    103,    0,      0.4                [position for end of pass 4
        202,    126,    101,    -3(0)
        113,    0,      0,      1.4(90/450)        [Inhibit fault printout after pass 6
        121,    101,    103,    -1                 [Pick up No. of faults
        163,    101,    0,      0
        163,    101,    0,      0
        104,    101,    0,      (50)               [Adjust last address
        113,    101,    0,      2(90/450)          [Store last address
        113,    103,    0,      12.4(24/450)       [Store modifier
        121,    108,    0,      (25/450)           [Set link
        121,    126,    0,      3(17/450)
33)     121,    100,    0,      0.7(25/450)
        121,    126,    0,      1(17/450)
```

```
39)     113,     0,       0,      7(58/400)            [Remove deck from Supervisor
        121,     100,     0,      *0000412
        113,     100,     0,      15*6003              [Disengage deck
        121,     100,     0,      *00000525
        113,     100,     0,      21*6003              [Reset TACR
        121,     100,     0,      4.1(28/450)
        121,     126,     0,      (31)                 [Instruct operator

40)     101,     100,     0,      2(90/450)
        113,     100,     0,      0.4(34)
        101,     100,     0,      7.4(24/450)
        113,     100,     0,      (22)
        214,     126,     100,    (29)
        121,     126,     0,      (1/202)

41)     113,     126,     0,      7.4(24/450)
        121,     126,     0,      (17/450)
```

[10.6.64

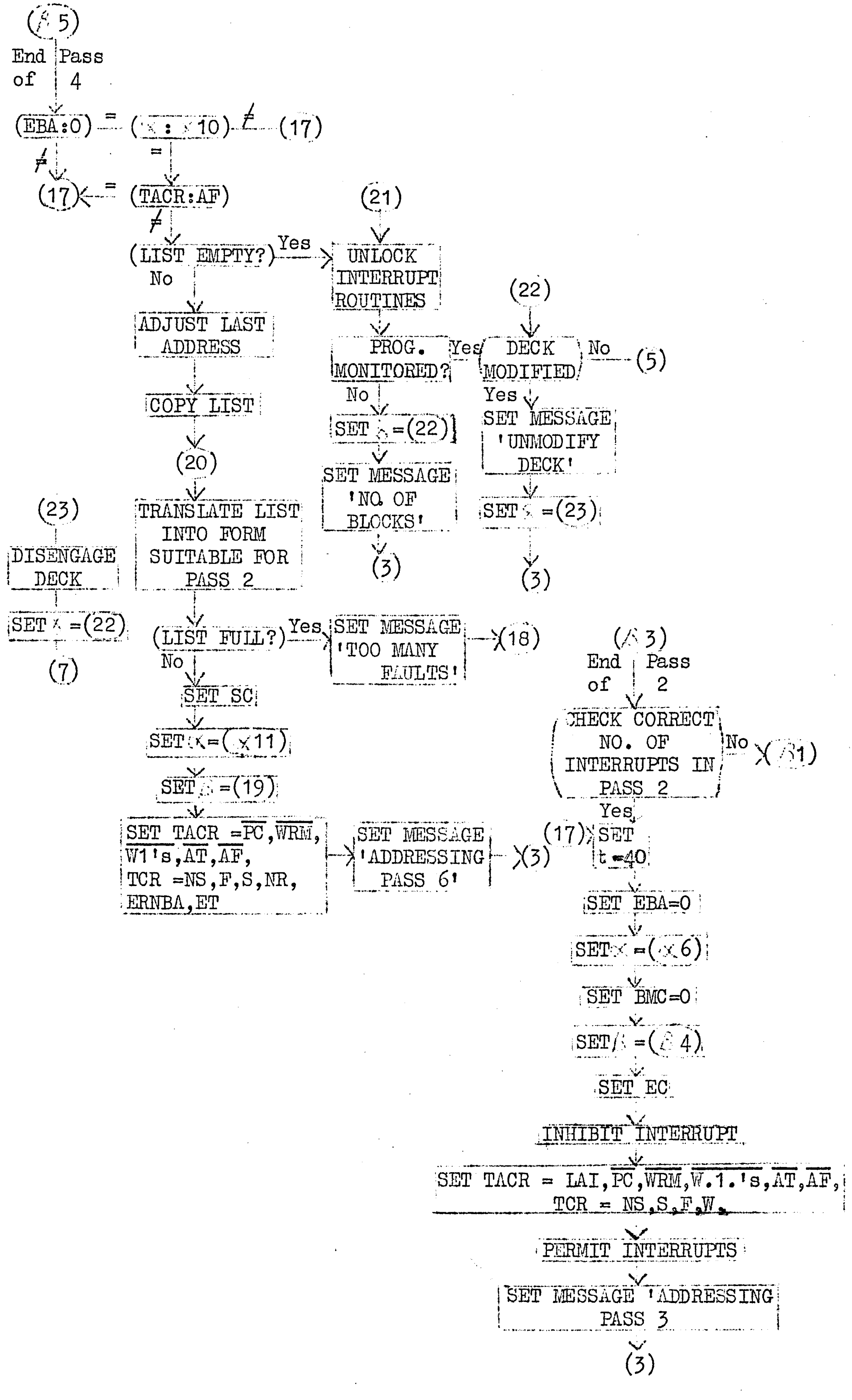
T
TAPE ADDRESSING AND RE-ADDRESSING ROUTINE   R450 AND R451

```
(1/450)       =     (17/450)
(8/450)       =     0.4
(93/450)      =     (3/450)
(50/451)      =     *4221(83/450)
(102/450)     =     7(54/400)
(1/201)       =     9*4007           |Extracode entry
(3/201)       =     19*4007          |Tape interrupt entry
(5/201)       =     395*7            |Current SER entry
(1/202)       =     50*4007          |Program Scan entry
(5/203)       =     948.4*7          |Current Program No.
(17/204)      =     213*4007         |Halt Main Program
(9/214)       =     184*40074        |Free Main Program
(1/216)       =     200*4002
(1/220)       =     230*4            |Reserve Operator's Output
(2/220)       =     233.1*4          |Free Operator's Output
(6/229)       =     941.4*7          |Main Clock
(1/240)       =     925719
(3/240)       =     925744
(1/318)       =     96*40044         |Call to Cores entry
(1/329)       =     208*4            |Remove Lock Down
(58/400)      =     312.4*7
(54/400)      =     296.4*7
(1/416)       =     926006
(7/201)       =     399*7            |Current SER Base
(1/213)       =     134*40074
(1/247)       =     235*4001
(4/203)       =     953*7
(6/202)       =     369*7
(7/204)       =     379*7
(9/204)       =     949*7
```

|10.6.64

R450,451/1

TAPE ADDRESSING AND RE-ADDRESSING



Addressing and Readdressing Routine for Atlas I Magnetic Tape

Purpose

To test and address magnetic tape in accordance with the format laid out in the Atlas I General Description and to remove, from existing addressed tapes, any blocks found to be faulty.

Method of Use.

1. Addressing:-
   Punch a steering tape as follows:-

```
JOB
VAS - ADDRESS TAPE

COMPILER TAD

A
n

***Z
```

Where n is the maximum number of blocks to be marked out (in decimal). If n is not punched the tape will be addressed from end to end.

2. Readdressing.

   Punch a steering tape as follows:-

```
JOB
VAS - READDRESS TAPE

COMPILER TAD

R
a
b
c
d.
d_k
***Z
```

Where a = no. of blocks (in decimal))
b = identifier (1st half)) attainable from the log
c = identifier (2nd Half)) of addressed tapes
$d_o \rightarrow d_k$ = Faulty block addresses (inoctal)

N.B. Spaces may be used to terminate any of the elements in the data, but other deviation from this format will cause the programme to monitor.

3. For both addressing and readdressing:

Mount the magnetic tape on deck 7 but do not engage until instructed to do so by the operator's teleprinter. Feed in the steer tape on any tape reader.

When the routine has been entered, the message 'ENGAGE DECK 7' will be printed. Engaging the deck will cause the program to start and instructions and comments will be printed from time to time.

N.B. The commands 'MODIFY CHANNEL 7', 'UNMODIFY CHANNEL 7', and 'PERMIT WRITE ON CHANNEL 7' are accompanied by the deck being disengaged. Obey the command and then re-engage the deck.

TAPE ADDRESSING AND READDRESSING

DESCRIPTION

Summary

A Supervisor routine which will address tapes or remove from previously addressed tapes any blocks whose addresses are specified by a steering tape. Deck time is approximately 7 mins. per 1000 blocks.

Method

The process is broken up into passes as follows:-

Pass 1:-

A 20 ft. length of clear tape is run out to serve as a leader. This is followed by marked out blocks containing 8191 in the leading and trailing addresses and "all 1/s" in the region where information can be written. During this pass, a block is deemed faulty if:

1. A Reference Mark cannot be written.
2. A Block Mark can not be written.
3. Either address read back is not 8191
4. All 1's cannot be written in the information area.

Since addresses are not written sequentially in this pass, the fault list is compiled as a set of strings, consecutive entries in the list referring to good and faulty regions of tape, respectively. Each entry in the list is the count of the number of good block marks in the corresponding region.

Pass 2:-

The last 6 blocks are regarded as faulty to serve as a trailer and, by reference to the list prepared during pass 1, the Reference Marks associated with the faulty blocks are erased.

Pass 3:-

Using the Reference Marks which are left after pass 2, blocks are rewritten w ith sequential addresses from 0 onwards (except that 0,5000 and the last block are addressed as 8189, 8190, and 8191 respectively). The information area is erased and no clock pulses are written. Snigs ar e written in the interblock gap and a block is deemed faulty if:

1; A block mark cannot be written
2. The leading address is read back incorrectly.
3. The trailing address is not read back as zero.

The presence of faults reverts the programme to a previous pass.

Pass 4:-

Checks in reverse that the addresses are correct and that there is at least one snig in each inter-block gap.

Pass 5:-

Rewind connecting pass 3 to pass 1.

Pass 6:-

Search routine to enter pass 2 from pass 4.

After the successful completion of pass 4, Block 0 is written, the tape name being EX-SYSTEM. The deck is then unmodified, and a block of floating point zeros is written from Block 1 to Block 4999 or the penultimate block,which ever is the earlier. The tape is then rewound and named FREE. These two passes are not given a number within the tape addressing process.

TAPE ADDRESSING AND READDRESSING

MODES OF BEHAVIOUR

1. Monitoring

The Programme ends under the following fault conditions:-

OPERATOR'S OUTPUT	Reason
Error in data	Illegal character, wrong no. of entries, incorrect identifiers or otherwise inconsistent data on the steering tape.
Short blocks.	Average block length during pass 1 nominally less than 7.5 inches.
Too many faults.	Insufficient working space to accommodate entries from data tape or from fault checking routines. Working space is adequate for at least 190 faulty blocks at any stage in the process. (During passes 1 and 3, due to the Stringing System a specific number of blocks cannot be quoted. Here the number of strings can be 277.)

2. POSSIBLE PATHS THROUGH THE ROUTINE

Normally, the process goes through passes 1,2,3, and 4 in that order. Reasons for reversion to previous passes are given below:-

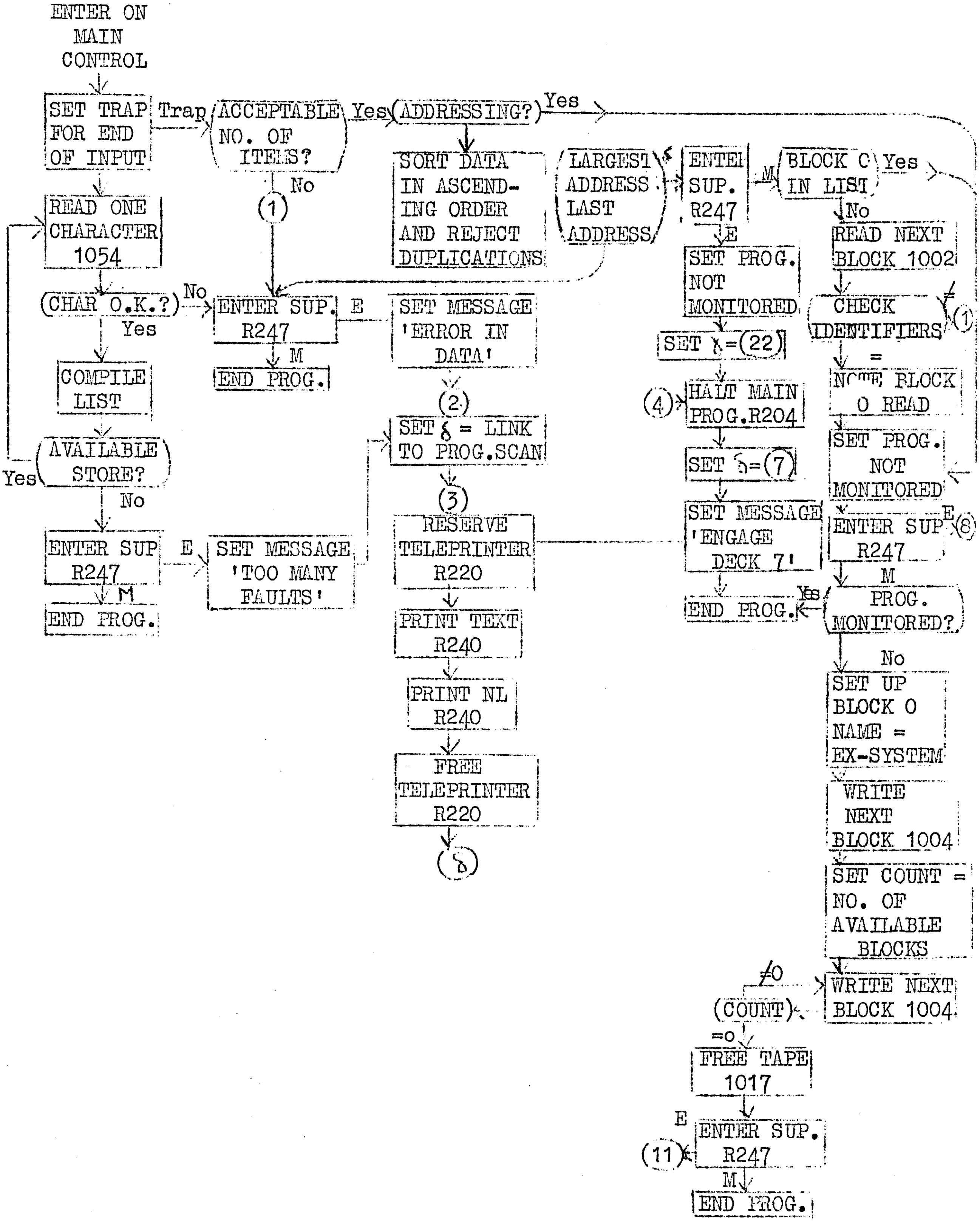
TRANSFER	REASON
2 → 1	No. of BM's read in pass 2 differs from the number classed as written during pass 1.
3 → 2	Faulty blocks written during pass 3.
3 → 5 → 1	No. of RM's read in pass 3 differs from the number expected to remain after erazing in pass 2.
4 → 3	No. of BM's read in pass 4 differs from the number classed as written in pass 3 or snig missing from interblock gap.
4 → 6 → 2	Address errors detected in pass 4.

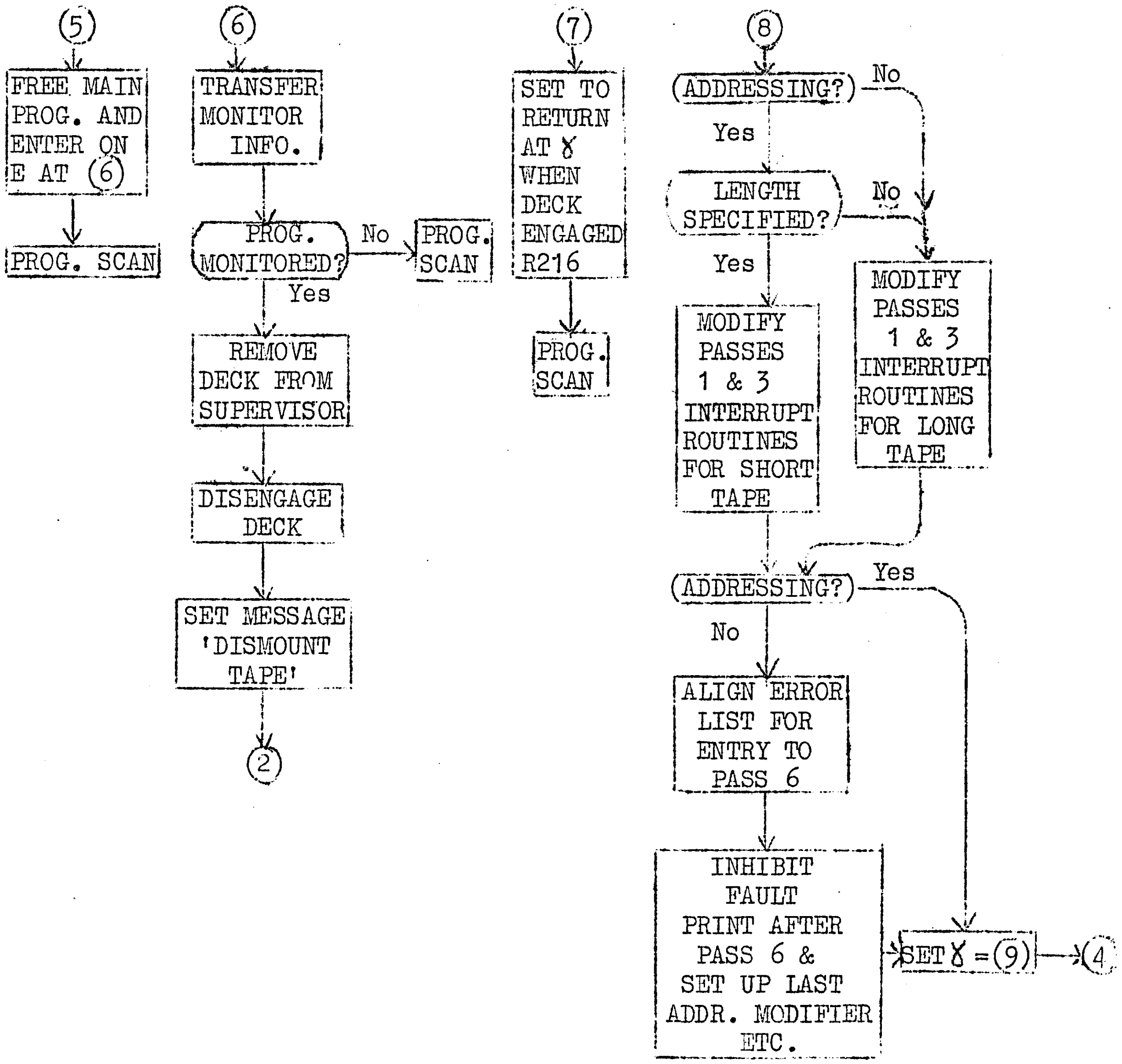
TAPE ADDRESSING AND READDRESSING
GLOSSARY OF TERMS

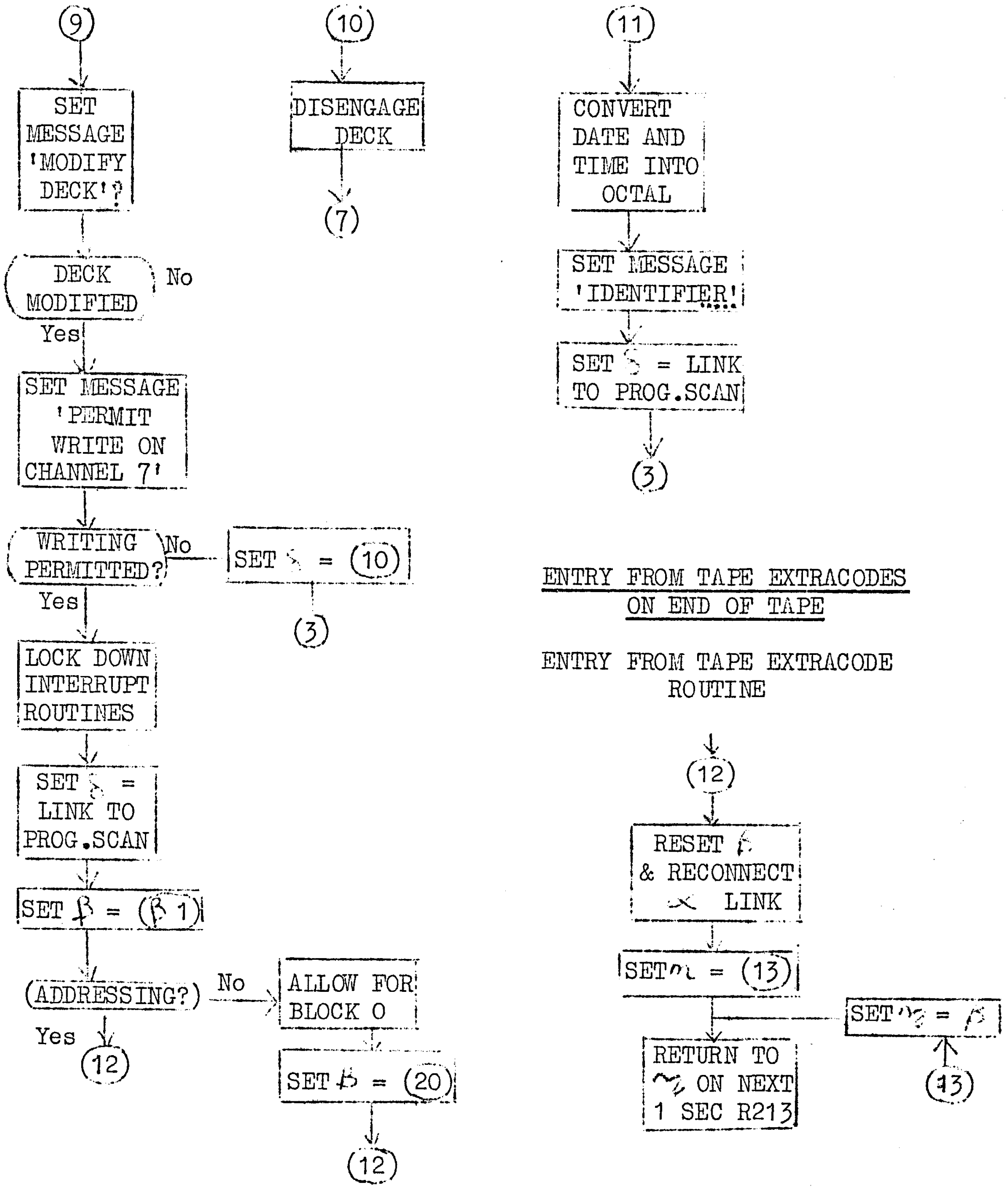
*	2048*4 - Exit to Sort Interrupts.
α	Link obeyed on a Block Address Interrupt
β	Link obeyed on End of tape.
γ	Link to return from R216 when Deck 7 is engaged
δ	Link obeyed after Operator's Output
η	Link from 1 sec. SER
AF	Address Fault.
AT	Address Tape
BA	Block Address
BBC	Bad Block Count.
BM	Block Mark
BMC	Block Mark Count.
DD	Disengage Deck.
EBA	Expected Block Address
EC	Error Count
ERNBA	End Read at Next Block Address
ET	End Transfer
F	Forwards
FS	Fast Speed
LAI	If Deck modified and AT reset, causes BA interrupt after 40 insec.
LBA.	Leading Block Address
NR	Normal Read
NS	Normal Speed
PBAR	Present Block Address Register
PC	Permit Count
R	Reverse
RM	Reference Mark
S	Start
SC	Search Count
t	Timing count for leader in passes 1 and 3
T	Time in seconds between 1st and last BM's during pass 1.
TAC	Tape Addressing Command.
TACR	Tape Addressing Command Register
TBA	Trailing Block Address
TCR	Tap e Command Register
W	Write
WBA	Wanted Block Address
W1's	Write 1's
WRM	Write Reference Mark.
SNIG	A "return to zero" pulse in the clock track recorded after the end of the information stripes because there are an odd number of these in the complete block.

TAPE ADDRESSING AND RE-ADDRESSING

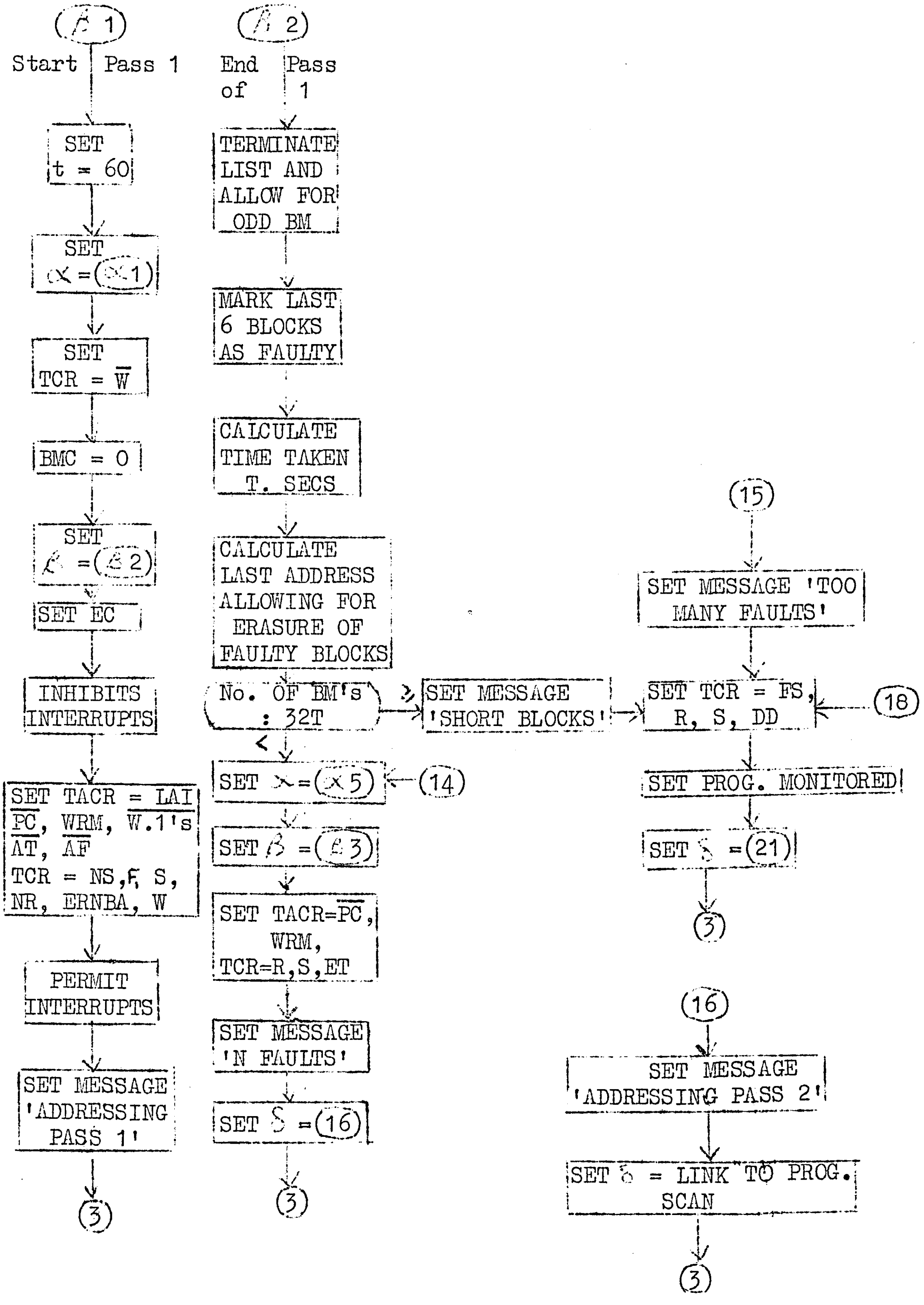
MAIN PROGRAMME AND MONITOR ROUTINES



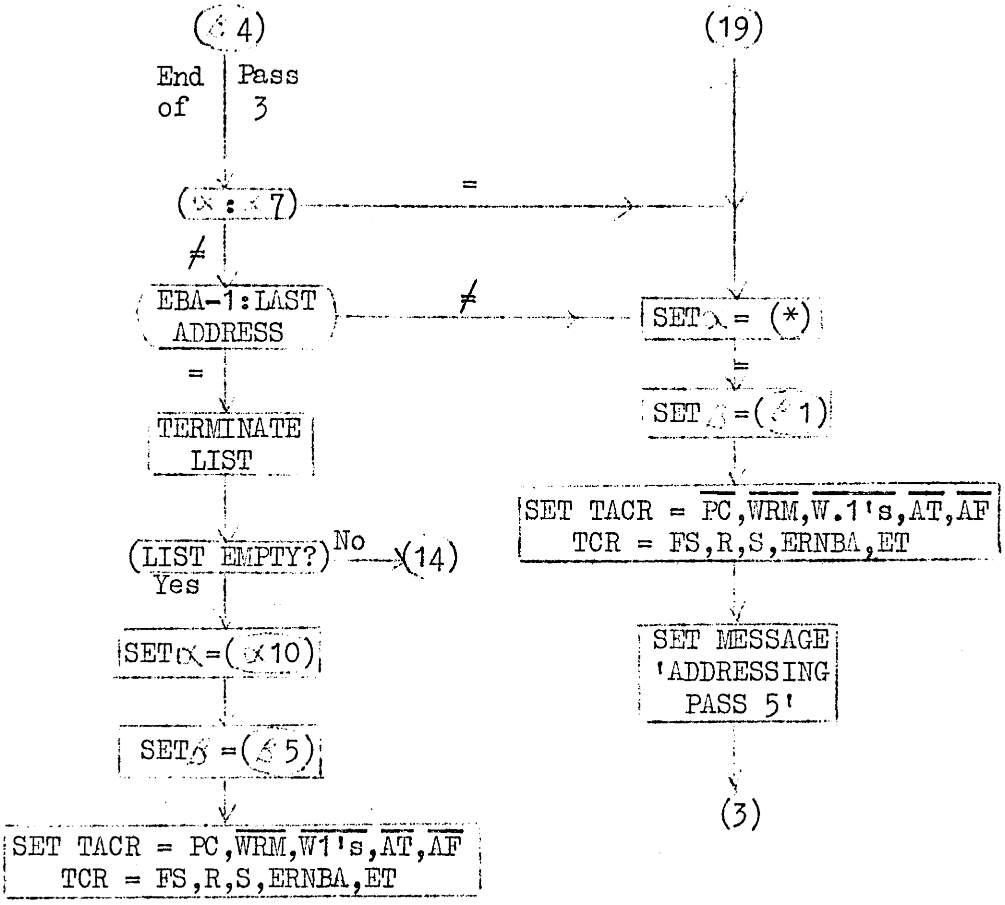

5
FREE MAIN PROG. AND ENTER ON E AT (6)
PROG. SCAN
6
TRANSFER MONITOR INFO.
PROG. MONITORED?
No
PROG. SCAN
Yes
REMOVE DECK FROM SUPERVISOR
DISENGAGE DECK
SET MESSAGE 'DISMOUNT TAPE'
2
7
SET TO RETURN AT γ WHEN DECK ENGAGED R216
PROG. SCAN
8
ADDRESSING?
No
Yes
LENGTH SPECIFIED?
No
Yes
MODIFY PASSES 1 & 3 INTERRUPT ROUTINES FOR SHORT TAPE
MODIFY PASSES 1 & 3 INTERRUPT ROUTINES FOR LONG TAPE
ADDRESSING?
Yes
No
ALIGN ERROR LIST FOR ENTRY TO PASS 6
INHIBIT FAULT PRINT AFTER PASS 6 & SET UP LAST ADDR. MODIFIER ETC.
SET γ = (9)
4


9
SET MESSAGE 'MODIFY DECK'?
DECK MODIFIED
No
Yes
SET MESSAGE 'PERMIT WRITE ON CHANNEL 7'
WRITING PERMITTED?
No
SET ζ = (10)
3
Yes
LOCK DOWN INTERRUPT ROUTINES
SET ζ = LINK TO PROG.SCAN
SET β = (β 1)
(ADDRESSING?)
No
ALLOW FOR BLOCK 0
SET β = (20)
12
Yes
12
10
DISENGAGE DECK
7
11
CONVERT DATE AND TIME INTO OCTAL
SET MESSAGE 'IDENTIFIER'
SET ζ = LINK TO PROG.SCAN
3
ENTRY FROM TAPE EXTRACODES ON END OF TAPE
ENTRY FROM TAPE EXTRACODE ROUTINE
12
RESET β & RECONNECT α LINK
SET η = (13)
SET η = β
RETURN TO η ON NEXT 1 SEC R213
13

TAPE ADDRESSING AND RE-ADDRESSING 'END OF TAPE' ROUTINES

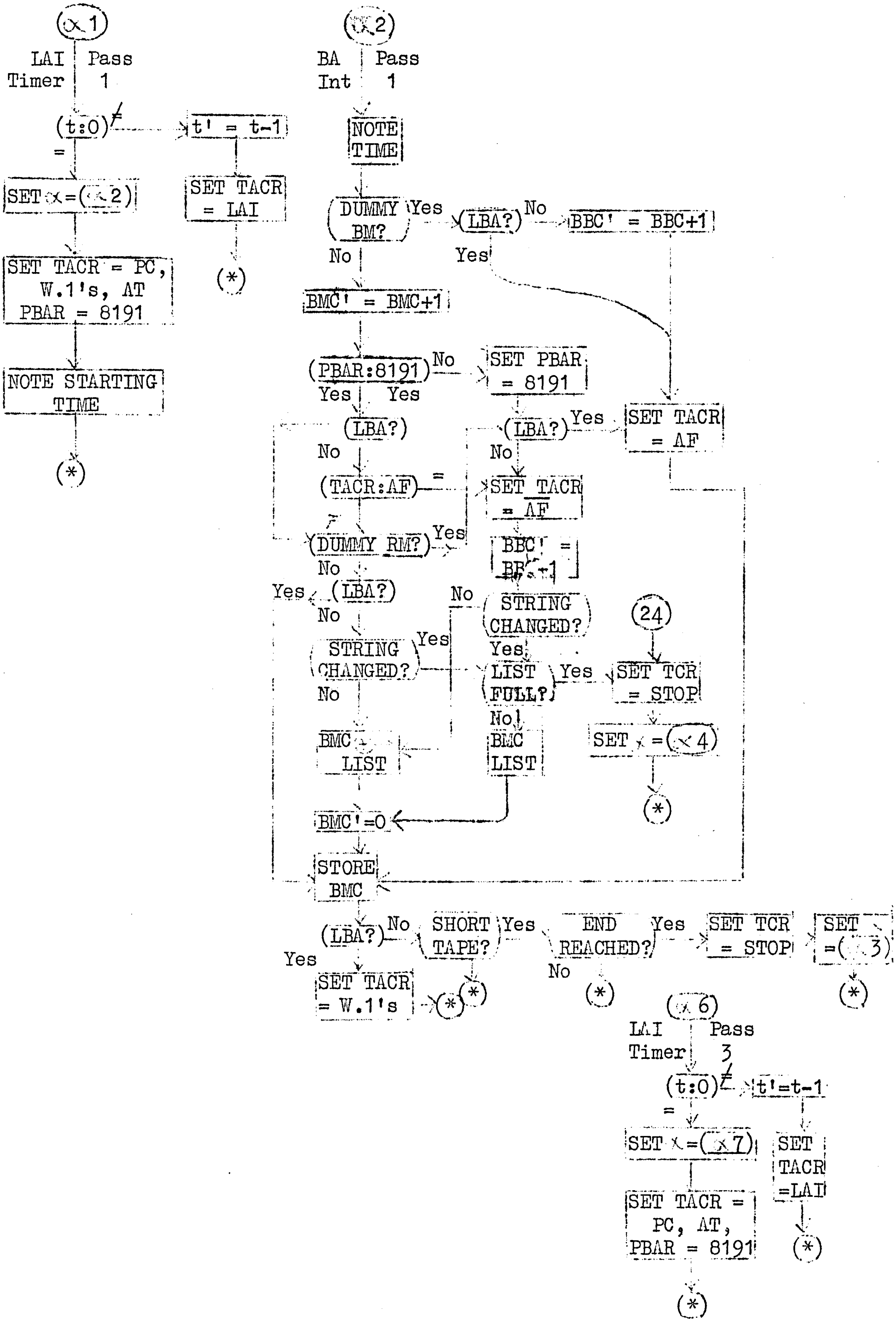


β1
Start Pass 1
SET t = 60
SET α = (α1)
SET TCR = W̄
BMC = 0
SET β = (β2)
SET EC
INHIBITS INTERRUPTS
SET TACR = LAI, P̄C̄, WRM, W̄.1's, ĀT̄, ĀF̄
TCR = NS, F, S, NR, ERNBA, W
PERMIT INTERRUPTS
SET MESSAGE 'ADDRESSING PASS 1'
3
β2
End of Pass 1
TERMINATE LIST AND ALLOW FOR ODD BM
MARK LAST 6 BLOCKS AS FAULTY
CALCULATE TIME TAKEN T. SECS
CALCULATE LAST ADDRESS ALLOWING FOR ERASURE OF FAULTY BLOCKS
No. OF BM's : 32T
≥
<
SET MESSAGE 'SHORT BLOCKS'
SET α = (α5)
14
SET β = (β3)
SET TACR = P̄C̄, WRM, TCR = R, S, ET
SET MESSAGE 'N FAULTS'
SET S = (16)
3
15
SET MESSAGE 'TOO MANY FAULTS'
SET TCR = FS, R, S, DD
18
SET PROG. MONITORED
SET S = (21)
3
16
SET MESSAGE 'ADDRESSING PASS 2'
SET S = LINK TO PROG. SCAN
3


(β4)
End of Pass 3
(α:α7)
=
≠
EBA-1:LAST ADDRESS
≠
=
TERMINATE LIST
(LIST EMPTY?)
No
(14)
Yes
SET α=(α10)
SET β=(β5)
SET TACR = PC,$\overline{\text{WRM}}$,$\overline{\text{W1's}}$,$\overline{\text{AT}}$,$\overline{\text{AF}}$
TCR = FS,R,S,ERNBA,ET
(19)
SET α = (*)
SET β=(β1)
SET TACR = $\overline{\text{PC}}$,$\overline{\text{WRM}}$,$\overline{\text{W.1's}}$,$\overline{\text{AT}}$,$\overline{\text{AF}}$
TCR = FS,R,S,ERNBA,ET
SET MESSAGE 'ADDRESSING PASS 5'
(3)

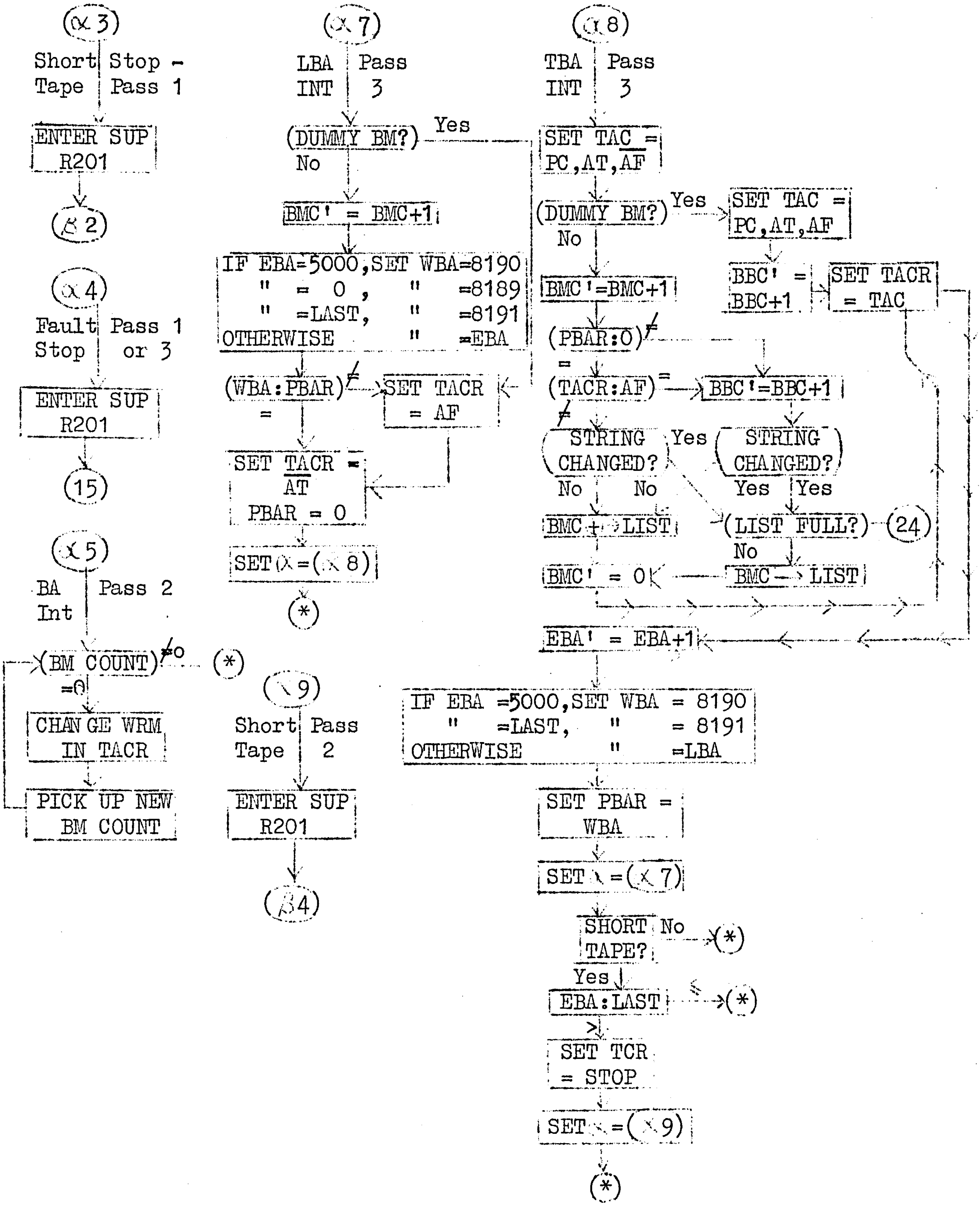
TAPE ADDRESSING AND RE-ADDRESSING

BA INTERRUPT ROUTINES

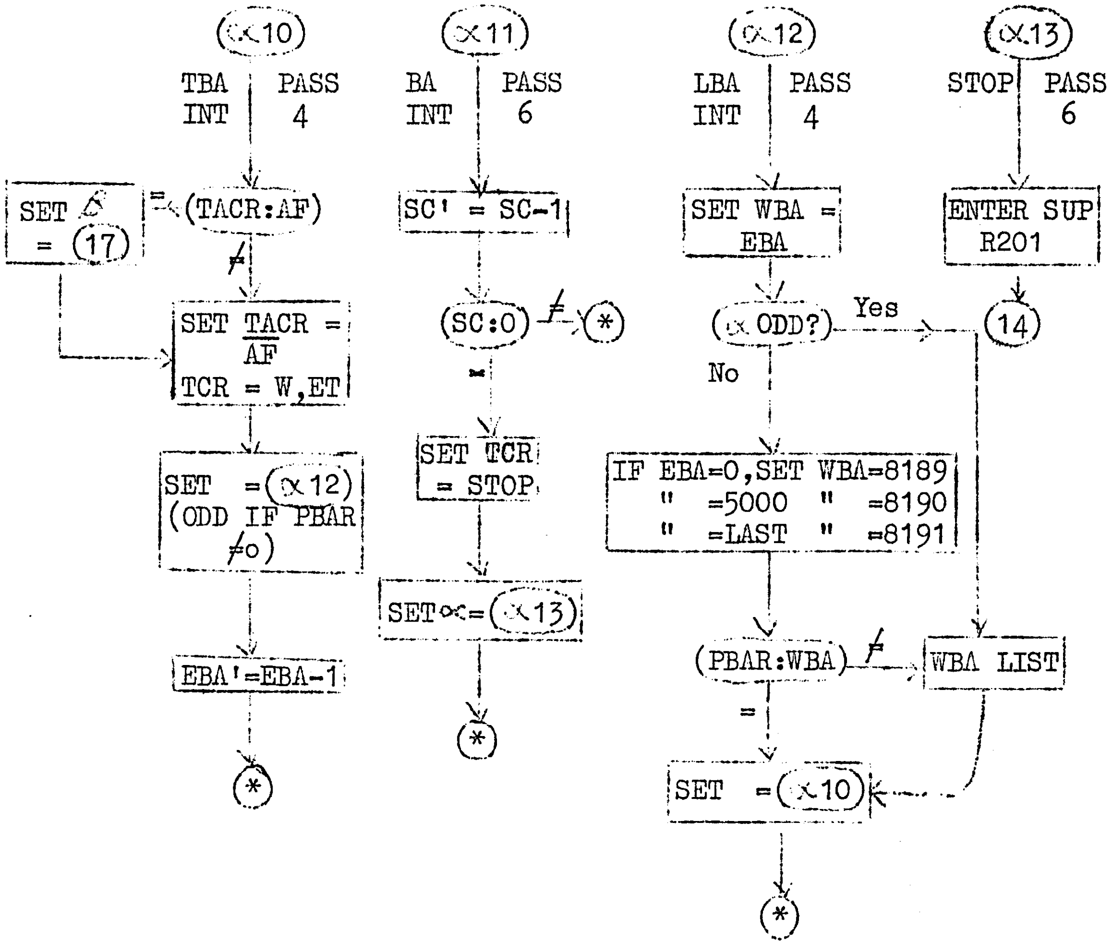


TAPE ADDRESSING AND RE-ADDRESSING

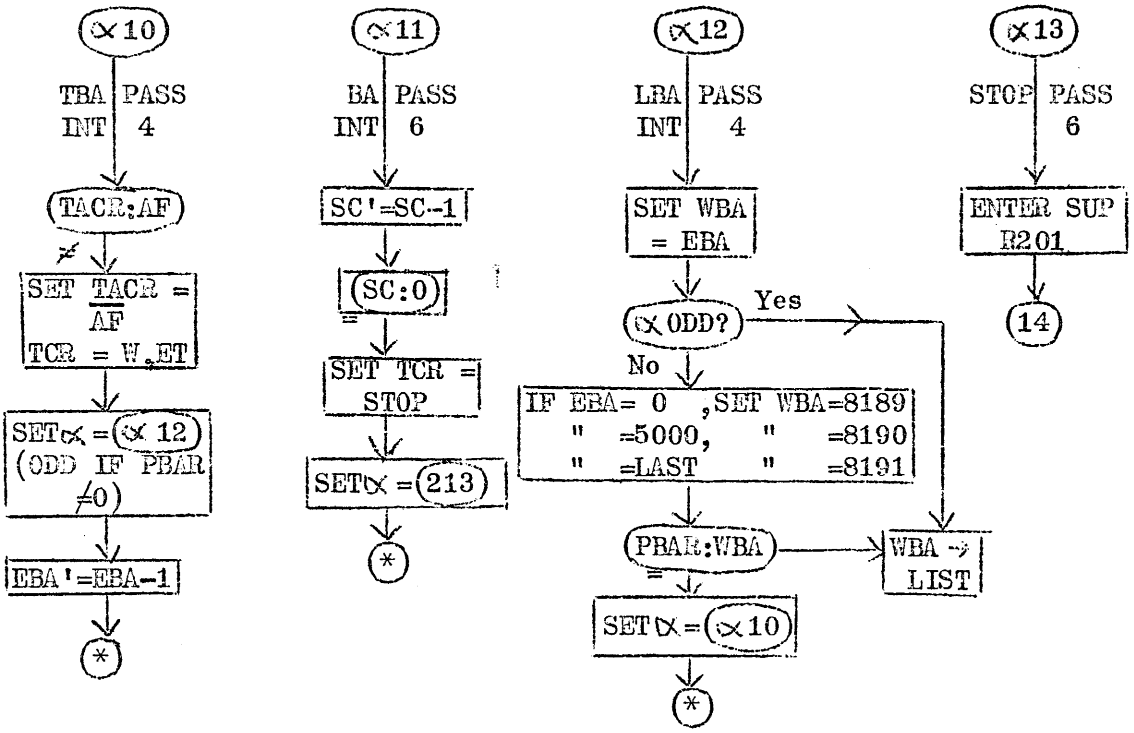
BA INTERRUPT ROUTINES



TAPE ADDRESSING AND RE-ADDRESSING
BA INTERRUPT ROUTINES



TAPE ADDRESSING AND RE-ADDRESSING
BA INTERRUPT ROUTINES



R480: Tape engage and disengage

Purpose: A main store SER to analyse decks detected by the One Second routine as having the engage status changed. A suitable SER is entered to the tape SER queue to deal with each deck calling for action.

Registers of main store: 26

Instructions obeyed: 3 + 6D+(8 to 23 per deck requiring attention) where D = no. of decks

Parameters used: (1) to (16)

Cross references:

(5)	=	(8/230)	Tape for action
(6)	=	(3/221)	Number of decks
(7)	=	(1/202)	Program scan
(8)	=	(68/400)	Record of engage tapes
(9)	=	(5/221)	Deck allocation directory
(10)	=	(1/217)	Tape exit to supervisor
(11)	=	(2/206)	Enter SER to queue
(12)	=	(5/214)	Base of SER queues
(13)	=	(3/213)	Halt positions in SER queues
(14)	=0.4	(53/421)	Expected Block Address
(15)	=	(1/482)	Entry to read title

Connections with other routines:

Entered at (1) via co-ordinator from entry in SER queue planted by the one second routine. B registers irrelevant.

Exit: to program scan (1/202) with action record (8/230) zero. The routine is never halted once entered.

Subroutines:

a) "Enter SER to queue"

Entered at (2/206) with
B107 = 1.0
B108 = Deck number(digits 7-3)
B109 = Entry to new SER (1/482)
B110 = Return address
B126 = odd

Return with B101, 108 unaltered

b) "Tape exit to Supervisor control"
Entered at (1/217) with
B109 = Deck number (digits 7-3)
B110 = Return address

Exit to return address, B registers irrelevant.

Temporary Working Space: B100-110 Bt

R480: continued

Notes:

1. For each deck requiring attention, an entry is made to the tape SER queue as follows: In all cases the information preserved is the deck number.

   a) Tape engage normally: SER 1/482 to read and check title.

   b) Tape engaged specially (e.g. for addressing testing or re-engaged after a fault): the SER recorded in the halted tape queue earlier via R216 is brough to the active part of the queue. This condition is indicated by 1 in digit 1 of the deck allocation directory. Digit 1 is reset to zero by R480.

   c) Tape disengaged following computer disengage. No action is taken. This condition is indicated by 1 in digit 11 of the deck allocation directory. Digit 11 is reset to zero by R480.

2. The above actions observed the limit of entries to the tape SER queue, one per deck and two per channel.

3. The record of tapes requiring action is reset to zero on exit.

4. The routine requires modification to deal with more than 16 decks, numbered 0 to 15.

R490: Fixed store tape organisation extracodes

Purpose: A fixed store extracode program entered from extracode jump table of extracodes 1007 to 1024 inclusive to enter relevant programs in main store, "in supervisor".

Registers of fixed store: 22

Instructions obeyed: 3 to 5 for each extracode before entry to R247.

Parameters used: (1) to (22)

Cross references:

(14)	=	(1/247)	Program load B
(15)	=	(3/247)	Prepare load store
(16)	=	(1/492)	Main store tape organisation
(17)	=	(1/498)	Exit for mount
(18)	=	(2/498)	Exit for mount free
(19)	=	(3/498)	Exit for mount next reel
(20)	=	(5/499)	Exit for accept - not used
(21)	=	(1/496)	Exit for release
(22)	=	(4/492)	Exit for rename

Connections with other routines:

All entries are direct from the extracode jump table. Exit is to 1/247 or 3/247 with B91, B92 as shown below. B121, 119 are unaltered. This causes exit to the address in B91, in supervisor with full recovery switch set

E Code	Entry to R491	Exit to (1) or (3) of 247	B91 on exit	B92 on exit
1007	(3)	3	(3/498)	-
1010	(1)	3	(1/498)	-
1011	(2)	3	(2/498)	-
1012	(1)	3	(3/498)	-
1013	(2)	3	(5/499)	-
1014	(5)	3	(1/492)	0.1
1015	(6)	3	(1/492)	0.4
1016	(7)	1	(1/492)	2.0
1017	(8)	1	(1/492)	2.5
1020	(9)	1	(1/492)	3.0
1021	(10)	1	(1/496)	-
1022	(11)	1	(4/492)	-
1023	(12)	3	(1/492)	1.1
1024	(13)	3	(1/492)	1.5

Temporary Working Space: Nil

Notes: The value of B92 where appropriate is carried over via R247 to the SER which starts in the address specified in B91.

R492: Main store tape organisational extracodes

Purpose: An SER in main store entered from fixed store R491 by extracodes referring to tape B. Finds the actual deck number involved and enters various routines to obey specific extracodes. Includes within itself the extracodes "Re-allocate" and "Tape length".

Registers of main store: 42

Instructions obeyed: Most extracodes: 6 + entry to R221 to find deck number.

Extracode "Length": 13 + entry to R221

Extracode "Re-allocate": 14 + entry to R221

Parameters used: (1) to (20)

Cross references:

(5)	=	(1/221)	Find deck number
(6)	=	(9/205)	Current program number in store control
(7)	=	(3/221)	Number of decks
(8)	=	(5/221)	Deck allocation directory
(9)	=	(9/230)	Deck timer directory
(10)	=	(5/201)	SER re-entry
(11)	=	(1/215)	Set full recovery switch
(12)	=	(99/900)	Extracode working space
(13)	=	(4/247)	Return to main program
(15)	=	(1/497)	"Where am I"? extracode
(16)	=	(2/494)	Write title
(17)	=	(1/494)	Read title
(18)	=	(2/495)	Unload
(19)	=	(1/495)	Free
(20)	=	(3/495)	Release tape

Connections with other routines:

Entered at 1) From R491 for extracodes 1014 - 1020, 1023 1024 with B92 as described for R491, full recovery switch set.

Exit: To monitor via 15(1/221) if deck not defined
To halt program via 8(1/221) if deck not available
Otherwise to the relevant routine as listed above with full recovery switch set, extracode B lines and working space unaltered, and re-entry address set to the start of the relevant routine.
B100 = absolute deck number (digits 7 - 3)

Entered at 4) For re-allocate via R491
B119 = new label, digits 9 - 3 rest irrelevant
ba = old label, digits 9 - 3 rest irrelevant

Exit: To (4/247) to return to main program with B119 unaltered.
B121 = 0

R492: continued

Re-entered at (3) for extracodes "Length of tape"

Exit: (4/247) to return to main program with

B91 = B92 = 0
(99/900) = Length of tape, digits 15-3
rest zero.

Subroutines:

a) "Set full recovery switch":
Entered at (1/215) with B109 = Address of SER to deal with specific extracodes.
Exit: to address in B109. B110 unaltered.

b) "Find deck number":
Entered at (1/221) to find Atlas deck
B109 = Return address
B100 = Programmers label, digits 8-2
digits 0 = 0

Exit: To return address with B100 = deck number
digits 7 - 3
or to monitor
or to halt program: Re-enter at (1/492) if deck not available.

Entered at 15(1/221) if Atlas or Orion deck not found,
to monitor

Entered at 8(1/221) if Atlas or Orion deck found,
B107 = deck number, digits 7 - 3
B106 = contents of deck allocation directory
digit 0.
B109 = return address

Exit: To return address if deck available,
B100 = deck label
To halt program if deck not available
Note that the full recovery switch is reset
on exit to return address

Temporary working space: Entry 1) B100, 106-109
Entry 4) B100, 101, 121

Notes:

1. On entry (1) if B92 is odd, R221 is entered to locate an Atlas tape (monitor if Orion tape). If B92 is even, a search is made for an Atlas or Orion tape of the correct label.

2. The program is halted via R221 if the tape referred to is not available (e.g. being mounted, under supervisor control, etc.).

R493: Tape message printer

Purpose: An SER subroutine in main store to print messages to the tape operator. Alternative entry conditions allow for printing the title of a tape in addition.

Registers of main store: 44

Instructions obeyed: 25 if message only; maximum 39 +2D if title also, where D = number of decks.
Also entries to R240 to assemble output which will dominate the number of instructions obeyed.

Parameters used: (1) to (13)

Cross references:

(5)	=	(1/220)	Reserve output
(6)	=	(2/220)	Free output
(8)	=	(12/213)	SER dump address
(9)	=	(5/201)	SER re-entry address
(10)	=	(1/240)	Print message
(11)	=	(8/494)	Deck title directory
(12)	=	(3/240)	Print layout

Connections with other routines:

Entered at 1) with B100 = Deck number (digits 7-3)
B104 = Return address (digits 22-3)
Digit 0 = 1 (print title)
0 (no print title)
B103 = Location in store of message
Digits 22-0 (main store, starting at any character position).
Digit 23 = 0 (use title from title directory)
1 (use title from B105)
B105 = Location of title if other than title directory

Exit : a) To re-entry address if operators output is busy with B102, 101 altered

b) To return address when output is assembled in the buffer with

B100, 103, 104 unaltered
B126 digits 2 - 0 = 0
Re-entry address set to return address,
Digits 2-0 = 0

R493: continued

Subroutines:

a) "Reserve operators output": entered at (1/220) to reserve output channel

```
             B101 = 0.4 (channel 1)
     B100 =  B102 = Deck number
             B110 = Return address

       Exit: To halt program (go back to re-entry) if busy
             To return address, with B100 = working area
             of output, if channel not busy.
```

b) "Free operators output": Entered at (2/220) at conclusion with

```
             B101 = 0.4
             B100 = Deck number
             B110 = Return address

       Exit: To return address with B100, 103, 104 unaltered.
```

c) "Supervisor output": Entered at (1/240) to print message with

```
             B100 = working store of output peripheral
             B108 = 0.1 (message ends on character 00)
                    1.0 (message of two characters)
             B109 = address of message
             B110 = return address digit 0 = 1 to recover
                    B100-104 on exit.
             SER dump address = (7/493) - working space for R493
       Exit after message written to buffer with
             B100-104 preserved
             Re-entry address set as B110 on entry

       Entered at (3/240) to print "New line"
             B100 = working area of peripheral
             B109 = 2.1
             B110 = Return address.  Digit 0=1 as above.
       Exit to return address with B100-104 preserved
       re-entry address set as B110 on entry.
```

Temporary working space: B101, 102, B105-110, Bt
B100 used but reset to original value.

Notes:

1. Printing consists of message followed by deck number on one line, followed optionally by the tape title on a separate line. Message and title are in internal code, inner set, in store, and are terminated by character (octal) 00.

2. A title is only printed if B104 is odd on entry. The title is in the deck title directory for this deck, or in a separate location specified in B105 on entry. If the first half word of title is zero, the title "FREE" is printed.

R493: continued

3. After reserving the output channel, this routine uses a dump area for B100-104 in the event of halts. Only one such area is required, since only one message can be printed at once. If the output channel is busy, the routine calling in R493 is halted, with B100 103, 104 unaltered, and is resumed at the specified re-entry address; if this has digit 1=0, only B100 is preserved on restarting. Usually this is sufficient, as it contains the deck number both on entry to R493 and when the routine is halted by R220.

4. The message and tape title may be in any supervisor main store block bearing a reserved block label in the block directory, which can be called to core store by non-equivalence in supervisor. They must not occupy a supervisor main store block with a non-reserved block label.

R494: Extracodes Read/Write tape title

Purpose: A main store SER to implement the extracode Read title, Write title. An alternative entry provides a subroutine to copy a title from object program store to supervisor store, compressing where necessary.

Registers of main store: 80

Instructions obeyed: Read: Around 10 +2D + 7 per half word of title where D = number of decks.

Write: Around 20 + 2D + 10 per half word + 5 to 20 per character.

Parameters Used: (1) to (12)

(8) = "Deck title directory" 10 words per deck, holding title of tape on deck d in words 10d onwards.

Cross references:

(9)	=	(99/900)	Extracode working space
(10)	=	(4/247)	Exit to main program
(11)	=	(5/221)	Deck allocation directory
(12)	=0.4	(6/201)	Main program controls.

Connections with other Routines

Entry at (1) from R492 for "Read title to store S", in supervisor with full recovery switch set.
B119 = S
B100 = Deck number (digits 7-3)

Exit to (4/247) to reset full recovery switch and exit to main program with
B91 = 0
B92 = number of half words Extracode working space filled less 1 (digits 4-2)
Extracode working space = title or remainder of title.
B97 even
B119 unaltered

Entry at (2) from R492 for "Write title from store S", in supervisor with full recovery switch set
B119 = S
B100 = Deck number (digits 7-3)

Exit to (4/247) to reset full recovery switch and exit to main program with main program controls set to resume in main control.

Entry at 1(1) for subroutine "Find title"
B110 = Return address
B100 = Deck number (digits 7-3)

R494: continued

Exit to return address with
B101 = Location of title of deck in deck title directory (absolute address)
B109 = altered

Entry at (3) for subroutine "Copy title to supervisor store"
B110 = Return address
B101 = Location in supervisor store for title
B119 = Location of the title in the store belonging to the current main program in control of store.

Exit to return address with
B91-97, B100-110, Bt, B119 altered
Title copied and compressed (see notes)
(This subroutine is used by R498, which implements the extracodes "Mount" etc.).

Subroutine:

"Co-ordinate organisational extracode"

Entered with (4/247)

a) To copy to program store

B91 = Re-entry address to R494
B92 = Number of half words extracode working space loaded less 1, digits 8-2 = 3.4
B97 even
Return to address in B91 with B95 unaltered, B119 stepped to next transfer address

b) To read from program store

B91 = Entry address to R494
B92 = 0 (read one half word)
B97 odd
Return to address in B91 with one half word working space filled, B119 stepped by 0.4,B95 unaltered.

Temporary Working Space: B91-97, B100-110, Bt

Notes:

1. The title read from the deck title directory is in internal code characters; up to 79 significant characters are permitted. The last half word of title must contain zero in digits 5-Q.

2. The title read from program store obeys the same rules. If 10 half words have been read and none has zero in digits 5-0, zero is forced to digits 5-0 of the last half word - the title is thus cut short.

R494: continued

3. After reading the title from program store and cutting short where necessary the title is analysed and condensed as follows:-

   a) Characters 03-07, 73-77 are omitted throughout.

   b) Character 02 (Tab) is replaced by 01 (space)

   c) At the start, characters 01, 12, 37 (space, comma, full stop) are ommitted.

   d) Throughout, multiple space characters are ignored (i.e. n spaces equal one space).

   e) The title is ended on character 00.

   (Octal internal code characters are used above).

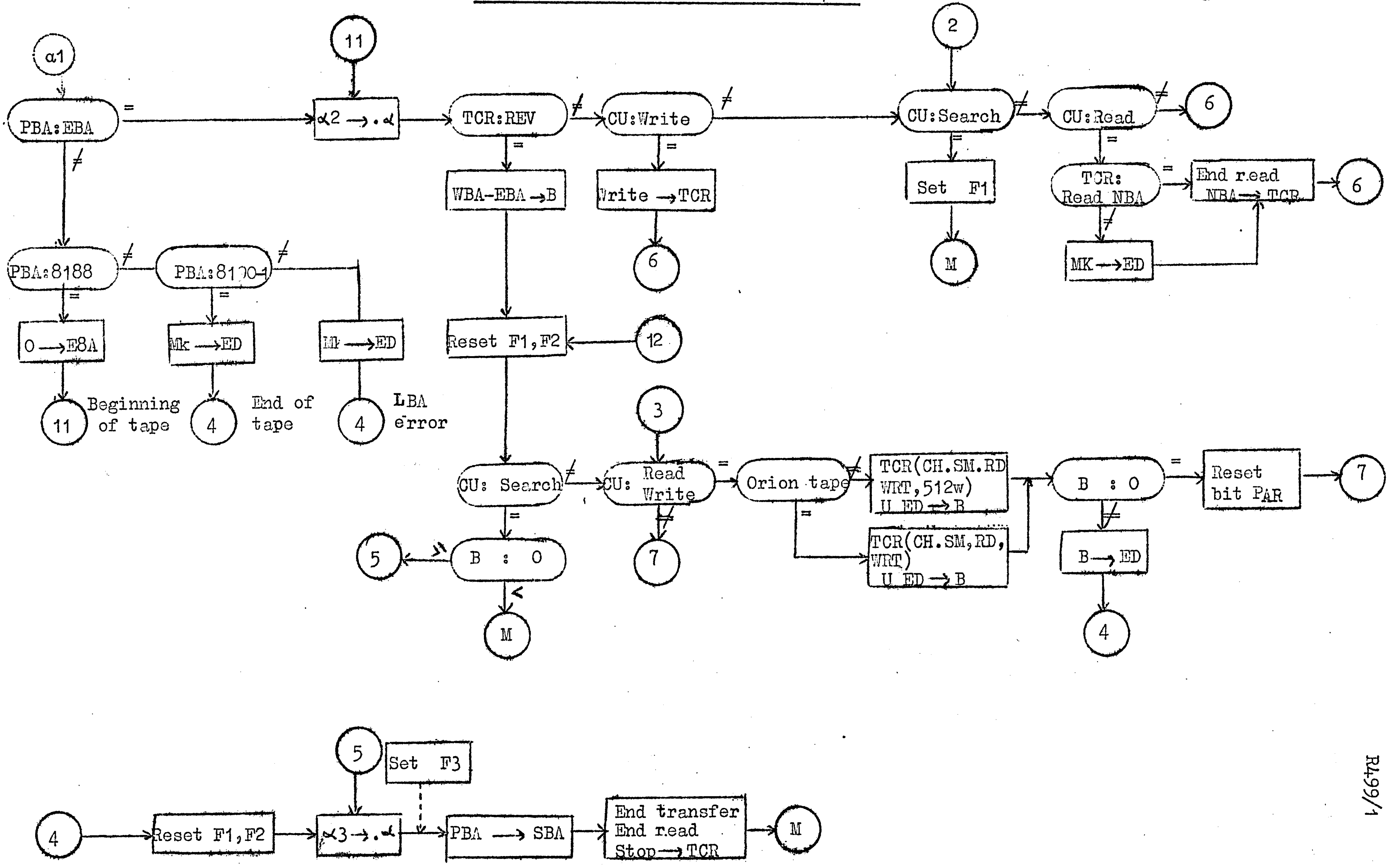
   The last, partially filled, half word is filled by R494 with characters 00.

4. On entry at (2), digit 13 of the deck allocation directory is forced to 1 to indicate that the title has been changed.

5. Although for efficiency this routine would occupy the same block of store as the deck title directory, the routine still functions correctly if these are in separate blocks, and in fact it is convenient elsewhere to use separate blocks for these.

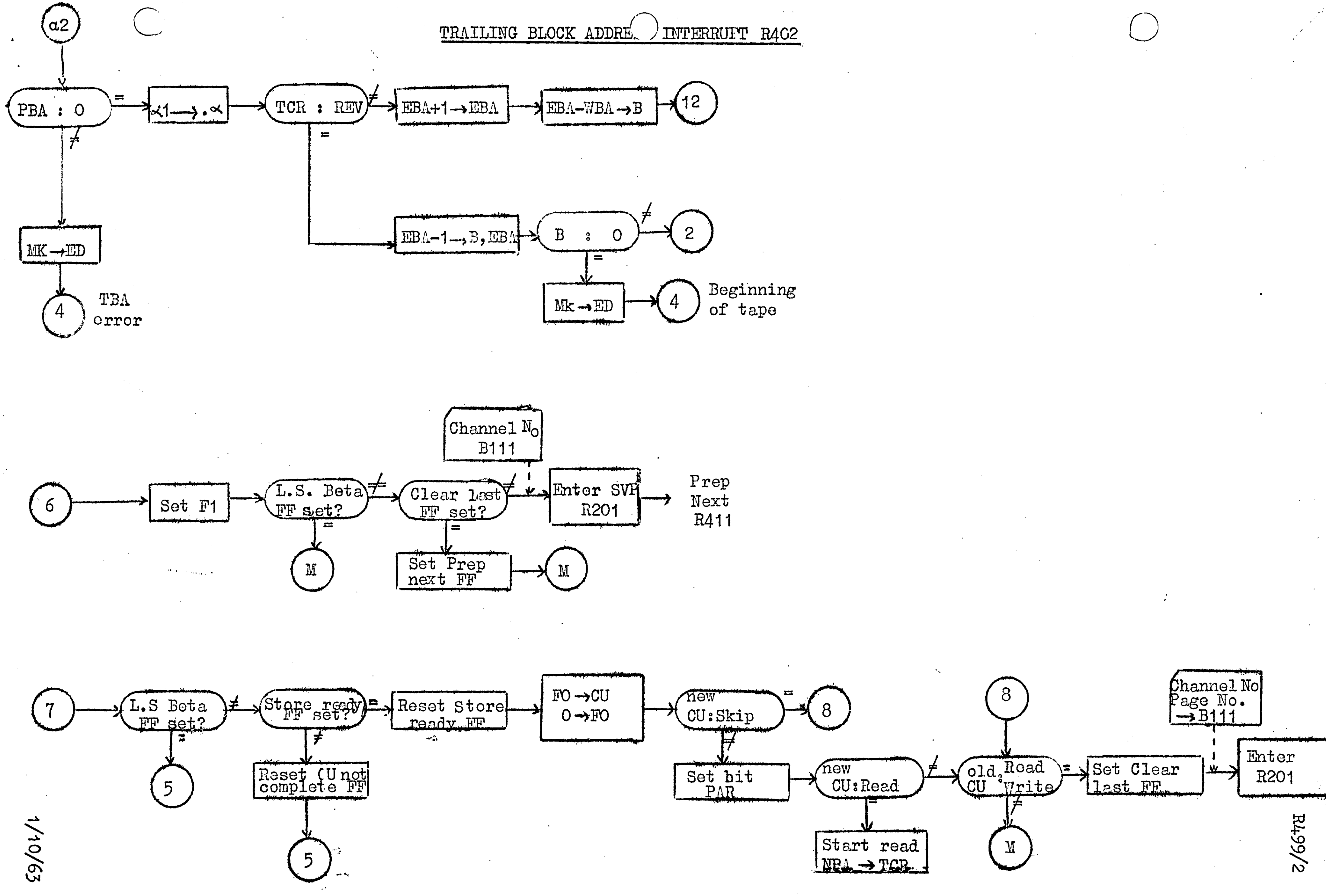
FLOW DIAGRAMS MAGNETIC TAPE ROUTINES

LEADING BLOCK ADDRESS INTERRUPT R402



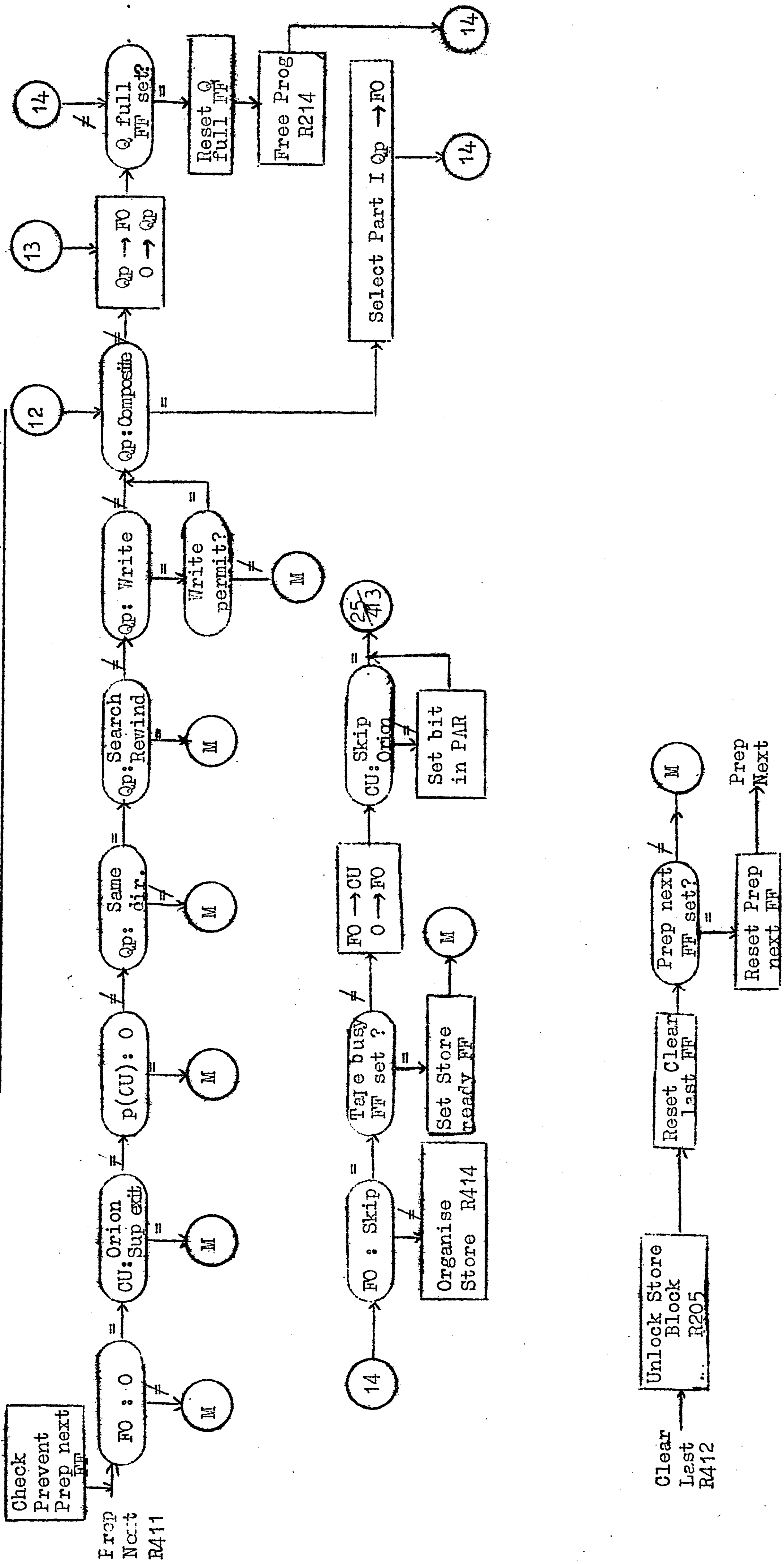
R499/2

TRAILING BLOCK ADDRE INTERRUPT R4C2

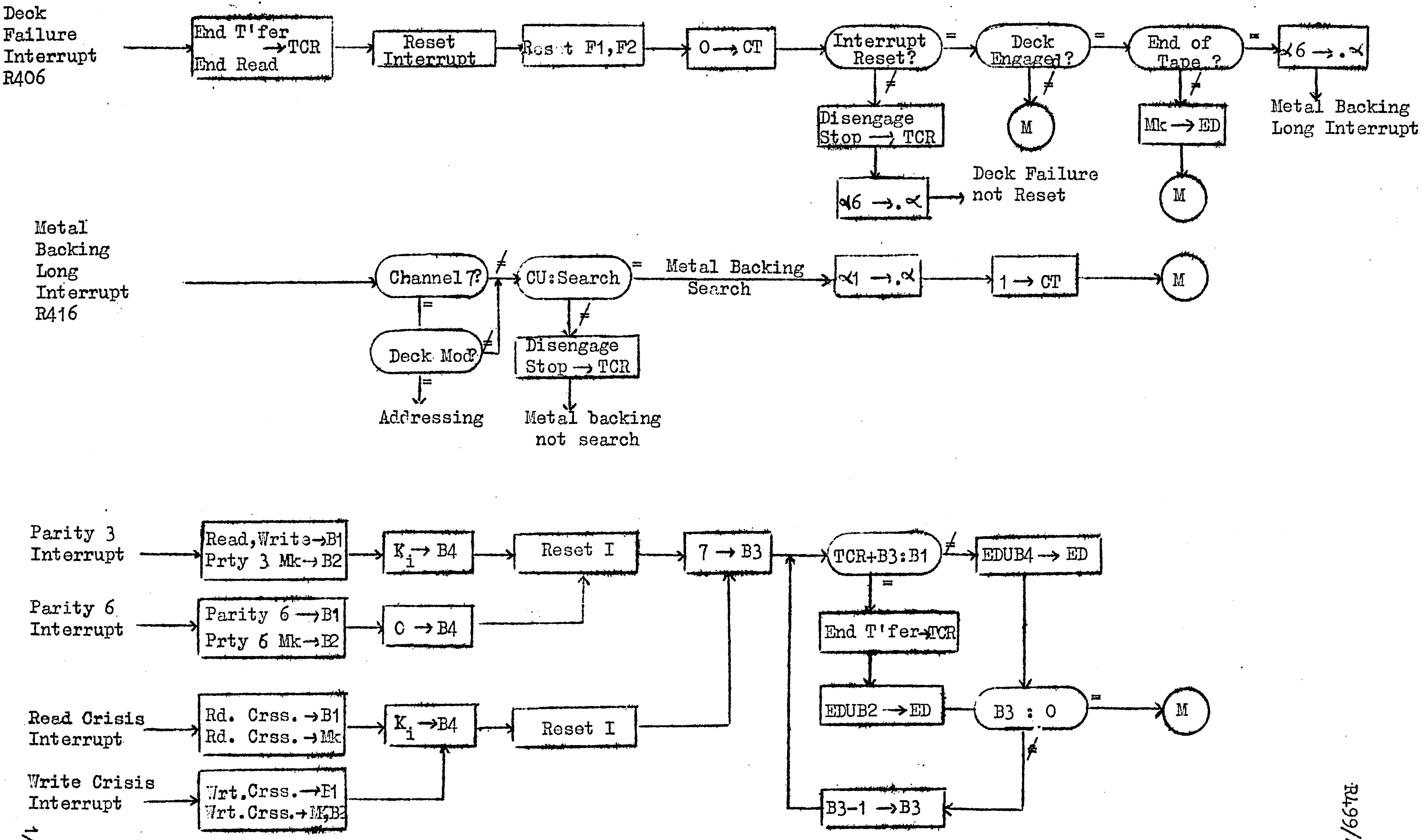


1/10/63

LONG INTERRUPTS PREPARED NEXT AND CLEAR LAST R411 AND R412

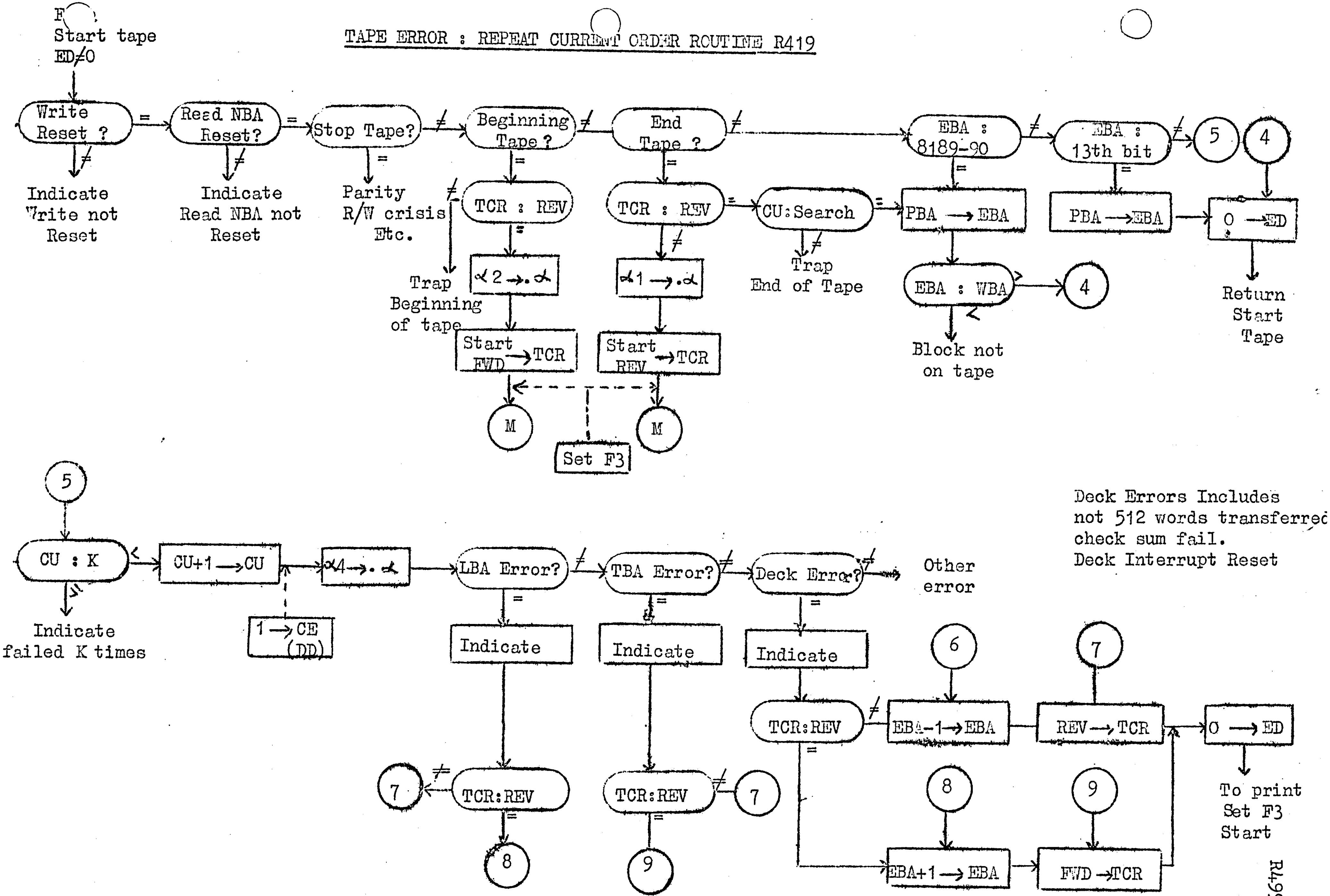


DECK FAILURE INTERRUPT, PARITY 3 & 6 and READ & WRITE CRISIS INTERRUPT R406, R407



1/10/63

TAPE ERROR : REPEAT CURRENT ORDER ROUTINE R419

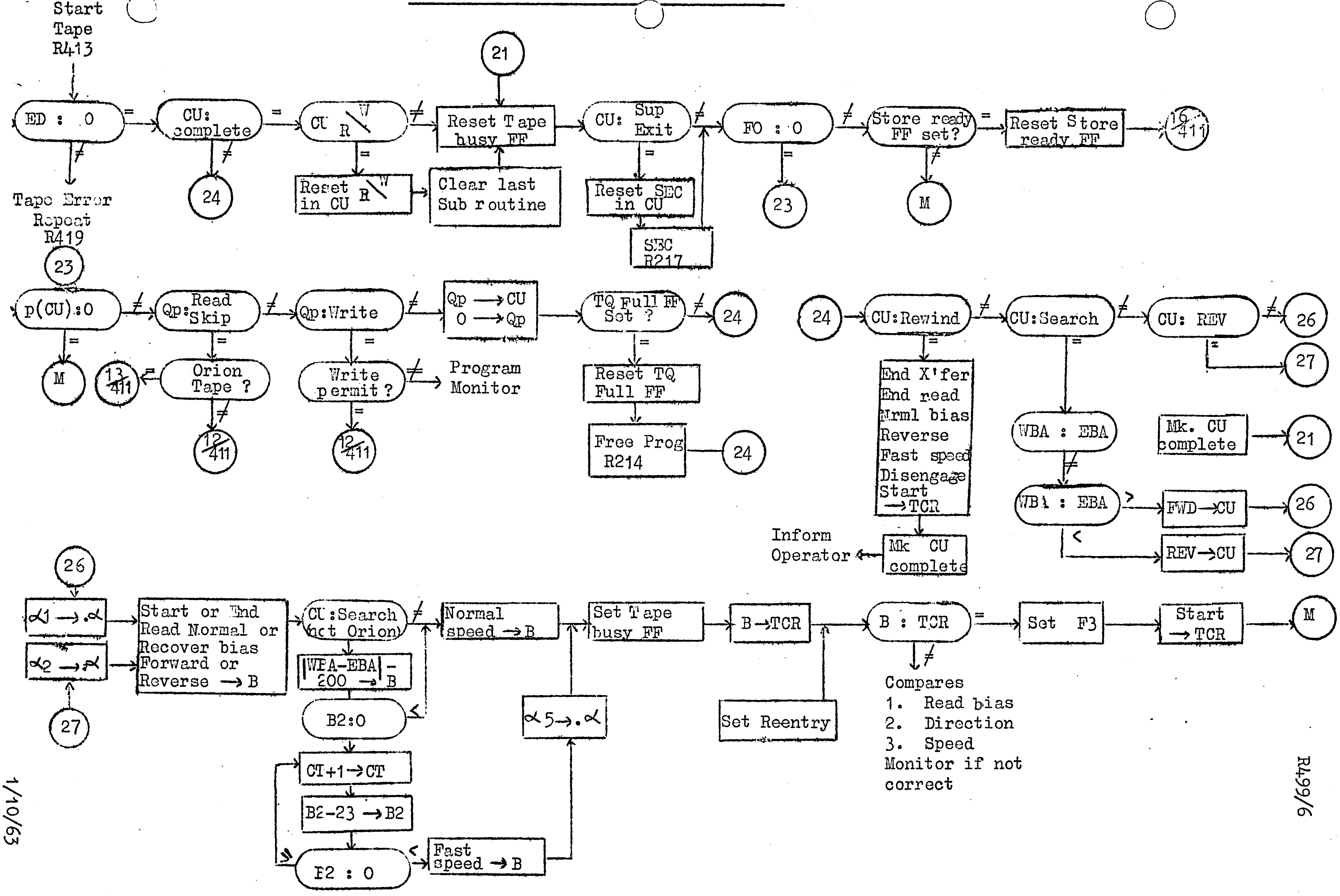


Deck Errors Includes
not 512 words transferred
check sum fail.
Deck Interrupt Reset

R499/5

1/10/63

LONG INTERRUPT START TAPE ROUTINE R413

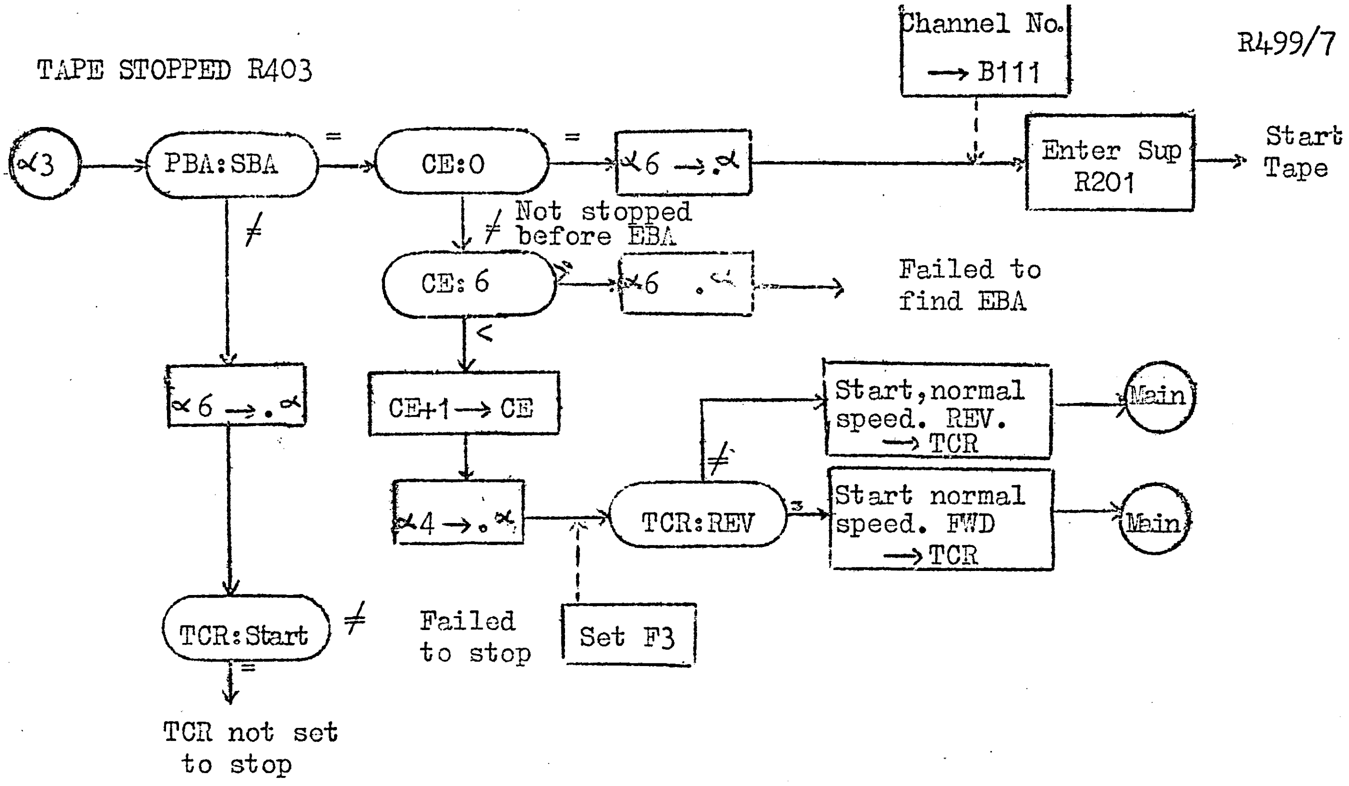


R499/6

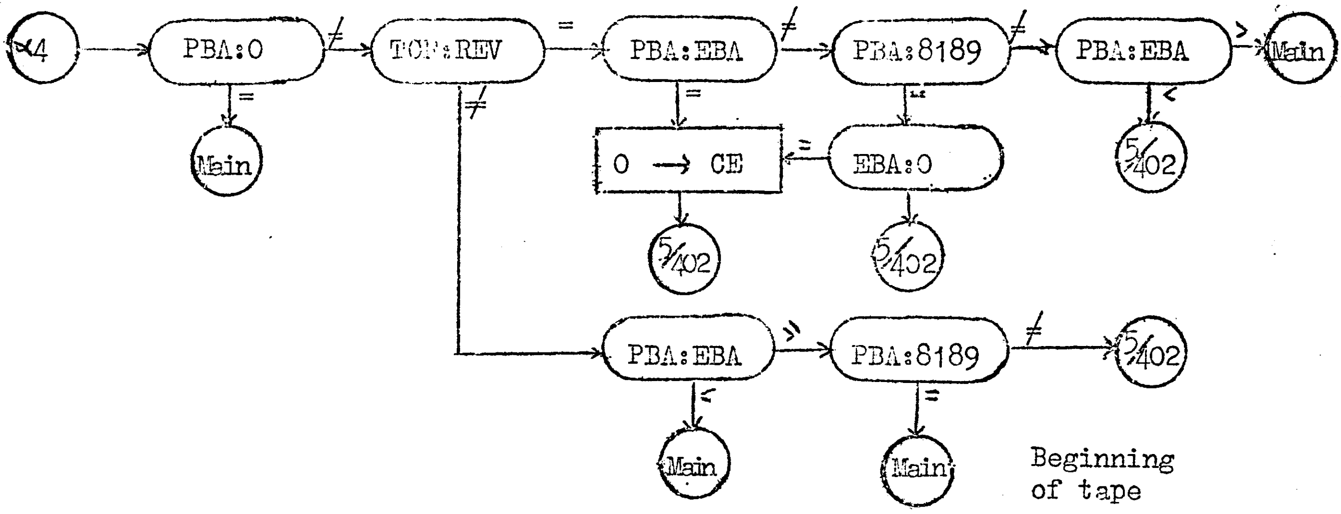
1/10/63

TAPE STOPPED INTERRUPT AND ALIGNMENT AND CALCULATION EBA INTERRUPTS R403, R405

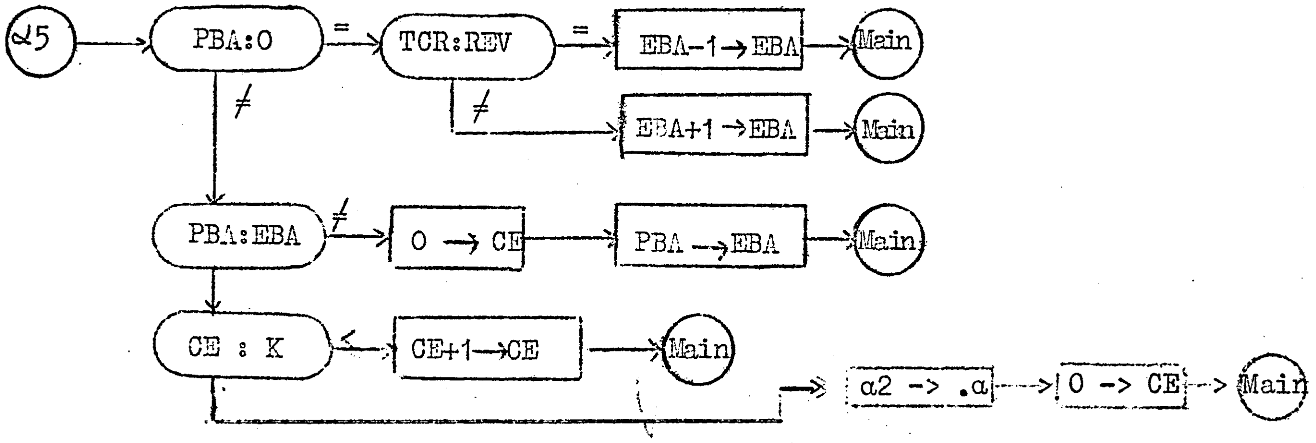
TAPE STOPPED R403



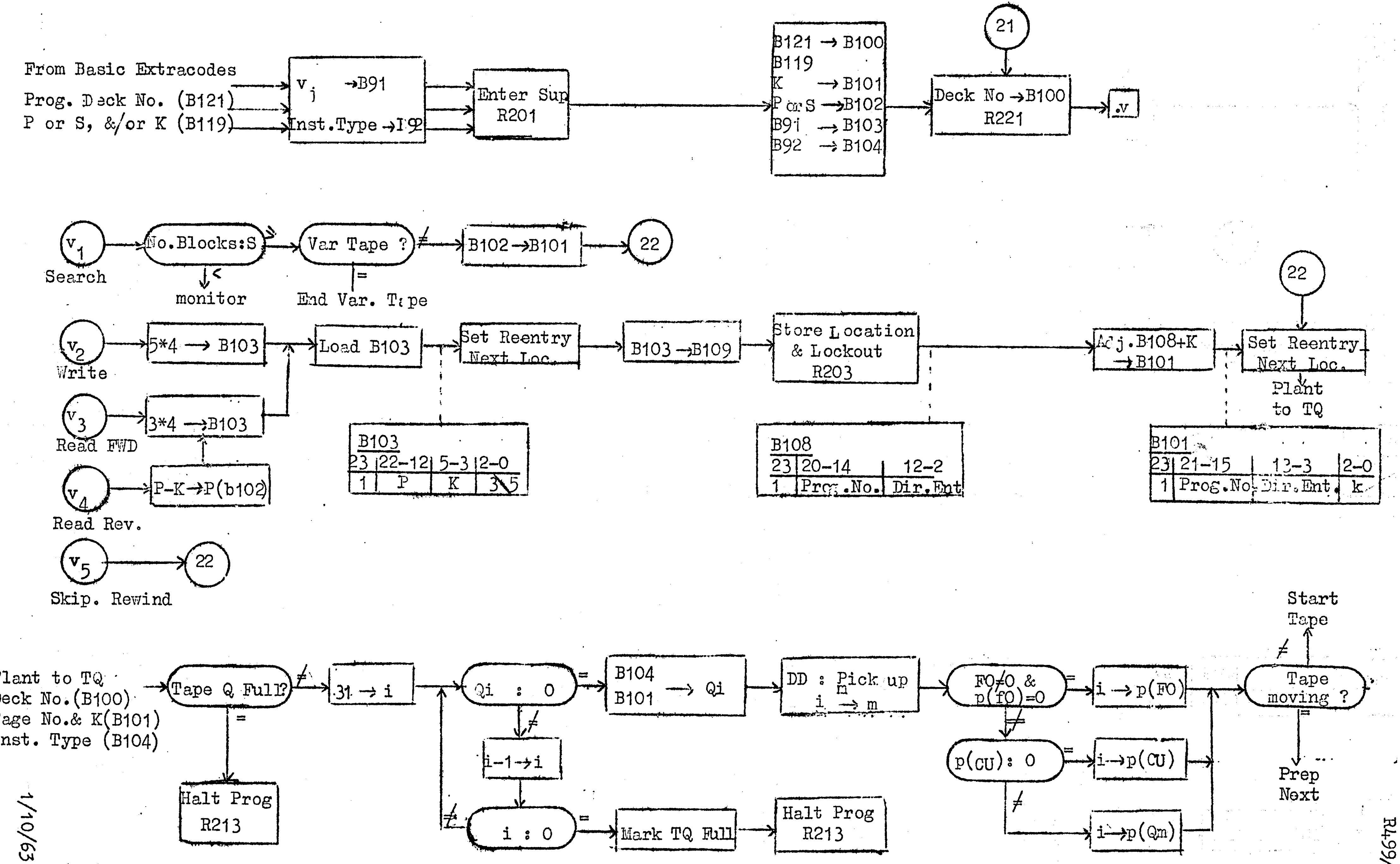
ALIGNMENT OF TAPE TO EBA AFTER AN R404



CALCULATION OF EBA AFTER AN OFF-CHANNEL SEARCH R405



BASIC TAPE ORDER TO TAPE QUEUE R421



```
          T
          FIXED STORE COLUMN 4004

          (0)=*4004

                                                                      |500

          R500                          |SORT INTERRUPTS

1)        101,      123,     0,         2*6            |Line 2
          101,      125,     123,       (2)
4)        101,      123,     0,         0*6            |Line 0
          101,      125,     123,       (12)

                                                       |Line 0        P2
12)       1(4)       /       1(7)                      | -            ICT printer
       254*4003      /       1(7)                      |≠ on I        -
          (6/407)    /       1(7)                      |Parity 6 MTB  Graphical output
          (2/241)    /       1(7)                      |Parity 5 FS   -
          (2/241)    /       (1/541)                   |Parity 4 SS   TR7
          (3/407)    /       1(7)                      |Parity 3 MFT  -
       0.1(2/340)    /       1(7)                      |Parity 2 D    -
          (2/241)    /       (1/532)                   |Parity 1 CS   Card reader EOC
          1(4)       /       (1)                       |-             Wink

7)        101,      123,     0,         31*600434      |P2
          101,      125,     123,       0.4(12)
8)        101,      123,     0,         30*600434      |P3
          101,      125,     123,       (23)

                                                       |P3            P4
23)       (1/571)    /       (1/229)                   |Teletypes     Clocks
          1(8)       /       (1/586)                   | -            Teleprinters
          1(8)       /       1(9)                      | -            -
          (1/566)    /       1(9)                      |TR5's         -
          1(8)       /       1(9)                      | -            -
          1(8)       /       (1/551)                   |Anelex        -
          (1/561)    /       (1/576)                   |Creed 3000    Card punch
          1(8)       /       1(9)                      | -            -
          (1)        /       (1)                       |Wink          Wink

9)        101,      123,     0,         29*600434      |P4
          101,      125,     123,       0.4(23)
11)       101,      123,     0,         1*6            |Line 1
          101,      125,     123,       0.4(31)

                                                       |Powers of 2   Line 1
31)       0.1        /       (1/700)                   | 0            Unassigned function
          0.2        /       (1/700)                   | 1            DO
          0.4        /       (1/700)                   | 2            SVI
          1          /       (1/700)                   | 3            SVO
          2          /       (1/242)                   | 4            ≠ tapes and drums
          4          /       (1/700)                   | 5            EO
          8          /       (1/406)                   | 6            Mag tape deck failure
          16         /       (1/322)                   | 7            Drums
          32         /       (1)                       | 8            -
          101,      126,     0,         (72/595)

                                                       |23/\\10/63.
```

```
        R500                                         |SORT INTERRUPTS

1)      101,    123,    0,      2*6                  |Line 2
        101,    125,    123,    (2)
4)      101,    123,    0,      0*6                  |Line 0
        101,    125,    123,    (12)

                                                     |Line 0              P2
12)     1(4)        /   1(7)                         | -                  ICT printer
     33*40044       /   1(7)                         |≠ on I              -
        (6/407)     /   1(7)                         |Parity 6 MTB        Graphical output
        (2/241)     /   1(7)                         |Parity 5 FS         -
        (2/241)     /   (1/541)                      |Parity 4 SS         TR7
        (3/407)     /   1(7)                         |Parity 3 MTT        -
      0.1(2/340)    /   1(7)                         |Parity 2 D          -
        (2/241)     /   (1/532)                      |Parity 1 CS         Card reader EOC
        1(4)        /   (1)                          |-                   Wink

7)      101,    123,    0,      31*600434            |P2
        101,    125,    123,    0.4(12)
8)      101,    123,    0,      30*600434            |P3
        101,    125,    123,    (23)

                                                     |P3                  P4
23)     (1/571)     /   (1/229)                      |Teletypes           Clocks
        1(8)        /   (1/586)                      | -                  Teleprinters
        1(8)        /   1(9)                         | -                  -
        (1/566)     /   (1/551)                      |TR5'Anelex (MANCHESTER has 1(9))
        1(8)        /   1(9)                         | -                  -
        1(8)        /   1(9)                         | -                - (MANCHESTER has (1/551))
        (1/561)     /   (1/576)                      |Creed 3000          Card punch
        1(8)        /   1(9)                         | -                  -
        (1)         /   (1)                          |Wink                Wink

9)      101,    123,    0,      29*600434            |P4
        101,    125,    123,    0.4(23)
11)     101,    123,    0,      1*6                  |Line 1
        101,    125,    123,    0.4(31)

                                                     |Powers of 2         Line 1
31)     0.1         /   (1/700)                      | 0                  Unassigned function
        0.2         /   (1/700)                      | 1                  DO
        0.4         /   (1/700)                      | 2                  SVI
        1           /   (1/700)                      | 3                  SVO
        2           /   (1/242)                      | 4                  ≠ tapes and drums
        4           /   (1/700)                      | 5                  EO
        8           /   (1/406)                      | 6                  Magtape deck failure
        16          /   (1/322)                      | 7                  Drums
        32          /   (1)                          | 8                  -
        101,    126,    0,      (72/595)
        R708                                        | Acquire blocks for compiler

        (2) = *001                                  |4 excess blocks
        (3) = (4/203)                               |Store directory
        (4) = (5/201)                               |Re-entry address
        (6) = (1/215)                               |Set fr switch
        (5) = (21/261)                              |Alt. entry to End Compiling
        (7) = (4/247)                               |Reset fr switch and exit

1)      121,    100,    0,      (7)                 |Link for main store
        101,    102,    106,    (3)
        124,    102,    0,      (2)                 |Step block counter
        113,    126,    0,      (4)
        121,    109,    0,      (5)
        121,    126,    0,      (6)                 |Exit setting full recovery switch
```

|18/1/63

```
      R501                                                  [Load private store of any peripheral

      (66)=(66/599)
      (50)=(50/599)
      (51)=(51/599)
      (52)=(52/599)
      (53)=(53/599)
      (56)=(56/599)
      (60)=(60/599)
      (61)=(61/599)
      (62)=(62/599)
      (67)=(67/599)
      (68)=(68/599)
      (8)=*5005 5301
      (9)=(5/599)                                           [Beginning of peripheral
                                                            [          subsidiary store table

1)    165,     108,     109,     *0000 3700                 [Address of number 0
      163,     108,     0,       0
      163,     108,     0,       0
      163,     108,     0,       0
      101,     107,     108,     (10)                       [Subsidiary store address
                                                            [         of number 0
      121,     106,     0,       (8)
      113,     106,     107,     8*7                        [Private store of 'number 8'
      164,     107,     109,     7.0
      13,      100,     107,     *7                         [Private store of selected
                                                            [         peripheral
      165,     107,     109,     *0000 3770
      124,     107,     0,       *0004 0000
      113,     107,     100,     (68)*7                     [V-store address (less *6)

2)    165,     109,     101,     *7777 7774
      122,     109,     0,       *7
      113,     109,     100,     (60)*7                     [Beginning of buffer
      165,     109,     102,     *7777 7774
      122,     109,     0,       *7
      113,     109,     100,     (61)*7                     [End of buffer
3)    101,     109,     100,     (60)*7                     [(1/504)
      113,     109,     100,     (62)*7
      113,     109,     100,     (67)*7
      121,     109,     0,       -0.4
      113,     109,     100,     (51)*7                     [M = -0.4
      113,     0,       100,     (52)*7
      113,     0,       100,     (53)*7
      113,     0,       100,     (66)*7
      113,     0,       100,     (50)*7
      113,     0,       100,     (56)*7

      101,     108,     100,     (68)*7                     [Find peripheral type
      127,     108,     0,       *0000 3700
      163,     108,     0,       0
      163,     108,     0,       0
      163,     108,     0,       0
      101,     126,     108,     0.4(10)                    [Jump, depending on peripheral
```

2.9.63

```
                                                        [R501.1

4)       121,     109,     0,       0.1                 [TR5 exit
         113,     109,     100,     (50)*7
7)       113,     0,       100,     (51)*7              [Card reader exit
5)       121,     126,     110,     0

6)       121,     109,     0,       60.0                [Anelex exit
         113,     109,     100,     (61)*7
         121,     109,     0,       *5
         113,     109,     100,     (62)*7
         113,     0,       100,     (60)*7
         113,     0,       100,     (67)*7              [Card punch exit
         121,     126,     110,     0

10)      10.0(9)*1         /        (7)                 [Card reader
         (8)               /        (5)
         (8)               /        (5)                 [Xeronic printer
         (8)               /        (5)
         9.0(9)*1          /        (4)                 [TR7
         (8)               /        (5)
         7.4(9)*1          /        (5)                 [Graphical output
         (8)               /        (5)

         13.4(9)*1         /        (6)                 [Anelex printer
         (8)               /        (5)
         (8)               /        (5)                 [I.B.M. tape
         (8)               /        (5)
         9.4(9)*1          /        (5)                 [Creed 3000
         (8)               /        (5)
         0.0(9)*1          /        (4)                 [TR5
         (8)               /        (5)

         0.4(9)*1          /        (5)                 [Teletypes 0 - 7
         4.4(9)*1          /        (5)                 [Teletypes 8 - 11
         10.4(9)*1         /        5(6)                [Card punch
         (8)               /        (5)
         (8)               /        (5)                 [Spare
         (8)               /        (5)
         13.0(9)*1         /        (5)                 [Teleprinter
         (8)               /        (5)

         (8)               /        (5)                 [(Clock)
         (8)               /        (5)
         (8)               /        (5)                 [Spare
         (8)               /        (5)
         (8)               /        (5)                 [(LAM's)
         (8)               /        (5)
         (8)               /        (5)                 [(LAM's)
         (8)               /        (5)
```

```
                                                        [R502

       R502                                             [Start reading from any input periph.

       (51)=(51/599)
       (52)=(52/599)
       (53)=(53/599)
       (56)=(56/599)
       (64)=(64/599)
       (65)=(65/599)
       (66)=(66/599)


       (30)=*3667                                       [Reserved block number.
       (20)=0.1                                         [Information for punching fault.
       (21)=0

  2)   101,     105,     100,     (52)*7
       127,     105,     0,       *7777 7770
       122,     105,     103,     0                     [If no code change retain previous
       214,     126,     105,     (6)                   [       shift
       113,     103,     100,     (52)*7
  6)   101,     105,     100,     (53)*7
       164,     104,     105,     *7777 7774            [Preserve count of previous faults
       113,     104,     100,     (53)*7

  3)   127,     101,     0,       *7777 7774            [Ignore l.s. bits.
       113,     101,     100,     (64)*7                [Starting address.

  4)   165,     103,     102,     *7777 0000
       216,     103,     103,     (30)
       127,     102,     0,       *0000 7777
       121,     104,     102,     *7777 0003            [Ensure there is one half
       216,     102,     104,     *0000 7774            [       word at end
       124,     102,     103,     *1                    [Reserved block number (less *7)
       113,     102,     100,     (65)*7
  5)   101,     102,     100,     (66)*7
       214,     126,     102,     (7)                   [Jump if last exit was with
       121,     102,     0,       -0.4                  [          main store full
       113,     102,     100,     (51)*7                [Otherwise, set M = -0.4

  7)   113,     110,     100,     (66)*7
       113,     126,     0,       (5/201)
       121,     110,     0,       (8)
       121,     126,     0,       (1/509)               [Find peripheral type.

  8)   101,     126,     109,     (9)                   [Enter appropriate P.E.R.

  9)   (1/533)           /        (99/599)              [Card reader   /  Xeronic
       (1/568)           /        (99/599)              [T.R.7         /  Graphical output
       (2/553)           /        (99/599)              [Anelex        /  I.B.M. tape
       (2/573)           /        (1/568)               [Creed 3000    /  T.R.5
       (2/573)           /        (2/579)               [Teletype      /  Card punch
       (99/599)          /        (2/573)               [ -            /  Teleprinter
       (99/599)          /        (99/599)              [ -            /    -
       (99/599)          /        (99/599)              [ -            /    -
```

2.9.63

```
        R503                            [Start writing to any output peripheral

        (50)=(50/599)
        (52)=(52/599)
        (56)=(56/599)
        (64)=(64/599)

7)      101,      102,      100,      (52)*7
        127,      102,      0,        *7777 7770      [If no code change retain
        122,      102,      103,      0               [         previous shift
        214,      126,      102,      (3)
6)      113,      103,      100,      (52)*7          [Set code table
3)      113,      101,      100,      (64)*7
        113,      0,        100,      (56)*7
        113,      0,        100,      (50)*7
2)      121,      126,      0,        (7/502)
```

```
                                   [R504
R504                      [Free any peripheral
(1)=(3/501)
```

2.9.63

```
                                                        [R508
     R508                                               [Peripheral one second subroutine

     (51)    =      (51/599)
     (66)    =      (66/599)
     (68)    =      (68/599)
     (80)    =      (5/599)                             [First of subsidiary store addresses
     (81)    =      (6/599)                             [Number of subsidiary store addresses


1)   121,    101,    0,      (81)                       [Set counter
2)   101,    100,    101,    (80)                       [Address of private store
     210,    126,    100,    (10)                       [Go to next peripheral if out of use
     101,    102,    100,    (68)*7                     [V store address
     101,    110,    102,    *6                         [Read V store
     210,    126,    110,    (10)                       [Go to next peripheral if disengaged

     121,    110,    0,      (3)                        [If engaged
     121,    126,    0,      (1/509)                    [Find peripheral type
3)   121,    110,    0,      0.1
     113,    110,    0,      3*6                        [INHIBIT INTERRUPTIONS.
     103,    103,    100,    (51)*7                     [Minus M.
     217,    126,    103,    (9)                        [Go to next peripheral if M>0.
     121,    104,    103,    -0.4                       [Go to next peripheral if M=-0.4
     214,    126,    104,    (9)
     101,    104,    100,    (66)*7                     [Find if peripheral is free
     101,    126,    109,    (6)                        [Go to peripheral fault testing
                                                        [   routine. Return to (7),(8) or (9)

6)   (1/530)         /       (99/599)                   [Card reader  /  Xeronic
     (1/540)         /       (99/599)                   [T.R.7        /  Graphical output
     (1/550)         /       (99/599)                   [Anelex       /  I.B.M. tape
     (1/560)         /       (1/565)                    [Creed 3000   /  T.R.5
     (1/570)         /       (1/575)                    [Teletype     /  C.P. (MANCHESTER has 197*40174
     (99/599)        /       (1/585)                    [ -           /  Teleprinter
     (99/599)        /       (99/599)                   [ -           /   -
     (99/599)        /       (99/599)                   [ -           /   -

7)   113,    108,    100,    (51)*7                     [Set new M
     113,    0,      0,      3*6                        [PERMIT INTERRUPTIONS
     121,    107,    0,      0
     121,    108,    100,    0
     121,    110,    0,      (10)
     121,    126,    0,      (2/206)                    [Call S.E.R. to queue

8)   113,    108,    100,    (51)*7                     [Set new M
9)   113,    0,      0,      3*6                        [PERMIT INTERRUPTIONS.
10)  202,    126,    101,    (2)                        [Next peripheral
     121,    126,    0,      (1/202)

     (0)  =  197*40174                                  |MANCHESTER ONLY
     172,    102,    0,      *0004 2200                 |Goneometer
     224,    126,    0,      (1/575)
     101,    126,    0,      915*7

     (0) = 2(10)

                                                        |1/9/64
```

```
                                          [R509


        R509                              [Find peripheral type.

        (68)    =     (68/599)

1)      101,    109,    100,    (68)*7
        124,    109,    109,    0
        125,    109,    0,      0
        125,    109,    0,      0
        125,    109,    0,      0
        127,    109,    0,      7.6
        121,    126,    110,    0
```

2.9.63

```
        R511                                 [Find store length available

        (56)     = (56/599)
        (65)     = (65/599)

1)      101,     105,     100,     (65)*7
        101,     106,     100,     (56)*7
        127,     106,     0,       *7777 7774
        215,     126,     106,     (3)           [Jump if already started on record

2)      122,     105,     101,     0.4           [Test space for next separator
        113,     0,       101,     *7            [Clear space for separator
        217,     126,     105,     (4)           [If no more space, exit
        114,     101,     100,     (56)*7        [Address of separator
        124,     101,     0,       0.4           [Address of next character
        121,     126,     0,       (4)

3)      122,     105,     101,     0             [Length remaining

4)      121,     126,     110,     0
```

2.9.63

```
                  [R512

        R512      [Shift up character in half word

2)      165,      106,      107,      *0303 0303     [Alternative entry
        125,      106,      0,        0              [2L.S. BITS of character up 6 places
        164,      106,      107,      *7474 7474
        121,      107,      106,      0
1)      165,      106,      110,      0.3            [Usual Entry
        214,      126,      106,      (3)
4)      125,      107,      0,        0
        122,      106,      0,        0.1
        215,      126,      106,      (4)
        165,      106,      110,      0.3
        124,      106,      106,      0
        124,      106,      106,      0
        120,      106,      0,        2.0
3)      121,      126,      110,      0
```

2.9.63

R513 [Restore character positions in half word.

```
1)      214,    126,    108,    (4)
  2)    214,    126,    108,    (5)
        120,    108,    0,      0.4
3)      125,    107,    0,      0
        122,    108,    0,      0.1
        215,    126,    108,    (3)
5)      121,    126,    110,    0

4)      121,    126,    110,    1.0
```

2.9.63

```
                                                                [R514

                R514                                            [Return to master routine from P.E.R.

        (66)    =       (66/599)
        (51)    =       (51/599)


1)      121,    101,    0,      0
2)      101,    110,    100,    (66)*7                          [Return address
        113,    101,    100,    (66)*7                          [Preserve reason for stopping
        113,    110,    0,      (5/201)
        121,    126,    110,    0


3)      101,    101,    100,    (51)*7
        121,    126,    0,      (2)
```

2.9.63

```
                                                  [R515

      R515                                        [Start any peripheral

      (51)     =      (51/599)
      (60)     =      (60/599)
      (62)     =      (62/599)
      (67)     =      (67/599)
      (68)     =      (68/599)


1)    101,     108,   100,     (60)*7
      113,     108,   100,     (62)*7
      113,     108,   100,     (67)*7

2)    113,     0,     100,     (51)*7             [Set M= 0
      101,     108,   100,     (68)*7             [V store address
      121,     107,   0,       0.1
      113,     107,   0,       3*6                [INHIBIT INTERRUPTIONS
      107,     109,   108,     *6                 [Test fault bits
      215,     126,   109,     (4)                [Exit if faulty or disengaged
      121,     107,   0,       1.0
      113,     107,   108,     *6                 [Start
4)    113,     0,     0,       3*6                [Permit interruptions
      121,     126,   0,       (1/202)
```

2.9.63

```
        R516                                            [Set code conversion parameters.

        (52)      =       (52/599)
        (90)      =       (90/599)
        (91)      =       (91/599)

1)      101,      109,    100,      (52)*7              [Address of parameter table.
        101,      108,    109,      0                   [Address of character table.
        113,      108,    0,        (90)

        165,      108,    109,      0.6
        124,      108,    108,      0
        101,      108,    108,      (3)                 [Jump table.
        113,      108,    0,        (91)
        165,      108,    109,      *7777 7770
        121,      126,    110,      0

3)      (20/517)0.0       /         (30/517)0.2
        (20/517)0.4       /         (30/517)0.6
        R517                                            [Character code conversion


        (90)     =      (90/599)                        [Subsidiary store working
        (91)     =      (91/599)                        [      space


1)      101,     109,     103,     *7                   [Pick up A code character

2)      104,     109,     0,       (90)                 [Address of line in table
        101,     109,     109,     0                    [Table look up

3)      165,     110,     109,     *7                   [3 bits from m.s. end of table
        105,     110,     0,       (91)
        101,     126,     110,     0                    [Jump to address contained
                                                        [in one of the registers:
                                                        [0.0(20) to 7.4(20)
                                                        [0,0(30) to 7.4(30)
```

2.9.63

```
                                                    [Last character was in
                                                    [A shift B shift / a shift B shift
20)    (21)    /    (21)                            [Special character
       (26)    /    (26)                            [Appears in both B code shifts
       (22)    /    (27)                            [A shift change character
       (23)    /    (22)                            [a shift  change character
       (26)    /    (26)                            [Any B shift character
       (24)    /    (28)                            [Any b shift character
       (26)    /    (26)                            [B shift change character
       (25)    /    (29)                            [b shift change


21)    101,    126,    108,    4.0

22)    101,    126,    108,    3.4

23)    101,    110,    108,    0.4
       113,    110,    0,      (90)                 [Change to table 'a'.
       121,    110,    0,      0.4(20)              [Expect next character in
       113,    110,    0,      (91)                 [       a shift B shift.
       101,    126,    108,    3.0

24)    121,    110,    0,      0.2(30)
       113,    110,    0,      (91)                 [Change to b shift
       101,    109,    108,    1.4                  [b shift change character
       101,    126,    108,    2.4

25)    121,    110,    0,      0.2(30)
       113,    110,    0,      (91)                 [Change to b shift
26)    101,    126,    108,    2.0

27)    101,    110,    108,    0.0
       113,    110,    0,      (90)                 [Change to table 'A'
       121,    110,    0,      0.0(20)              [Expect next character in
       113,    110,    0,      (91)                 [       A shift B shift
       101,    126,    108,    3.0
28)    121,    110,    0,      0.6(30)
       113,    110,    0,      (91)                 [Change to b shift
       101,    109,    108,    1.4                  [b shift change character
       101,    126,    108,    2.4

29)    121,    110,    0,      0.6(30)
       113,    110,    0,      (91)                 [change to b shift
       101,    126,    108,    2.0
```

```
                                                [Last character was in
                                                [A shift b shift / a shift b shift
30)    (21)    /      (21)                      [Special character
       (26)    /      (26)                      [Appears in both B code shifts
       (22)    /      (37)                      [A shift change character
       (34)    /      (22)                      [a shift change character
       (35)    /      (38)                      [Any B shift character
       (26)    /      (26)                      [Any b shift character
       (36)    /      (39)                      [B shift change character
       (26)    /      (26)                      [b shift change character

34)    101,    110,   108,    0.4
       113,    110,   0,      (90)              [Change to table 'a'
       121,    110,   0,      0.6(30)           [Expect next character in
       113,    110,   0,      (91)              [      a shift b shift
       101,    126,   108,    3.0

35)    121,    110,   0,      0.0(20)
       113,    110,   0,      (91)              [Change to B shift
       101,    109,   108,    1.0               [B shift change character
       101,    126,   108,    2.4

36)    121,    110,   0,      0.0(20)
       113,    110,   0,      (91)              [Change to B shift
       101,    126,   108,    2.0

37)    101,    110,   108,    0.0
       113,    110,   0,      (90)              [Change to table 'A'
       121,    110,   0,      0.2(30)           [Expect next character in
       113,    110,   0,      (91)              [      A shift b shift
       101,    126,   108,    3.0

38)    121,    110,   0,      0.4(20)
       113,    110,   0,      (91)              [Change to B shift
       101,    109,   108,    1.0               [ B shift change character
       101,    126,   108,    2.4

39)    121,    110,   0,      0.4(20)
       113,    110,   0,      (91)              [Change to B shift
       101,    126,   108,    2.0
```

```
                                                  [R518

       R518                                       [Preserve code conversion
                                                  [     parameters.


       (52)      =       (52/599)
       (91)      =       (91/599)

1)     121,      109,    0,       0.6
2)     121,      108,    0,       *7777 7771      [Clear bits 1 and 2.
       117,      108,    100,     (52)*7
       107,      109,    0,       (91)
       114,      109,    100,     (52)*7          [Insert bits 1 and 2.
       121,      125,    0,       (1/516)         [Reset parameters, in case
                                                  [     required later.
```

2.9.63

```
                                                  [R519

        R519                                      [Insert Seperator

        (52)     =      (52/599)
        (56)     =      (56/599)

1)      121,     109,   0,       0                [Enter here if end of record
        121,     108,   0,       0
        121,     126,   0,       (3)

2)      121,     109,   0,       *1               [Enter here if record continues
        121,     108,   0,       0.1
3)      101,     107,   100,     (52)*7
        211,     126,   107,     (4)              [Jump if internal code
        124,     109,   0,       *4               [Bit 23 = 1 if binary
4)      101,     107,   100,     (56)*7
        211,     126,   107,     (5)              [Jump if this was new record
        124,     109,   0,       *2               [Bit 22 = 1 if continuation
5)      127,     107,   0,       *77777774
        124,     109,   101,     0
        122,     109,   107,     0.4              [Add length of record
        113,     109,   107,     *7               [Send to store
        124,     101,   0,       0.3
        127,     101,   0,       *7777 7774       [Round up next address
        113,     0,     101,     *7               [Clear that half word
        113,     108,   100,     (56)*7           [Set (56) = 0 or 0.1
        121,     109,   0,       0.4              [Retain input code shift, but
        210,     109,   108,     0.6              [   return to inner set if
        121,     126,   0,       (2/518)          [   true end of record
```

2.9.63

```
        R520                                        [Set reserved block label


        (7)     =       *0000 1001                  [Information for R318


        (8)     =       *3667                       [New block label
        (64)    =       (64/599)

1)      101,    101,    100,    (64)*7              [Main store address.
        217,    126,    101,    (5)                 [Jump if not main store.
        165,    109,    101,    *3777               [Block number
        124,    109,    0,      (7)
        121,    126,    0,      (1/318)             [Call block to cores


2)      101,    101,    100,    (64)*7              [Main store address.
        217,    126,    101,    (5)                 [Jump if not main store.
        127,    101,    0,      *0000 7777
        124,    101,    0,      (8)*1               [New block address
        121,    102,    109,    0                   [Copy page number to B102
        121,    108,    0,      (8)
        121,    126,    0,      (1/312)             [Set PAR

5)      121,    102,    0,      *4                  [*4 if not main store
        122,    101,    0,      *7                  [Subtract *7
        121,    126,    110,    0
        R521                                                [Pick up record separator

        (50)=(50/599)
        (52)=(52/599)
        (56)=(56/599)
        (99)=(99/599)

1)      124,     101,     0,       0.3
        127,     101,     0,       *7777 7774               [Round up next address
        101,     108,     101,     *7                       [Separator
        165,     107,     108,     *0004                    [If separator has bit 14=1 treat as
        215,     108,     107,     *3                       [ record with zero character count
        165,     107,     108,     *0000 7777
        113,     107,     100,     (56)*7                   [Character count
        214,     126,     108,     (7)                      [Jump if zero separator
        124,     101,     0,       0.4                      [Address of first character
        164,     107,     101,     *0000 7777
        121,     109,     107,     *7777 0003               [Monitor if next separator is
        216,     126,     109,     (99)                     [        not in same block
        164,     107,     101,     *7777 0000
        122,     107,     0,       0.1                      [Address of carriage control ch.

        165,     109,     108,     *2
        215,     126,     109,     (3)                      [Jump if not beginning new record
        121,     109,     0,       *7777 7772               [If beginning, clear bits 2, 0
        117,     109,     100,     (52)*7
        216,     126,     108,     (3)
        121,     109,     0,       0.1                      [If binary insert 0.1
        114,     109,     100,     (52)*7

3)      165,     109,     108,     *1
        215,     126,     109,     (7)                      [Jump if record continues
        121,     109,     0,       0.1                      [If record ends, subtract
        110,     109,     100,     (56)*7                   [        0.1 from character
        121,     109,     110,     0                        [        count
        121,     110,     0,       (6)
        164,     110,     107,     0.3
        101,     107,     107,     *7                       [Shift required characters to
        121,     126,     0,       (1/512)                  [        top end
6)      127,     107,     0,       *7700 0000
        125,     107,     0,       *4                       [Shift to l.s. end and add *4
        113,     107,     100,     (50)*7                   [Control character to (50)
        121,     126,     109,     0                        [Return

7)      113,     0,       100,     (50)*7                   [Clear (50)
        121,     126,     110,     0                        [Return
```

2.9.63

```
                                          [R522

          R522                            [Find peripheral buffer in
                                          [     part page

          (61)      =       (61/599)
          (62)      =       (62/599)
          (67)      =       (67/599)

  1)      101,      104,    100,      (61)*7
          121,      126,    0,        (3)

  2)      101,      104,    100,      (62)*7
  3)      101,      103,    100,      (67)*7
          122,      104,    103,      0
          121,      126,    110,      0
```

2.9.63

[R523

```
      R523                                         [Remove reserved block label.

            (64)     =       (64/599)
            (67)     =       (67/599)

 1)   113,     103,     100,    (67)*7             [Preserve next address in buffer.
 4)   101,     109,     100,    (64)*7             [Old (main) store address
      127,     101,     0,      *00007777          [l.s. bits of next address
      164,     101,     109,    *7777              [Add in block number
      113,     101,     100,    (64)*7             [Preserve

 2)   121,     108,     0,      *4                 [Lock out bit
      121,     109,     102,    0                  [Page number
      216,     126,     102,    (1/312)            [Set PAR
      121,     126,     110,    0                  [Exit if not main store
```

2.9.63

```
        R527                        [Carriage control code conversion

        (50)=(50/599)
        (52)=(52/599)


1)      101,    109,    100,    (50)*7          [Pick up control character.
        215,    126,    109,    (2)
        101,    126,    108,    7.0             [Exit if no control character
2)      127,    109,    0,      7.7             [Shift up 2 places
        124,    109,    109,    0
        124,    109,    109,    0
3)      104,    109,    108,    5.0             [Add address of table.
        101,    109,    109,    0               [Table look up
        101,    107,    100,    (52)*7
        127,    107,    0,      0.2             [Find present output shift
        165,    110,    109,    *2
        217,    126,    109,    (14)            [Jump if character is in
                                                [     second, or both output shifts
        215,    126,    110,    (10)
        101,    126,    108,    6.4             [Exit with special character, or zero


10)     214,    126,    107,    (18)
        101,    109,    108,    1.0             [Insert shift if required
        121,    126,    0,      (15)

14)     215,    126,    110,    (18)            [Jump if either shift will do
        126,    107,    0,      0.2
        214,    126,    107,    (18)
        101,    109,    108,    1.4             [Insert shift if required.
15)     116,    107,    100,    (52)*7          [Alter private store.
        101,    126,    108,    6.0             [Exit with shift character.

18)     165,    110,    109,    *0077           [Bits 17-12
        125,    110,    0,      0
        125,    110,    0,      *4              [Character for next time
        113,    110,    100,    (50)*7          [Store C.Control char: for next time

        101,    126,    108,    5.4             [Exit with character
```

2.9.63

```
                                                       [R530

      R530                                             [Card reader fault test

      (10)      =       (5/724)

1)    101,     106,     102,     *6
      165,     107,     106,     0.2
      214,     126,     107,     (9/508)               [Return if started
      214,     126,     104,     (7)                   [Call input master if reader
                                                       [      is free
      165,     107,     106,     0.4
      214,     126,     107,     (2)
      121,     108,     0,       1.0                   [If disabled set M=1.0
      121,     126,     0,       (4)

2)    165,     107,     106,     *0000 0100
      214,     126,     107,     (3)
      121,     109,     0,       *0000 0103            [Reset overdue and disengage
      121,     108,     0,       2.0                   [Set M = 2.0
      121,     126,     0,       (5)

3)    165,     107,     106,     2.0
      214,     126,     107,     (7)
      121,     108,     0,       4.4                   [If cards low set M = 4.4
4)    121,     109,     0,       0.3                   [Stop and disengage
5)    113,     109,     102,     *6
6)    121,     109,     0,       (1/533)               [Call P.E.R. to queue
      121,     126,     0,       (7/508)

7)    121,     108,     0,       -0.4                  [Set M =  -0.4
8)    121,     109,     0,       (10)                  [Call input master
      121,     126,     0,       (7/508)
```

2.9.63

```
                                                     [R531

        R531                                         [Card reader column interruption

        (50)=(50/599)
        (62)=(62/599)
        (10)=(5/599)10.0
        (99)=(99/599)

1)      101,    123,    0,      *6004 3700
        101,    125,    123,    (2)

2)      (3)     /       0
        (3)     /       0                            [MANCHESTER has (99)
        (99)    /       0
        (99)    /       0
        (99)    /       0
        (99)    /       0
        (99)    /       0
        (99)    /       0
        (1/500) /       0

3)      101,    111,    123,    (10)                 [Find private store
        101,    113,    111,    (62)*7               [Present address
        121,    114,    113,    0.4
        113,    114,    111,    (62)*7               [Next address

        101,    115,    123,    *6004 0000           [Copy card column to
        113,    115,    113,    *7                   [    buffer

        101,    116,    111,    (50)*7
        121,    117,    116,    0.4                  [Count columns
        214,    125,    116,    (5)                  [Jump if first column
        122,    113,    0,      0.4
        101,    116,    113,    *7                   [Read previous character
        106,    116,    123,    8*6004 0000
        127,    116,    0,      *7777                [compare with check station
        214,    125,    116,    (7)
        124,    117,    0,      *0001                [If different count up faults
        121,    125,    0,      (7)

5)      167,    115,    0,      *0000 2000           [If first column, set
        113,    115,    113,    *7                   [   bit 10 = 1
        126,    115,    0,      *0006
        127,    115,    0,      *7777                [Test for end of pack punching
        215,    125,    115,    (7)
        167,    117,    0,      0.1                  [If end, force bit 0 = 1
7)      113,    117,    111,    (50)*7
        121,    112,    0,      0.4
        113,    112,    123,    *6004 0000           [POLAM
        165,    118,    117,    *0000 7774           [Check not too many card
        122,    118,    0,      41.0                 [    columns
        217,    125,    118,    (1/500)
        121,    125,    0,      (2/532)


                                                     [1/9/64
```

```
                                                        [R532
         R532                                           [Card reader E.O.C. interruption

         (50)=(50/599)
         (51)=(51/599)
         (61)=(61/599)
         (62)=(62/599)
         (10)=(10/531)

1)       101,     123,     0,       *6004 3670          [Which card reader
         101,     111,     123,     (10)                [Find private store

         101,     112,     111,     (50)*7
         165,     113,     112,     *7777 7774
         122,     113,     0,       40.4
         214,     125,     113,     (4)                 [Jump provided count
                                                        [    correct
2)       121,     112,     0,       7.0                 [Check failed
         113,     112,     111,     (51)*7              [Set M = 7.0
         121,     112,     0,       2.7
         113,     112,     123,     *6004 0000          [POLAM, Stop, Disengage
         121,     112,     0,       (1/533)
         121,     125,     0,       (4/201)             [Call PER to queue

4)       113,     0,       111,     (50)*7              [Reset count
         121,     113,     0,       6.0
         101,     114,     111,     (62)*7
         122,     114,     0,       0.4
         121,     115,     0,       *0000 4000          [Set end of card marker
         113,     115,     114,     *7
         124,     114,     0,       40.4
         100,     114,     111,     (61)*7
         217,     113,     114,     6.2*4
         121,     114,     0,       -0.4                [Set M = -0.4 if buffer full
         210,     113,     112,     6.3*4
         210,     114,     112,     5.4                 [Set M = 5.4 if ending character
         113,     113,     123,     *6004 0000          [Do not divert, POLAM
         216,     125,     113,     (1/500)             [Exit unless stopping
         113,     114,     111,     (51)*7
         121,     112,     0,       (1/533)
         121,     125,     0,       (4/201)             [Call PER to queue
```

2.9.63

```
                                                    [R533.1

       R533                                         [Card reader PER

       (50)=(50/599)
       (51)=(51/599)
       (53)=(53/599)
       (62)=(62/599)
       (67)=(67/599)
       (68)=(68/599)
       (90)=(90/599)
       (92)=(92/599)


1)     101,     101,     100,     (50)*7
       165,     102,     101,     *0000 7774        [Step back to end of last
       110,     102,     100,     (62)*7            [   good card
       113,     0,       100,     (50)*7            [Clear column count

       121,     110,     0,       (2)
       121,     126,     0,       (1/520)
2)     121,     110,     0,       (3)
       121,     126,     0,       (2/520)           [Block to cores
                                                    [   sets B101, B102
3)     113,     102,     0,       (92)              [Preserve B102 in temporary storage
       121,     110,     0,       (4)
       121,     126,     0,       (2/522)           [Find part page
                                                    [   sets B103, B104
4)     121,     110,     0,       (6)               [Set code conversion parameters
       121,     126,     0,       (1/516)           [   sets B108, B109

6)     121,     110,     0,       (7)
       121,     126,     0,       (1/511)           [Find store length available
                                                    [   sets B105
7)     217,     126,     105,     (36)              [Go to end if exceeded
       124,     105,     105,     0
       124,     105,     105,     0
       101,     107,     101,     *7                [Current half word
       121,     110,     0,       (8)
       164,     110,     101,     0.3               [Shift next available
       121,     126,     0,       (1/512)           [   space to m.s. end
                                                    [   sets B106
8)     210,     126,     109,     (10)              [Jump if binary
       202,     126,     106,     (24)              [Jump and set count if
       121,     106,     0,       1.4               [   in middle of half word
       121,     107,     0,       0                 [Set count and half word
       121,     126,     0,       (24)              [   if at beginning
```

2.9.63

```
                                                   [R533.2

                                                   [BINARY

10)    127,    105,    0,      *7777 7770          [Round down counters
       124,    106,    0,      0.4
       127,    106,    0,      1.0
       203,    126,    106,    (14)                [Jump if in middle of
       121,    106,    0,      1.0                 [   half word
       121,    107,    0,      0
       121,    126,    0,      (14)

12)    101,    109,    103,    *7                  [Pick out character and
       164,    107,    109,    *7777               [    pack into half word
       125,    107,    0,      0
       125,    107,    0,      0
       127,    109,    0,      *0000 4000          [Test for end of card
       215,    126,    109,    (19)
       203,    126,    106,    (13)
       113,    107,    101,    *7                  [If half word full, send
       121,    106,    0,      1.0                 [   to store and reset
       121,    107,    0,      0                   [   count
13)    124,    103,    0,      0.4                 [Advance buffer address
       124,    101,    0,      0.2                 [Advance store address
14)    202,    126,    104,    (16)                [Jump unless buffer empty
       101,    105,    100,    (51)*7              [Find reason for stopping
       121,    126,    0,      (17)

16)    203,    126,    105,    (12)                [Jump unless store full
17)    121,    109,    0,      *5                  [Prepare for binary separator
       121,    126,    0,      (27)

19)    121,    109,    0,      *4                  [Prepare for binary separator
       121,    126,    0,      (31)
```

2.9.63

```
23)     122,    103,    0,      0.4
        124,    104,    0,      0.4

20)     127,    109,    0,      7.7
        125,    107,    109,    0
        202,    126,    106,    (22)          [Jump unless half word is full
        113,    107,    101,    *7            [Send to store
        121,    106,    0,      1.4
        121,    107,    0,      0
22)     124,    101,    0,      0.1           [Advance store address
        124,    103,    0,      0.4           [Advance buffer address
24)     202,    126,    104,    (25)          [Jump provided buffer space
        101,    105,    100,    (51)*7        [If buffer finished, find reason
        121,    126,    0,      (26)          [   for stopping.

25)     202,    126,    105,    (29)          [Jump if store space
26)     121,    109,    0,      *1            [Prepare for internal code separator
27)     165,    108,    101,    0.3
        121,    110,    0,      (28)          [If no store space, write
        121,    126,    0,      (1/513)       [   away last half word
28)     113,    107,    101,    *7
        121,    108,    0,      0.1
        121,    110,    0,      (37)
        216,    126,    105,    (4/519)       [Insert separator
        121,    126,    0,      (1/518)       [If buffer finished, preserve
                                              [   code conversion parameters
29)     101,    102,    103,    *7            [Pick up next character
        101,    126,    0,      (90)          [   from buffer and jump
                                              [   to code conversion
                                              [Return to (20), (23) or (30)

30)     121,    109,    0,      0             [Prepare for internal code separator
31)     125,    107,    0,      2.1           [End of card character
        113,    107,    101,    *7            [Write to store
        124,    101,    0,      0.1
        124,    103,    0,      0.4
        165,    108,    101,    0.3
        121,    110,    0,      (32)
        121,    126,    0,      (1/513)       [Write away last half word
32)     113,    107,    101,    *7
        121,    108,    0,      0
        121,    110,    0,      (33)
        121,    126,    0,      (4/519)       [Insert separator
33)     121,    109,    0,      0             [Next card expected for
        121,    110,    0,      (34)          [   internal code if reading
        121,    126,    0,      (2/518)       [   mixed cards
34)     215,    126,    104,    (6)           [Convert next card
        101,    105,    100,    (51)*7
        121,    126,    0,      (37)

36)     121,    105,    0,      0
37)     101,    102,    0,      (92)
        121,    110,    0,      (38)
        121,    126,    0,      (1/523)       [Remove reserved block label

38)     121,    101,    105,    0
        124,    105,    0,      0.4           [If stopped because of
        215,    126,    105,    (2/514)       [   fault, return to master
        121,    109,    0,      *0000 0125
        121,    126,    0,      (1/515)       [Otherwise start reader
```

```
                                                   [R533.4

                                                   [Harwell code conversion

40)     (41)      /         (48)
        0.5       /         0.4
        (20)      /         (23)
        (48)      /         0

41)     125,      102,      0,        0
        165,      109,      102,      7.4
        125,      102,      0,        0
        217,      126,      102,      (30)             [Exit if end of card
        165,      110,      102,      *0000 0374       [Rows 2 to 7
        125,      102,      0,        0
        164,      109,      102,      *0000 0300       [Rows 8 9 + - 0 1

        101,      110,      110,      (49)
        101,      109,      109,      (49)

        127,      109,      0,        *0002 2742       [Bits 13, 10, 8, 7, 6, 5, 1
        163,      109,      0,        0                [Shift down
        164,      109,      110,      *0001 4035       [Bits 12, 11, 4, 3, 2, 0
        165,      110,      109,      *0002 5002       [Bits 13, 11, 9, 1
        215,      126,      110,      (43)             [Exit if punching error in
                                                       [  either half, or if two
                                                       [  halves incompatable
        127,      109,      0,        31.4
        101,      109,      109,      (49)
        125,      109,      0,        0
        121,      126,      0,        (3/517)          [Return to (20) or (23)

43)     165,      109,      102,      2.0              [If fault is on first column
        215,      126,      109,      (47)             [   translate as binary
        101,      110,      100,      (53)*7
        124,      110,      0,        0.4              [Count errors
        113,      110,      100,      (53)*7           [If (53) even, continue
        121,      109,      0,        *4000 0077       [   with Fault character
        211,      126,      110,      (3/517)
        121,      104,      103,      0.4
        113,      104,      100,      (62)*7
        121,      104,      0,        4.0              [Otherwise indicate
        113,      104,      100,      (51)*7           [   faulty punching
        121,      104,      0,        0.3
        101,      110,      100,      (68)*7
        113,      104,      110,      *6               [Disengage reader
        121,      104,      0,        0
        121,      126,      0,        (3/517)

47)     121,      109,      0,        *3               [Change to binary
        121,      126,      0,        (3/517)          [   for this card only
48)     124,      104,      0,        0.4
        124,      105,      0,        0.4
        121,      126,      0,        (10)

                                                              2.9.63
```

```
                                                [R533.5

                                                [Harwell code conversion table

49)   *0140 0000  /  *2240 0077                 [Sp           2
      *2340 0331  /  *2440 4342                 [3            4
      *2140 0225  /  *2540 4242                 [1            5
      *2640 6000  /  *2740 6000                 [6            7
      *3540 0115  /  *4240 4142                 [+            B
      *4340 6000  /  *4440 6000                 [C            D
      *4140 6000  /  *4540 6000                 [A            E
      *4640 6000  /  *4740 6000                 [F            G

      *3640 0453  /  *5340 6000                 [-            K
      *5440 4742  /  *5540 6000                 [L            M
      *5240 4642  /  *5640 6000                 [J            N
      *5740 6000  /  *6040 6000                 [O            P
      *2040 4542  /  *6340 6000                 [0            S
      *6440 6000  /  *6540 6000                 [T            U
      *1740 6000  /  *6640 6000                 [/            V
      *6740 6000  /  *7040 6000                 [W            X

      *3043 0405  /  *0000 6000                 [8            (2,8)
      *3442 4700  /  *4040 6000                 [=            '
      *3142 4600  /  *1540 6000                 [9            &
      *1750 6000  /  *7740 6000                 [:            (7,8)
      *5042 4500  /  *0000 6000                 [H            (+,2,8)
      *3740 6000  /  *1140 6000                 [.            )
      *5140 6000  /  *3340 6000                 [I            >
      *2650 6000  /  *2750 6000                 [_            →

      *6140 6000  /  *0000 6000                 [Q            (-,2,8)
      *1340 6000  /  *1640 6000                 [             *
      *6240 6000  /  *1440 6000                 [R            ?
      *2250 6000  /  *2150 6000                 []            [
      *7140 6000  /  *0000 6000                 [Y            (0,2,8)
      *1240 6000  /  *1040 6000                 [,            (
      *7240 6000  /  *3240 6000                 [Z            ≥
      *7740 6000  /  *7740 6000                 [(0,6,8)      (0,7,8)
```

2.9.63

```
                                              [R540
      R540                                    [TR7 fault testing routine

      (10)     =      (5/724)                 [Entry to input master

1)    101,    109,   102,    *6
      165,    108,   109,    0.2
      215,    126,   108,    (6)              [Jump if stopped

      215,    126,   103,    (2)
      121,    108,   0,      -1.0             [If M = 0 set M = -1.0
      121,    126,   0,      (8/508)

2)    121,    108,   0,      1.4              [If M = -1.0, no LAM's
3)    121,    109,   0,      0.3
4)    113,    109,   102,    *6               [Stop and disengage
5)    121,    109,   0,      (1/568)          [Call tape reader P.E.R.
      121,    126,   0,      (7/508)

6)    214,    126,   104,    (9)              [Call input master if reader is free
      165,    108,   109,    *0001 0000
      214,    126,   108,    (7)
      121,    108,   0,      1.0              [If disabled, set M = 1.0
      121,    126,   0,      (3)              [Call P.E.R.

7)    165,    108,   109,    *0000 2000
      214,    126,   108,    (8)
      121,    109,   0,      2.3              [Reset overdue, stop, disengage
      121,    108,   0,      2.0              [M = 2.0
      121,    126,   0,      (4)              [Call P.E.R.

8)    165,    108,   109,    *0000 4000
      214,    126,   108,    (9)
      121,    108,   0,      6.0              [Tape low, set M = 6.0
      121,    126,   0,      (3)              [Call P.E.R.

9)    121,    108,   0,      -0.4             [If no fault, set M = -0.4
      121,    109,   0,      (10)             [Call input master
      121,    126,   0,      (7/508)
```

2.9.63

```
                                                      [R541

      R541                                            [TR7 interruption

      (10)     =       (5/599)9.0                     [Subsidiary store address

      (50)     =       (50/599)
      (51)     =       (51/599)
      (53)     =       (53/599)
      (61)     =       (61/599)
      (62)     =       (62/599)

 1)   101,     123,    0,       *6004 3640            [TR7 number
      101,     111,    123,     (10)                  [Address of private store
      101,     115,    123,     *6004 0400            [Read
      113,     0,      111,     (51)*7                [M = 0
      121,     112,    0,       0.4                   [Prepare POLAM
      101,     113,    111,     (62)*7
      121,     114,    113,     0.4                   [Next address
      113,     114,    111,     (62)*7
      113,     115,    113,     *7                    [Write to buffer
      102,     114,    111,     (61)*7
      217,     125,    114,     (2)                   [Test buffer full

      121,     112,    0,       0.6*4                 [POLAM and stop
      121,     117,    0,       -0.4                  [M = -0.4
 2)   101,     116,    115,     (11/568)              [LC/FS table
      101,     114,    11,      (50)*7
      127,     114,    0,       *0077 7777
      215,     125,    114,     (3)                   [Test last three characters

      165,     118,    116,     *6
      122,     118,    0,       *2                    [Ignore shift changes
      214,     125,    118,     (4)
      121,     112,    0,       0.6*4                 [POLAM and stop
      121,     117,    0,       0.4                   [M = 0.4
 3)   125,     114,    116,     *7777 7762            [Subtract * and add to last three
      113,     114,    111,     (50)*7                [Preserve for next time
 4)   215,     125,    116,     (5)                   [Test parity fault

      101,     114,    111,     (53)*7
      210,     112,    114,     0.6*4                 [POLAM and stop
      210,     117,    114,     4.0                   [M = 4.0
      124,     114,    0,       0.4
      113,     114,    111,     (53)*7                [Count of parity faults
 5)   113,     112,    123,     *6004 0400
      216,     125,    112,     (1/500)               [Exit to M/E unless stopping
      113,     117,    111,     (51)*7                [Set M
      121,     112,    0,       (1/568)               [Call R568 to S.E.R. queue
      121,     125,    0,       (4/201)
```

2.9.63

```
            R 550                                        [Anelex fault testing routine

         (10)=(3/514)

1)       101,     106,     102,     *6
         165,     107,     106,     0.2
         215,     126,     107,     (2)                  [Testing if stopped
         121,     126,     0,       (9/508)              [Exit if started

2)       127,     106,     0,       1.4                  [If stopped
         101,     108,     106,     (3)                  [Set m
         101,     126,     106,     (4)

3)       -0.4     /        3.0                           [No fault     :   Paper low
         1.0      /        1.0                           [Disabled     :   Disabled and paper low

4)       (5)      /        (6)
         (6)      /        (6)

5)       121,     109,     0,       (2/553)              [Stopped without fault
         121,     126,     0,       (7/508)

6)       121,     109,     0,       (10)                 [Stopped with fault
         121,     106,     0,       0.3
         113,     106,     102,     *6                   [Stop and disengage
         121,     126,     0,       (7/508)
```

2.9.63

```
          R551                                         [Anelex Interruption


          (74)      =       13.4(5/599)                [Subsidary address of printer
          (51)      =       (51/599)
          (99)      =       (99/599)


    1)    101,      123,    0,        *6004 3430
          101,      125,    123,      (2)

    3)    121,      113,    0,        0.7
          113,      113,    123,      *6004 1000       [Put out LAM, stop, and disengage
          101,      111,    123,      (74)             [Find private store address
          121,      114,    0,        5.0
          113,      114,    111,      (51)*7           [Note overflow
          121,      112,    0,        (3/514)          [Exit to output master routine
          121,      125,    0,        (4/201)

    4)    101,      111,    123,      -4.0(74)         [Private store address
          121,      113,    0,        0.6
          113,      113,    123,      -4.0*6004 1000   [Put out LAM and stop
          121,      114,    0,        -0.4
          113,      114,    111,      (51)*7           [Printing finished
          121,      112,    0,        (1/553)
          121,      125,    0,        (4/201)

    2)    (99)      /       (99)
          (3)       /       (99)                       [Overflow
          (99)      /       (99)
          (99)      /       (99)
          (99)      /       (99)
          (4)       /       (99)                       [Print
          (99)      /       (99)
          (99)      /       (99)
          (1/500)   /       (99)




                                                       2.9.63
          R553                                            [R553.1
                                                          [Anelex printer PER

          (56)      =         (56/599)
          (62)      =         (62/599)
          (51)      =         (51/599)
          (50)      =         (50/599)
          (61)      =         (61/599)
          (67)      =         (67/599)
          (99)      =         (99/599)
          (68)      =         (68/599)

1)        101,      102,      100,      (56)*7            [Characters left in present record
          104,      102,      100,      (50)*7            [Add in carriage control character
          214,      126,      102,      (1/514)           [Exit if printing finished

2)        121,      110,      0,        (3)
          121,      126,      0,        (1/520)           [Block to cores
3)        121,      110,      0,        (4)               [Sets 101 and 102
          121,      126,      0,        (2/520)

4)        101,      104,      100,      (61)*7            [Set 104 to available room in buffer
          102,      104,      100,      (67)*7
          101,      103,      100,      (68)*7            [V store address
5)        101,      105,      100,      (56)*7            [Character count
          124,      105,      105,      0
          124,      105,      105,      0
          121,      110,      0,        (7)
          121,      126,      0,        (1/516)           [Set code conversion parameters

7)        121,      110,      0,        (24)              [Internal code
          210,      110,      109,      (27)              [Binary
          164,      110,      101,      0.3
          101,      107,      101,      *7                [First half word
          211,      126,      109,      (2/512)           [Shift up half word (Int code)
          121,      126,      0,        (1/512)           [ditto              (binary)

8)        (9)       /         (10)                        [I.S Table/OS. Table
          (99)      /         (99)
          (21)      /         (99)
          (20)      /         (20)
          (20)      /         (99)
          (11)      /         (30)                        [Carriage control
          (99)      /         (32)
          (36)      /         (99)
```

2.9.63

```
21)    202,    126,    104,    (22)           [Test if 120 characters
       121,    126,    0,      (43)           [Go to cause line feed

22)    113,    109,    103,    *6             [Transfer character to buffer
20)    124,    101,    0,      0.1            [Increase store address

24)    202,    126,    106,    (25)           [Jump unless beginning half word

       121,    106,    0,      1.4
       101,    109,    101,    *7

       165,    107,    109,    *0303 0303
       125,    107,    0,      0
       164,    107,    109,    *7474 7474

25)    125,    107,    0,      0
       165,    109,    107,    *0000 0374     [Character in B 109
       202,    126,    105,    (2/517)        [Convert into Anelex code, Return to
                                              [  20, or 21
                                              [No characters remaining

19)    101,    109,    100,    (62)*7
       214,    126,    109,    2(0)           [Jump unless first time
       113,    109,    100,    (50)*7         [Marker to clear store and start printer
23)    113,    105,    100,    (56)*7         [Store number of characters remaining
                                              [  in record, or zero

       121,    110,    0,      2(0)
       121,    126,    0,      (1/518)        [Preserve current shift
       121,    107,    0,      0.1            [Inhibit interruptions
       113,    107,    0,      3*6
       101,    107,    103,    *6
       211,    126,    107,    (17)           [Jump if engaged
       113,    0,      0,      3*6            [Permit interruptions
       113,    0,      100,    (51)*7         [Exit to co ordinator until
       121,    110,    0,      (1/202)        [ printer engaged
       121,    126,    0,      (4/523)

17)    214,    126,    105,    (1/527)        [Carriage control, return to
                                              [   (30),(32),(36)
       121,    109,    0,      *0000 2020     [Set for one line feed
       121,    126,    0,      (34)
```

```
                                                   [R553.3
                                                   [Binary

29)     113,    109,    103,    *6                 [Print character
        124,    101,    0,      0.2
27)     203,    126,    106,    (26)
        121,    106,    0,      1.0
        101,    107,    101,    *7                 [Read next half word
26)     125,    107,    0,      0
        125,    107,    0,      0
        165,    109,    107,    *0000 0077         [And out character
        125,    109,    0,      0
        163,    109,    0,      0                  [Shift character to position for
        163,    109,    0,      -128               [ printing
        203,    126,    105,    (28)               [Jump if not end of record
        121,    126,    0,      (19)               [Return to print record

28)     202,    126,    104,    (29)               [Jump if line not full
        124,    105,    0,      0.4
        121,    126,    0,      (43)               [Go to cause new line
```

2.9.63

```
30)   124,   101,   0,     0.1              [Increase store address
34)   124,   109,   0,     *0000 4000       [End of line
31)   113,   109,   103,   *6               [Vertical format character
      113,   0,     0,     3*6              [Permit interruptions
      121,   110,   0,     2(0)
      214,   126,   105,   (1/521)          [Go to read next separator if at
                                            [  end of record.

41)   121,   110,   0,     (42)
      121,   126,   0,     (4/523)          [Remove reserved block label
42)   121,   109,   0,     1.5              [Disabled,disengaged,and paper out,bit
s
      121,   126,   0,     (1/515)          [Start printer

35)   124,   101,   0,     0.1
36)   113,   0,     0,     3*6              [Permit interruptions
      121,   110,   0,     (33)
      121,   126,   0,     (1/521)          [Read next separator
33)   215,   126,   108,   (5)              [Return unless final separator
      120,   104,   0,     60               [Store position along buffer
      113,   104,   100,   (67)*7
      121,   110,   0,     (1/514)
      121,   126,   0,     (4/523)          [Exit to output master routine

43)   101,   109,   100,   (61)*7
      113,   109,   100,   (67)*7           [Buffer full
      163,   105,   0,     0
      163,   105,   0,     -0.1             [Restore character count
      121,   126,   0,     (23)

32)   101,   108,   100,   (50)*7
      122,   108,   0,     *5
      215,   126,   108,   (35)             [Return unless printer requires starti
      121,   109,   0,     *0001 0002       [Clear core store and stop printer
      121,   126,   0,     (31)
```

```
                                              [553.5
                                              [Inner set table
9)     (98)            /     *3000 0000       [00    04
       *4000 2340      /     *4000 3640       [10    14
       *4000 2020      /     *4000 3220       [20    24
       *4000 3420      /     *4000 2560       [30    34
       *4000 2440      /     *4000 2640       [40    44
       *4000 3740      /     *4000 3040       [50    54
       *4000 2140      /     *4000 2240       [60    64
       *4000 3340      /     (98)             [70    74

       *4000 3000      /     *2000 0000       [01    05
       *4000 2400      /     *4000 3300       [11    15
       *4000 3060      /     *4000 2260       [21    25
       *4000 2460      /     *4000 3140       [31    35
       *4000 3560      /     *4000 3660       [41    45
       *4000 2760      /     *4000 2060       [51    55
       *4000 3160      /     *4000 3260       [61    65
       *4000 2360      /     (98)             [71    75

       *4000 3000      /     0                [02    06
       *4000 2660      /     *4000 3440       [12    16
       *4000 3120      /     *4000 2320       [22    26
       *4000 3520      /     *4000 2100       [32    36
       *4000 3600      /     *4000 2700       [42    46
       *4000 2000      /     *4000 3100       [52    56
       *4000 3200      /     *4000 2300       [62    66
       *4000 3400      /     (98)             [72    76

       (98)            /     0                [03    07
       *4000 2720      /     *4000 3500       [13    17
       *4000 2160      /     *4000 3360       [23    27
       *4000 3540      /     *4000 2200       [33    37
       *4000 2620      /     *4000 3720       [43    47
       *4000 3020      /     *4000 2120       [53    57
       *4000 2220      /     *4000 3320       [63    67
       (98)            /     (98)             [73    77

       (98)            =     *4000 2200
```

2.9.63

```
                                                           [R553.6
                                                           [Outer set table

10)     (98)                /         *3000 0000           [00    04
        (98)                /         (98)                 [10    14
        (98)                /         (98)                 [20    24
        *4000 3700          /         *4000 2040           [30    34
        (98)                /         *4000 2640           [40    44
        *4000 3740          /         *4000 3040           [50    54
        *4000 2140          /         *4000 2240           [60    64
        *4000 3340          /         (98)                 [70    74

        *4000 3000          /         *2000 0000           [01    05
        (98)                /         (98)                 [11    15
        *4000 3460          /         (98)                 [21    25
        (98)                /         *4000 2520           [31    35
        *4000 3560          /         *4000 3660           [41    45
        *4000 2760          /         *4000 2060           [51    55
        *4000 3160          /         *4000 3260           [61    65
        *4000 2360          /         (98)                 [71    75

        (98)                /         0                    [02    06
        (98)                /         (98)                 [12    16
        *4000 2500          /         *4000 2600           [22    26
        *4000 2740          /         *4000 2540           [32    36
        *4000 3600          /         *4000 2700           [42    46
        *4000 2000          /         *4000 3100           [52    56
        *4000 3200          /         *4000 2300           [62    66
        *4000 3400          /         (98)                 [72    76

        *4000 3240          /         0                    [03    07
        (98)                /         *4000 2420           [13    17
        (98)                /         *4000 3620           [23    27
        *4000 3760          /         (98)                 [33    37
        *4000 2620          /         *4000 3720           [43    47
        *4000 3020          /         *4000 2120           [53    57
        *4000 2220          /         *4000 3320           [63    67
        (98)                /         (98)                 [73    77
```

2.9.63

```
                                                    [R553.7
                                                    [Carriage control table
11)     *0000 2000      /       *6000 2020          [0-17 N line feeds 0-15
        *6000 2040      /       *6000 2060
        *6000 2100      /       *6000 2120
        *6000 2140      /       *6000 2160
        *6000 2200      /       *6000 2220
        *6000 2240      /       *6000 2260
        *6000 2300      /       *6000 2320
        *6000 2340      /       *6000 2360

        *6000 2000      /       *6000 2020          [20-37, Carriage return and
        *6000 2040      /       *6000 2060          [ line feed 0-15 New Lines.
        *6000 2100      /       *6000 2120
        *6000 2140      /       *6000 2160
        *6000 2200      /       *6000 2220
        *6000 2240      /       *6000 2260
        *6000 2300      /       *6000 2320
        *6000 2340      /       *6000 2360

        *6000 3620      /       *6000 3620          [Home on channels 1-7
        *6000 3640      /       *6000 3660
        *6000 3700      /       *6000 3720
        *6000 3740      /       *6000 3760

        *6000 3620      /       *6000 3620          [Home on channels 1-7
        *6000 3640      /       *6000 3660
        *6000 3700      /       *6000 3720
        *6000 3740      /       *6000 3760

        *0000 2000      /       *0000 2000
        *0000 2000      /       *0000 2000
        *0000 2000      /       *0000 2000
        *0000 2000      /       *0000 2000
        *0000 2000      /       *0000 2000
        *0000 2000      /       *0000 2000
        *0000 2000      /       *0000 2000
        *0000 2000      /       *0000 2000
```

2.9.63

```
                                                  [R 560

        R 560                                     [Creed 3000 fault testing routine

        (10)    =       (3/514)                   [Return to master routine


 1)     101,    106,    102,    *6
        165,    107,    106,    0.2               [Test stopped
        214,    126,    107,    (9/508)           [Exit if started

        165,    107,    106,    1.0
        214,    126,    107,    (3)
        121,    108,    0,      1.0               [Disabled, set M = 1.0
        121,    126,    0,      (6)

 3)     165,    107,    106,    2.0
        214,    126,    107,    (4)
        121,    108,    0,      6.4               [Check failed, set M = 6.4
        121,    109,    0,      2.3               [Reset check failed
        121,    126,    0,      (7)
 4)     165,    107,    106,    0.4
        214,    126,    107,    (9)
        121,    108,    0,      3.0               [Tape out, set M = 3.0
 6)     121,    109,    0,      0.3
 7)     113,    109,    102,    *6                [Stop and disengage
 8)     121,    109,    0,      (10)
        121,    126,    0,      (7/508)           [Return to master routine

 9)     121,    108,    0,      -0.4
        121,    109,    0,      (2/573)
        121,    126,    0,      (7/508)           [Return to P.E.R.
```

2.9.63

```
                                                  [R561

       R 561                                      [Creed 3000 interruption

       (10)    =      (5/599)9.4                  [Subsidiary store address

       (51)    =      (51/599)
       (62)    =      (62/599)


1)     101,    123,   0,      *6004 3560          [Which Creed 3000
       101,    111,   123,    (10)                [Find private store
       101,    113,   111,    (62)*7              [Present address
       121,    114,   113,    0.4
       113,    114,   111,    (62)*7              [Next address
       101,    112,   113,    *7                  [Pick up character and POLAM
       113,    112,   123,    *6004 1400          [Send to punch
       127,    112,   0,      0.2
       214,    125,   112,    (1/500)             [Exit to M/E unless stopping
       121,    112,   0,      -0.4
       113,    112,   11,     (51)*7
       121,    112,   0,      (1/573)
       121,    125,   0,      (4/201)             [Call P.E.R. to queue
```

2.9.63

```
                                                        [R565

        R565                                            [TR5 Fault testing routine

        (10)     =        (5/724)                       [Entry address of Input Master Routine

 1)     101,     109,     102,     *6                   [V store line
        165,     108,     109,     0.2
        215,     126,     108,     (5)                  [Jump if stopped

        215,     126,     103,     (3)
        121,     108,     0,       -1.0                 [If M = 0 set M = -1.0
        121,     126,     0,       (8/508)

 3)     121,     109,     0,       0.3                  [If M = -1.0
        121,     108,     0,       1.4                  [        set M = 1.4
 4)     113,     109,     102,     *6                   [        disengage
        121,     109,     0,       (1/568)              [        and call P.E.R.
        121,     126,     0,       (7/508)

 5)     214,     126,     104,     (6)                  [Call input master if reader is free
        127,     109,     0,       *0000 2000
        215,     126,     109,     (7)                  [Jump if overdue

 6)     121,     108,     0,       -0.4                 [Set M = -0.4
        121,     109,     0,       (10)                 [Call input master
        121,     126,     0,       (7/508)

 7)     121,     108,     0,       2.0                  [If overdue set M = 2.0
        121,     109,     0,       2.3                  [Reset overdue and disengage
        121,     126,     0,       (4)                  [Call P.E.R.
```

2.9.63

```
                                                         [R566

          R566                                           [T.R.5 interruption

          (70)     =      *6004 3530                     [T.R.5 Look at mes.
          (71)     =      (5/599)                        [Subsidiary store address.
          (72)     =      *6004 1600                     [V store address.
          (50)     =      (50/599)
          (51)     =      (51/599)
          (53)     =      (53/599)
          (61)     =      (61/599)
          (62)     =      (62/599)


  1)      101,     123,   0,       (70)                  [T.R.5 number.
          101,     111,   123,     (71)                  ddress of private store.
          101,     115,   123,     (72)                  [V store bits.
          113,     0,     111,     (51)*7                [M=0
          121,     112,   0,       0.4                   [Prepare P.O.L.A.M.
          101,     113,   111,     (62)*7
          121,     114,   113,     0.4                   [Next address.
          113,     114,   111,     (62)*7
          113,     115,   113,     *7                    [Write to buffer.
          102,     114,   111,     (61)*7
  10)     217,     125,   114,     (2)
                                                         [If buffer full:
          121,     112,   0,       0.6*4                 [P.O.L.A.M. and stop.
          121,     117,   0,       -0.4                  [M=-0.4
  2)      101,     116,   115,     (11/568)              [LC/FS table.
          101,     114,   111,     (50)*7                [Last three characters.
          127,     114,   0,       *0077 7777
          215,     125,   114,     (3)
          165,     118,   116,     *6                    [If ending sequence:
          12,      118,   0,       *2                    [Ignore shift changes
          214,     125,   118,     (5)
          121,     112,   0,       0.6*4                 [P.O.L.A.M. and stop.
          121,     117,   0,       0.4                   [M=0.4
  3)      125,     114,   116,     *7777 7762            [Subtract * and add to last three.
          113,     114,   111,     (50)*7                [Preserve for next time.
  5)      215,     125,   116,     (4)
                                                         [If parity fault:
          101,     114,   111,     (53)*7                [If (53) odd:
          210,     112,   114,     0.7*4                 [P.O.L.A.M. stop and disengage
          210,     117,   114,     4.0                   [M=4.0
          124,     114,   0,       0.4
          113,     114,   111,     (53)*7                [Preserve count of faults.

  4)      113,     112,   123,     (72)
          216,     125,   112,     (1/500)               [Exit to M/E unless stopping.
          113,     117,   111,     (51)*7                [Set M.
          121,     112,   0,       (1/568)               [Call R568 to S.E.R. queue.
          121,     125,   0,       (4/201)
```

2.9.63

```
                                                  [R568.1

            R568                                  [TR5 P.E.R.


                  (51)=(51/599)

 1)         121,      110,      0,      (2)
            121,      126,      0,      (1/520)   [Block to cores
 2)         121,      110,      0,      (3)
            121,      126,      0,      (2/520)   [Set reserved block label
                                                  [         Sets B101, B102
 3)         121,      110,      0,      (4)
            121,      126,      0,      (2/522)   [Find part page
                                                  [         Sets B103, B104
 4)         121,      110,      0,      (6)
            121,      126,      0,      (1/516)   [Set code conversion parameters
                                                  [         Sets B108,B109
 6)         121,      110,      0,      (7)

            121,      126,      0,      (1/511)   [Find store length available.
                                                  [         Sets B105
 7)         217,      126,      105,    (42)      [Go to end if exceeded
            124,      105,      105,    0
            124,      105,      105,    0
            101,      107,      101,    *7        [ Current half word
            121,      110,      0,      (8)
            164,      110,      101,    0.3
            121,      126,      0,      (1/512)   [Shift next available space to top
                                                  [         end.   Sets B106


 8)         210,      126,      109,    (14)      [Jump if binary
            202,      126,      106,    (26)      [Jump and set count if in middle of
                                                  [                     half word
            121,      106,      0,      1.4       [Set count and half word
            121,      107,      0,      0         [         if at beginning
            121,      126,      0,      (26)
```

2.9.63

```
                                                        [R568.2


                                                        [BINARY.

14)     127,    105,    0,      *77777770
        127,    106,    0,      1.0                     [Round down counters
        203,    126,    106,    (15)                    [Jump if in middle of half word.
        121,    106,    0,      1.0                     [Set count and half word if
        121,    107,    0,      0                       [       at beginning.
        121,    126,    0,      (15)

17)     101,    109,    103,    *7
        163,    109,    0,      0                       [Single out character
        163,    109,    0,      0                       [ and pack into
        127,    109,    0,      15.7                    [  half word
        125,    107,    0,      0
        125,    107,    109,    0
        203,    126,    106,    (18)
        113,    107,    101,    *7                      [If half word full, send to
        121,    106,    0,      1.0                     [  store and reset
        121,    107,    0,      0                       [  count
18)     124,    103,    0,      0.4                     [Advance buffer address
        124,    101,    0,      0.2                     [Advance store address
15)     202,    126,    104,    (16)                    [Jump unless buffer empty
        121,    126,    0,      (30)                    [Exit if empty

16)     203,    126,    105,    (17)                    [Jump unless store full
        121,    126,    0,      (33)
```

```
5)     (10)     /      (11)                     [UC and LS table / LC and FS table
       0.5      /      0.4                      [LS              /  CS
       (21)     /      (22)                     [Ordinary char.  /  The other set
       (20)     /      (21)                     [Significant shift / Redundent shift
       (25)     /      (99/599)                 [NL or wrong parity  /  spare


                                                [INTERNAL CODE

22)    122,     103,     0,      0.4            [Step back buffer address.
       124,     104,     0,      0.4
21)    127,     109,     0,      7.7            [Pick out character.
       125,     107,     109,    0              [Put into half word.
       202,     126,     106,    (24)           [Jump unless half word is full.
       113,     107,     101,    *7             [Send to store.
       121,     106,     0,      1.4
       121,     107,     0,      0
24)    124,     101,     0,      0.1            [Advance main store address.
23)    124,     103,     0,      0.4            [Advance buffer address.

26)    202,     126,     104,    (28)           [Jump if buffer space
30)    101,     105,     100,    (51)*7         [If buffer finished, find reason for
       121,     126,     0,      (33)           [       stopping.

28)    202,     126,     105,    (1/517)        [ Code conversion if store space,
                                                [ return to (20) (21) (22) or (25)


33)    165,     108,     101,    0.3
       121,     110,     0,      (34)
       121,     126,     0,      (1/513)
34)    113,     107,     101,    *7             [Write away last half word.
       121,     110,     0,      (47)
       216,     126,     105,    (2/519)        [Insert separator
       121,     126,     0,      (1/518)        [If buffer full, preserve parameters

25)    215,     126,     109,    (36)           [Jump if end of record.
29)    121,     109,     0,      *4000 0077     [If parity fault, insert Fault.
       121,     126,     0,      (3/517)


36)    127,     109,     0,      7.7
37)    125,     107,     109,    0              [Insert carriage control character.
       113,     107,     101,    *7             [Write to store.
       124,     101,     0,      0.1
       124,     103,     0,      0.4            [Advance addresses
       165,     108,     101,    0.3
       121,     110,     0,      (38)
       121,     126,     0,      (1/513)
38)    113,     107,     101,    *7             [Write away last half word
       101,     105,     100,    (51)*7
       121,     110,     0,      (6)            [Set return to next character,
       214,     110,     104,    (47)           [    unless buffer is empty
       121,     126,     0,      (1/519)        [Insert separator.

20)    124,     105,     0,      0.4
       121,     126,     0,      (23)
```

```
42)     121,    105,    0,      0               [Set M=0

47)     121,    110,    0,      (48)
        121,    126,    0,      (1/523)         [Remove reserved block label

48)     121,    101,    105,    0               [Copy M.
        124,    105,    0,      0.4             [Test for -0.4
        215,    126,    105,    (2/514)         [If  M ≠ -0.4, exit to main program
49)     121,    109,    0,      *0001 6001      [Fault bits  TR5, TR7
        121,    126,    0,      (1/515)         [If M=-0.4, start reader.
```

2.9.63

```
                               [FLEXOWRITER UPPER CASE
10)   0        /      0.0*1   [0000.000 Fault              0000.001 (Spare)
      2.1      /      0       [     010 NL                      011 Fault
      0.2*4    /      0       [     100 Tab                     101 Fault
      0        /      0.7*2   [     110 Fault                   111 UC

      1.0*5    /      0       [0001.000 (Spare)            0001.001 Fault
      0        /      1.3*5   [     010 Fault                   011 (Spare)
      0        /      1.5*5   [     100 Fault                   101 Punch on
      1.6*5    /      0       [     110 Punch off               111 Fault

      0.1*1    /      0       [0010.000 Sp                 0010.001 Fault
      0        /      4.0     [     010 Fault                   011 P Throw
      0        /      0.3*4   [     100 Fault                   101 B SP.
      0.6*3    /      0       [     110 LC                      111 Fault

      0        /      1.1*5   [0011.000 Fault              0011.001 (Spare)
      1.2*5    /      0       [     010 (Spare)                 011 Fault
      1.4*5    /      0       [     100 Stop                    101 Fault
      0        /      1.7*4   [     110 Fault                   111  /

      2.0*4    /      0       [0100.000 Zero               0100.001 Fault
      0        /      2.3*4   [     010 Fault                   011 3
      0        /      2.5*4   [     100 Fault                   101 5
      2.6*4    /      0       [     110 6                       111 Fault

      0        /      3.1*4   [0101.000 Fault              0101.001 9
      3.2*5    /      0       [     010 &                       011 Fault
      3.4*5    /      0       [     100 ½                       101 Fault
      0        /      3.7*4   [     110 Fault                   111 .

      0        /      2.1*4   [0110.000 Fault              0110.001 1
      2.2*4    /      0       [     010 2                       011 Fault
      2.4*4    /      0       [     100 4                       101 Fault
      0        /      2.7*4   [     110 Fault                   111 7

      3.0*4    /      0       [0111.000 8                  0111.001 Fault
      0        /      3.3*5   [     010 Fault                   011 II
      0        /      3.5*4   [     100 Fault                   101 +
      3.6*4    /      0       [     110 -                       111 Fault
```

```
                              [FLEXOWRITER UPPER CASE
3.7*5     /     0             [1000.000 (Spare)             1000.001 Fault
0         /     4.3*4         [     010 Fault                    011 C
0         /     4.5*4         [     100 Fault                    101 E
4.6*4     /     0             [     110 F                        111 Fault

0         /     5.1*4         [1001.000 Fault               1001.001 I
5.2*4     /     0             [     010 J                        011 Fault
5.4*4     /     0             [     100 L                        101 Fault
0         /     5.7*4         [     110 Fault                    111 O

0         /     4.1*4         [1010.000 Fault               1010.001 A
4.2*4     /     0             [     010 B                        011 Fault
4.4*4     /     0             [     100 D                        101 Fault
0         /     4.7*4         [     110 Fault                    111 G

5.0*4     /     0             [1011.000 H                   1011.001 Fault
0         /     5.3*4         [     010 Fault                    011 K
0         /     5.5*4         [     100 Fault                    101 M
5.6*4     /     0             [     110 N                        111 Fault

0         /     6.1*4         [1100.000 Fault               1100.001 Q
6.2*4     /     0             [     010 R                        011 Fault
6.4*4     /     0             [     100 T                        101 Fault
0         /     6.7*4         [     110 Fault                    111 W

7.0*4     /     0             [1101.000 X                   1101.001 Fault
0         /     7.3*4         [     010 Fault                    011 (Spare)
0         /     7.5*4         [     100 Fault                    101 (Spare)
7.6*4     /     0             [     110 (Spare)                  111 Fault


6.0*4     /     0             [1110.000 P                   1110.001 Fault
0         /     6.3*4         [     010 Fault                    011 S
0         /     6.5*4         [     100 Fault                    101 U
6.6*4     /     0             [     110 V                        111 Fault

0         /     7.1*4         [1111.000 Fault               1111.001 Y
7.2*4     /     0             [     010 Z                        011 Fault
7.4*4     /     0             [     100 (Spare)                  101 Fault
0         /     7.7*5         [     110 Fault                    111 Erase
```

[R568.7

```
                                        [5 CHANNEL LETTERS
0.7*3     /            6.0*4   [00.000 FS         00.001 P
5.0*4     /            7.0*4   [    010 H            011 X
4.4*4     /            6.4*4   [    100 D            101 T
5.4*4     /            3.7*4   [    110 L            111 .
4.2*4     /            6.2*4   [01.000 B          01.001 R
5.2*4     /            7.2*4   [    010 J            011 Z
4.6*4     /            6.6*4   [    100 F            101 V
5.6*4     /            1.3*4   [    110 N            111

4.1*4     /            6.1*4   [10.000 A          10.001 Q
5.1*4     /            7.1*4   [    010 I            011 Y
4.5*4     /            6.5*4   [    100 E            101 U
5.5*4     /            1.4*4   [    110 M            111 ?
4.3*4     /            6.3*4   [11.000 C          11.001 S
5.3*4     /            0.6*2   [    010 K            011 LS
4.7*4     /            6.7*4   [    100 G            101 W
5.7*4     /            7.7*5   [    110 O            111 Erase
```

2.9.63

```
                                    [FLEXOWRITER LOWER CASE
11)  0         /     0.0*1     [0000.000 Fault          0000.001 (Spare)
     2.1       /     0         [     010 NL                  011 Fault
     0.2*4     /     0         [     100 Tab                 101 Fault
     0         /     0.7*2     [     110 Fault               111 UC

     1.0*5     /     0         [0001.000 (Spare)        0001.001 Fault
     0         /     1.3*5     [     010 Fault               011 (Spare)
     0         /     1.5*5     [     100 Fault               101 Punch on
     1.6*5     /     0         [     110 Punch off           111 Fault

     0.1*1     /     0         [0010.000 Sp             0010.001 Fault
     0         /     4.0       [     010 Fault               011 P.Th.
     0         /     0.3*4     [     100 Fault               101 BSp
     0.6*3     /     0         [     110 LC                  111 Fault

     0         /     1.1*5     [0011.000 Fault          0011.001 (Spare)
     1.2*5     /     0         [     010 (Spare)             011 Fault
     1.4*5     /     0         [     100 Stop                101 Fault
     0         /     1.7*5     [     110 Fault               111 :

     4.0*4     /     0         [0100.000 '              0100.001 Fault
     0         /     3.2*4     [     010 Fault               011 ≥
     0         /     3.4*4     [     100 Fault               101 =
     2.6*5     /     0         [     110 __                  111 Fault

     0         /     1.1*4     [0101.000 Fault          0101.001  )
     3.0*5     /     0         [     010                     011 Fault
     1.4*4     /     0         [     100 ?                   101 Fault
     0         /     1.2*4     [     110 Fault               111 ,

     0         /     2.1*5     [0110.000 Fault          0100.001 [
     2.2*5     /     0         [     010 ]                   011 Fault
     3.3*4     /     0         [     100 >                   101 Fault
     0         /     2.7*5     [     110 Fault               111 →

     1.0*4     /     0         [0111.000 (              0101.001 Fault
     0         /     1.3*4     [     010 Fault               011
     0         /     1.5*4     [     100 Fault               101 &
     1.6*4     /     0         [     110 *                   111 Fault
```

2.9.63

```
                              [FLEXOWRITER LOWER CASE
4.0*5    /       o            [1000.000 (Spare)             1000.001 Fault
o        /       4.3*5        [      010 Fault                   011 C
o        /       4.5*5        [      100 Fault                   101 e
4.6*5    /       o            [      110 f                       111 Fault

o        /       5.1*5        [1001.000 Fault               1001.001 i
5.2*5    /       o            [      010 j                       011 Fault
5.4*5    /       o            [      100 l                       101 Fault
o        /       5.7*5        [      110 Fault                   111 o

o        /       4.1*5        [1010.000 Fault               1010.001 a
4.2*5    /       o            [      010 b                       011 Fault
4.4*5    /       o            [      100 d                       101 Fault
o        /       4.7*5        [      110 Fault                   111 g

5.0*5    /       o            [1011.000 h                   1011.001 Fault
o        /       5.3*5        [      010 Fault                   011 k
o        /       5.5*5        [      100 Fault                   101 m
5.6*5    /       o            [      110 n                       111 Fault

o        /       6.1*5        [1100.000 Fault               1100.001 q
6.2*5    /       o            [      010 v                       011 Fault
6.4*5    /       o            [      100 t                       101 Fault
o        /       6.7*5        [      110 Fault                   111 w

7.0*5    /       o            [1101.000 x                   1101.001 Fault
o        /       7.3*5        [      010 Fault                   011 (Spare)
o        /       7.5*5        [      100 Fault                   101 (Spare)
7.6*5    /       o            [      110 (Spare)                 111 Fault

6.0*5    /       o            [1110.000 p                   1110.001 Fault
o        /       6.3*5        [      010 Fault                   011 s
o        /       6.5*5        [      100 Fault                   101 u
6.6*5    /       o            [      110 u                       111 Fault

o        /       7.1*5        [1111.000 Fault               1111.001 y
7.2*5    /       o            [      010 z                       011 Fault
7.4*5    /       o            [      100 (Spare)                 101 Fault
o        /       7.7*5        [      110 Fault                   111 Erase
```

[R568.10

```
                              [5 CHANNEL FIGURES
0.7*3    /      2.0*4   [00.000 FS       00.001 Zero
3.0*4    /      2.0*5   [   010 8           011 Ø
2.4*4    /      2.3*5   [   100 4           101 ->
3.1*5    /      3.7*4   [   110 ÷           111 .

2.2*4    /      2.4*5   [01.000 2        01.001 ≥
3.4*4    /      3.5*4   [   010 =           011 +
1.1*4    /      2.6*4   [   100 )           101 6
0.1*1    /      2.0     [   110 Sp          111 CR

2.1*4    /      3.3*4   [10.000 1        10.001 >
2.5*5    /      3.1*4   [   010 ≠           011 9
1.0*4    /      2.5*4   [   100 (           101 5
0.1      /      4.0*4   [   110 LF          111 '

1.6*4    /      2.3*4   [11.000 *        11.001 3
3.6*4    /      0.6*2   [   010 -           011 LS
2.7*4    /      1.7*4   [   100 7           101 /
1.2*4    /      7.7*5   [   110 ,           111 Erase
```

2.9.63

```
                                                          [R570



                R570                                      [Teletype fault testing routine

                (10)=(3/514)                              [Return to Master Routine

        1)      101,     106,     102,     *6
                165,     107,     106,     0.2            [Test stopped
                215,     126,     107,     (2)
                121,     126,     0,       (9/508)        [Exit if started

        2)      127,     106,     0,       1.4            [If stopped:
                101,     108,     106,     (3)            [Set M
                101,     126,     106,     (4)            [Set address


        3)      -0.4     /        3.0                     [No fault     /  Paper out
                1.0      /        1.0                     [Disabled     /  Disabled and paper out

        4)      (5)      /        (6)
                (6)      /        (6)

        5)      121,     109,     0,       (43/573)       [Stopped without fault
                121,     126,     0,       (7/508)

        6)      121,     109,     0,       (10)           [Stopped with fault
                121,     106,     0,       0.3
                113,     106,     102,     *6             [Stop and disengage
                121,     126,     0,       (7/508)
```

2.9.63

```
                                                      [R571

       R571                                           [Teletype Interruption
                                                      [Teletypes 0 - 7

       (74)      =        (5/599)0.4                  [Subsidiary store address of
       (62)      =        (62/599)                    [     teletype 0
       (51)      =        (51/599)

1)     101,      123,     0,        *6004 3500        [Which teletype
       101,      111,     123,      (74)              [Find private store
       101,      113,     111,      (62)*7            [Address of this character
       121,      114,     113,      0.4               [Address of next character
       113,      114,     111,      (62)*7
3)     101,      112,     113,      *7                [Pick up character and POLAM
       113,      112,     123,      *6004 200         [Send to punch
       127,      112,     0,        0.2               [Exit to M/E unless stopping
       214,      125,     112,      (1/500)
       121,      112,     0,        -0.4
       113,      112,     111,      (51)*7
       121,      112,     0,        (1/573)
       121,      125,     0,        (4/201)           [Call P.E.R. to queue
```

2.9.63

```
                                                [R573.1

        R573                                    [Teletype punch P.E.R.

        (50)=(50/599)
        (56)=(56/599)
        (60)=(60/599)
        (62)=(62/599)
        (99)=(99/599)
        (17)=(9)
        (18)=(10)
        (19)=(11)


1)      101,      102,      100,      (56)*7        [Characters left in present record
        104,      102,      100,      (50)*7        [Add in carriage control character
        214,      126,      102,      (1/514)       [Exit if printing finished.

2)      121,      110,      0,        (3)
        121,      126,      0,        (1/520)       [Block to cores.
3)      121,      110,      0,        (4)
        121,      126,      0,        (2/520)       [Set reserved block label
                                                    [      Sets B101, B102
4)      121,      110,      0,        (5)
        121,      126,      0,        (1/522)       [Find output buffer.
                                                    [      Sets B103, B104.
5)      101,      105,      100,      (56)*7        [Character count
        124,      105,      105,      0
        124,      105,      105,      0
        121,      110,      0,        (7)
        121,      126,      0,        (1/516)       [Set code conversion parameters
                                                    [      Sets B108,B109
7)      121,      110,      0,        (24)          [Internal code
        210,      110,      109,      (27)          [Binary
        164,      110,      101,      0.3
        101,      107,      101,      *7            [First half word
        211,      126,      109,      (2/512)       [Shift up half word (Int. code)
        121,      126,      0,        (1/512)       [Ditto          (Binary)
```

2.9.63

```
                                                     [R573.2
8)        (9)                /      (10)             [7 hole tables.
          *0000 0344         /      *0000 1304       [UC + POLAM / LC + POLAM
          (21)               /      (22)
          (20)               /      (20)
          (99)               /      (99)
          (11)               /      (31)             [7 hole carriage control
          (31)               /      (32)
          (36)               /      (99)

12)       (13)               /      (14)             [5 hole tables
          *0000 0004         /      *0000 1544       [FS + POLAM / LS + POLAM
          (21)               /      (22)
          (20)               /      (20)
          (99)               /      (99)
          (15)               /      (31)             [5 hole carriage control
          (31)               /      (32)
          (36)               /      (99)

16)       (17)               /      (18)             [Teletype tables
          0                  /      0
          (21)               /      (22)
          (20)               /      (20)
          (99)               /      (99)
          (19)               /      (31)
          (31)               /      (32)
          (36)               /      (99)
```

2.9.63

```
                                                  [R573.3

20)    124,    104,    0,      0.4                [Restore buffer address
       121,    126,    0,      (23)

21)    113,    109,    103,    *7
       124,    103,    0,      0.4                [Put in buffer
23)    124,    101,    0,      0.1                [Advance main store address
24)    202,    126,    106,    (25)               [Jump unless beginning half word

       121,    106,    0,      1.4
       101,    109,    101,    *7
       165,    107,    109,    *0303 0303
       125,    107,    0,      0
       164,    107,    109,    *7474 7474

25)    125,    107,    0,      0
       165,    109,    107,    *0000 0374         [Character in B109
       202,    126,    105,    (28)               [Go to see if room in buffer
                                                  [If no characters remaining:
       113,    0,      100,    (56)*7             [Clear character count
       121,    110,    0,      (34)               [Preserve current shift
       121,    126,    0,      (1/518)

22)    113,    109,    103,    *7
       124,    103,    0,      0.4                [Put in buffer
       165,    109,    107,    *0000 0374         [Pick out same character again
28)    202,    126,    104,    (2/517)            [Jump provided there is buffer space
29)    163,    105,    0,      0                  [If not, restore B105, adding
       163,    105,    0,      -0.1               [    0.1 back to count
       113,    105,    100,    (56)*7             [Preserve remaining character count
       121,    110,    0,      (40)               [Preserve current shift and
       121,    126,    0,      (1/518)            [    start punch


                                                  [BINARY

51)    125,    109,    0,      1.0                [Shift up 5 places and add in
       163,    109,    0,      0                  [    POLAM
       113,    109,    103,    *7                 [Put in buffer
       124,    103,    0,      0.4                [Advance buffer address
       124,    101,    0,      0.2                [Advance main store address

27)    203,    126,    106,    (53)               [Jump if 2nd. character
       101,    107,    101,    *7                 [Read next half word
       121,    106,    0,      1.0
53)    125,    107,    0,      0
       125,    107,    0,      0
       165,    109,    107,    *0000 0177         [And out character
       203,    126,    105,    (52)
       113,    0,      100,    (56)*7             [No characters remaining
       101,    109,    100,    (50)*7
       214,    126,    109,    (36)               [If no carriage control character
       121,    126,    0,      (32)

52)    202,    126,    104,    (51)               [Jump if buffer space
       124,    105,    0,      0.4
       121,    126,    0,      (29)
                                                            2.9.63
```

```
31)     113,    109,    103,    *7              [Present character to buffer
        124,    103,    0,      0.4             [Advance buffer address

34)     202,    126,    104,    (1/527)         [Jump provided there is buffer
                                                [    Return to (31), (32) or(36)
        121,    126,    0,      (40)            [If none, go to print

32)     124,    101,    0,      0.1
36)     124,    104,    0,      0.4
        121,    110,    0,      (33)
        121,    126,    0,      (1/521)         [Read next separator
33)     215,    126,    108,    (5)             [Repeat, unless final separator


40)     121,    104,    103,    -0.4            [Address of previous character
        102,    104,    100,    (60)*7
        121,    110,    0,      (1/514)         [If no character to print,
        217,    126,    104,    (1/523)         [         remove reserved block
                                                [         number and return to
        121,    104,    103,    -0.4            [         master routine.
        101,    109,    104,    *7              [
        167,    109,    0,      0.2             [Put stop bit on last character
        113,    109,    104,    *7

        121,    110,    0,      (42)
        121,    126,    0,      (1/523)         [Remove reserved block number.
42)     121,    109,    0,      1.5             [Disabled and disengaged bits
        121,    126,    0,      (1/515)         [Start punch

43)     121,    109,    0,      1.5             [Return from fault testing
        121,    126,    0,      (2/515)
```

```
                                          [R573.5

                                          [INNER SET 7 HOLE TABLE

9)     *1000 0044    /      *3000 0000    [    00    /    04
       *5000 3404    /      *5000 2604    [    10    /    14
       *4000 2004    /      *4000 3204    [    20    /    24
       *4000 3404    /      *5000 2244    [    30    /    34
       *5000 2004    /      *4000 5204    [    40    /    44
       *4000 5404    /      *4000 4604    [    50    /    54
       *4000 7004    /      *4000 6204    [    60    /    64
       *4000 6404    /      *4000 7604    [    70    /    74

       *1000 1004    /      *2000 0000    [    01    /    05
       *5000 2444    /      *5000 3644    [    11    /    15
       *4000 3044    /      *4000 2244    [    21    /    25
       *4000 2444    /      *4000 3644    [    31    /    35
       *4000 5044    /      *4000 4244    [    41    /    45
       *4000 4444    /      *4000 5644    [    51    /    55
       *4000 6044    /      *4000 7244    [    61    /    65
       *4000 7444    /      *4000 6644    [    71    /    75

       *1000 0204    /      *5000 1304    [    02    /    06
       *5000 2744    /      *5000 3704    [    12    /    16
       *4000 3104    /      *4000 2304    [    22    /    26
       *5000 2144    /      *4000 3704    [    32    /    36
       *4000 5104    /      *4000 4304    [    42    /    46
       *4000 4504    /      *4000 5704    [    52    /    56
       *4000 6104    /      *4000 7304    [    62    /    66
       *4000 7504    /      *4000 6704    [    72    /    76

       *1000 1244    /      *4000 0344    [    03    /    07
       *5000 3544    /      *4000 1744    [    13    /    17
       *4000 2144    /      *4000 3344    [    23    /    27
       *5000 3204    /      *4000 2744    [    33    /    37
       *4000 4144    /      *4000 5344    [    43    /    47
       *4000 5544    /      *4000 4744    [    53    /    57
       *4000 7144    /      *4000 634     [    63    /    67
       *4000 6544    /      (98)          [T   73    /    77

       (98)     =     *4000 2744          [Not printable
```

2.9.63

```
                                                  [R573.6

                                                  [OUTER SET 7 HOLE TABLE.

10)     *1000 0044      /       *3000 0000       [     00   /   04
        *1000 1444      /       *1000 1604       [     10       14
        (98)            /       (98)             [     20       24
        *5000 2504      /       *4000 2604       [     30       34
        *5000 4004      /       *5000 5204       [     40       44
        *5000 5404      /       *5000 4604       [     50       54
        *5000 7004      /       *5000 6204       [     60       64
        *5000 6404      /       *5000 7604       [     70       74

        *1000 1004      /       *2000 0000       [     01       05
        *1000 1444      /       *1000 0644       [     11       15
        *5000 3044      /       (98)             [     21       25
        (98)            /       (98)             [     31       35
        *5000 5044      /       *5000 4244       [     41       45
        *5000 4444      /       *5000 5644       [     51       55
        *5000 6044      /       *5000 7244       [     61       65
        *5000 7444      /       *5000 6644       [     71       75

        (98)            /       *5000 1304       [     02       06
        *1000 1504      /       *1000 0704       [     12       16
        *5000 3104      /       *5000 2304       [     2       26
        *4000 2504      /       (98)             [     32       36
        *5000 5104      /       *5000 4304       [     42       46
        *5000 4504      /       *5000 5704       [     52       56
        *5000 6104      /       *5000 7304       [     62       66
        *5000 7504      /       *5000 6704       [     72       76

        (98)            /       *4000 0344       [     03       07
        *1000 0544      /       *5000 1744       [     13       17
        (98)            /       *5000 3344       [     23       27
        *4000 3544      /       *4000 4004       [     33       37
        *5000 4144      /       *5000 5344       [     43       47
        *5000 5544      /       *5000 4744       [     53       57
        *5000 7144      /       *5000 6344       [     63       67
        *5000 6544      /       *1000 7744       [     73       77
```

2.9.63

```
                                                          [R573.7

                                                          [CARRIAGE CONTROL,  7 HOLE TABLE.

11)          0           /       *6000 0104      [  00   /   01
       *6001 0104        /       *6002 0104      [  02   /   03
       *6003 0104        /       *6004 0104      [  04   /   05
       *6005 0104        /       *6006 0104      [  06   /   07
       *6007 0104        /       *6010 0104      [  10   /   11
       *6011 0104        /       *6012 0104      [  12   /   13
       *6013 0104        /       *6014 0104      [  14   /   15
       *6015 0104        /       *6016 0104      [  16   /   17

             0           /       *6000 0104      [  20   /   21
       *6021 0104        /       *6022 0104      [  22   /   23
       *6023 0104        /       *6024 0104      [  24   /   24
       *6025 0104        /       *6026 0104      [  26   /   27
       *6027 0104        /       *6030 0104      [  30   /   31
       *6031 0104        /       *6032 0104      [  32   /   33
       *6033 0104        /       *6034 0104      [  34   /   35
       *6035 0104        /       *6036 0104      [  36   /   37

       *6000 1144        /       *6000 1144      [  40   /   41
       *6000 1144        /       *6000 1144      [  42   /   43
       *6000 1144        /       *6000 1144      [  44   /   45
       *6000 1144        /       *6000 1144      [  46   /   47
       *6000 1144        /       *6000 1144      [  50   /   51
       *6000 1144        /       *6000 1144      [  52   /   53
       *6000 1144        /       *6000 1144      [  54   /   55
       *6000 1144        /       *6000 1144      [  56   /   57

             0           /       0               [  60   /   61
             0           /       0               [  62   /   63
             0           /       0               [  64   /   65
             0           /       0               [  66   /   67
             0           /       0               [  70   /   71
             0           /       0               [  72   /   73
             0           /       0               [  74   /   75
             0           /       0               [  76   /   77
```

2.9.63

```
                                                  [R573.8
                                                  [INNER SET 5 HOLE TABLE
13)   (97)                /      *3000  0000      [00  04
      *4000 1204          /      *5000  1344      [10  14
      *4000  0044         /      *4000  0204      [20  24
      *4000  0104         /      *4000  0504      [30  34
      *4000  1344         /      *5000  0204      [40  44
      *5000  0104         /      *5000  0304      [50  54
      *5000  0044         /      *5000  0244      [60  64
      *5000  0144         /      (97)             [70  74

      *4000  0704         /      *2000  0000      [01  05
      *4000  0604         /      (97)             [11  15
      *4000  1004         /      *4000  1244      [21  25
      *4000  1144         /      *4000  0544      [31  35
      *5000  1004         /      *5000  1204      [41  45
      *5000  1104         /      *5000  1304      [51  55
      *5000  1044         /      *5000  1244      [61  65
      *5000  1144         /      (97)             [71  75

      *4000  0704         /      *5000  1544      [02  06
      *4000  1704         /      *4000  1404      [12  16
      *4000  0404         /      *4000  0644      [22  26
      (97)                /      *4000  1504      [32  36
      *5000  0404         /      *5000  0604      [42  46
      *5000  0504         /      *5000  0704      [52  56
      *5000  0444         /      *5000  0644      [62  66
      *5000  0544         /      (97)             [72  76

      (97)                /      *4000  0004      [03  07
      *5000  0744         /      *4000  1644      [13  17
      *4000  1444         /      *4000  1604      [23  27
      *4000  1044         /      *1000  0344      [33  37
      *5000  1404         /      *5000  1604      [43  47
      *5000  1504         /      *5000  1704      [53  57
      *5000  1444         /      *5000  1644      [63  67
      (97)                /      (97)             [73  77

      (97)                =      *1000  0344      [Not printable
```

2.9.63

```
                                                        [R573.9

                                                        [OUTER SET, 5 HOLE TABLE.
14)    (97)                /          *3000 0000        [00 / 04
       (97)                /          (97)              [10 / 14
       *4000 0144          /          *4000 0444        [20 / 24
       (97)                /          (97)              [30 / 34
       (97)                /          *5000 0204        [40 / 44
       *5000 0104          /          *5000 0304        [50 / 54
       *5000 0044          /          *5000 0244        [60 / 64
       *5000 0144          /          (97)              [70 / 74

       *4000 0704          /          *2000 0000        [01 / 05
       (97)                /          (97)              [11 / 15
       (97)                /          *4000 1104        [21 / 25
       *4000 0304          /          (97)              [31 / 35
       *5000 1004          /          *5000 1204        [41 / 45
       *5000 1104          /          *5000 1304        [51 / 55
       *5000 1044          /          *5000 1244        [61 / 65
       *5000 1144          /          (97)              [71 / 75

       (97)                /          *5000 1544        [02 / 06
       (97)                /          (97)              [12 / 16
       (97)                /          (97)              [22 / 26
       (97)                /          (97)              [32 / 36
       *5000 0404          /          *5000 0604        [42 / 46
       *5000 0504          /          *5000 0704        [52 / 56
       *5000 0444          /          *5000 0644        [62 / 66
       *5000 0544          /          (97)              [72 / 76

       (97)                /          *4000 0004        [03 / 07
       (97)                /          (97)              [13 / 17
       *4000 0244          /          (97)              [23 / 27
       (97)                /          (97)              [33 / 37
       *5000 1404          /          *5000 1604        [43 / 47
       *5000 1504          /          *5000 1704        [53 / 57
       *5000 1444          /          *5000 1644        [63 / 67
       (97)                /          *1000 1744        [73 / 77
```

2.9.63

```
                                                [573.10

                                                [CARRIAGE CONTROL, 5 HOLE TABLE.

15)              0          /       *2000 1304  [  00  /  01
       *2001 1304           /       *2002 1304  [  02  /  03
       *2003 1304           /       *2004 1304  [  04  /  05
       *2005 1304           /       *2006 1304  [  06  /  07
       *2007 1304           /       *2010 1304  [  10  /  11
       *2011 1304           /       *2012 1304  [  12  /  13
       *2013 1304           /       *2014 1304  [  14  /  15
       *2015 1304           /       *2016 1304  [  16  /  17

       *2000 0744           /       *2001 0744  [  20  /  21
       *2002 0744           /       *2003 0744  [  22  /  23
       *2004 0744           /       *2005 0744  [  24  /  25
       *2006 0744           /       *2007 0744  [  26  /  27
       *2010 0744           /       *2011 0744  [  30  /  31
       *2012 0744           /       *2013 0744  [  32  /  33
       *2014 0744           /       *2015 0744  [  34  /  35
       *2016 0744           /       *2017 0744  [  36  /  37

       *2000 1304           /       *2000 1304  [  40  /  41
       *2000 1304           /       *2000 1304  [  42  /  43
       *2000 1304           /       *2000 1304  [  44  /  45
       *2000 1304           /       *2000 1304  [  46  /  47
       *2040 0744           /       *2041 0744  [  50  /  51
       *2042 0744           /       *2043 0744  [  52  /  53
       *2044 0744           /       *2045 0744  [  54  /  55
       *2046 0744           /       *2047 0744  [  56  /  57

                 0          /        0          [  60  /  61
                 0          /        0          [  62  /  63
                 0          /        0          [  64  /  65
                 0          /        0          [  66  /  67
                 0          /        0          [  70  /  71
                 0          /        0          [  72  /  73
                 0          /        0          [  74  /  75
                 0          /        0          [  76  /  77
```

2.9.63

```
                                  R 575                     [Card punch fault testing


                 (62)     =       (62/599)
                 (60)     =       (60/599)
                 (65)     =       (65/599)
                 (68)     =       (68/599)

        1)       101,     106,    102,     *6
                 165,     107,    106,     0.2
                 214,     126,    107,     (9/508)          [Exit if started

                 121,     110,    0,       (7/508)
                 121,     109,    0,       (3/514)
                 165,     108,    106,     *0004 0000
                 214,     126,    108,     (6)
                 121,     108,    0,       1.0              [Disabled M = 1.0
                 121,     126,    0,       (3)

        6)       165,     108,    106,     *0002 0000
                 214,     126,    108,     (5)
                 121,     107,    0,       *0004 0003       [Put out overdue,stop and disengage
                 121,     108,    0,       2.0              [Overdue, M= 2.0
                 121,     126,    0,       (4)

        5)       165,     108,    106,     *0010 0000
                 214,     126,    108,     (7)

        2)       121,     108,    0,       3.4              [Out of cards, M= 3.4
                 121,     126,    0,       (3)

        7)       121,     109,    0,       (47/579)         [No fault
                 121,     126,    0,       (7/508)

        15)      101,     106,    100,     (65)*7           [Read number of check fails
                 122,     106,    0,       0.4
                 215,     126,    106,     (16)             [Jump if cards to be repunched
                 121,     110,    0,       (3/514)
                 101,     102,    100,     (68)*7

        3)       121,     107,    0,       0.3
        4)       113,     107,    102,     *6               [Stop and disengage punch

        13)      101,     106,    100,     (62)*7           [Alter (62) to beginning
                 165,     107,    106,     *7777 7776       [  of present card
        12)      122,     107,    0,       2.0
                 101,     106,    107,     *7
                 127,     106,    0,       *0000 7777
                 214,     126,    106,     (12)
                 124,     107,    0,       2.0
                 113,     107,    100,     (62)*7
        14)      121,     126,    110,     0

        16)      124,     106,    0,       0.5
                 113,     106,    100,     (65)*7           [Count of check fails
                 121,     110,    0,       2(0)
                 121,     126,    0,       (13)
                 122,     107,    0,       26.0             [Reset to beginning
                 113,     107,    100,     (62)*7           [  of previous card
                 121,     126,    0,       (48/579)

                                                               2.9.63
        R576                                         [Card punch, Punch Row Interruption

        (72)=*6004 2200                              [V-store address
        (70)=*6004 3460                              [L.A.M. for Card Punch
        (78)=(5/599)10.4                             [Private store address
        (62)=(62/599)
        (99)=(99/599)

1)      101,    123,    0,      (70)
        165,    115,    123,    1.0
        101,    125,    123,    (2)

2)      (1/578)         /       (99)                 [End of card Punch 0
        (99)            /       (99)                 [ ''  ''    ''    '' 1
        (3)             /       (99)                 [Punch row Punch 0
        (99)            /       (99)                 [ ''    ''    '' 1
        (1/577)         /       (99)                 [Check Row Punch 0
        (99)            /       (99)                 [ ''    ''    '' 1
        (99)            /       (99)
        (99)            /       (99)
        (1/500)         /       (99)

3)      101,    111,    123,    (78)-2.0             [Pick up private store address

5)      101,    113,    111,    (62)*7               [1st half word of card
        121,    114,    113,    2.0
        113,    114,    111,    (62)*7               [Increase bufffer address
        101,    112,    113,    *7
        165,    114,    112,    6.0
        113,    112,    123,    -2.0(72)             [Punch 1st half word
        101,    112,    113,    0.4*7
        113,    112,    123,    2.0(72)              [2nd half word
        101,    112,    113,    1.0*7
        113,    112,    123,    6.0(72)              [3rd half word
        101,    112,    113,    1.4*7
        113,    112,    123,    10.0(72)             [4th half word

        121,    116,    0,      0.4
        113,    116,    123,    -2.0(72)             [Put out L.A.M.
        214,    125,    114,    (1/500)              [Exit to M/E unless fault
        113,    113,    111,    (62)*7
        121,    125,    0,      (14/578)
```

2.9.63

```
            R577                                [Card Punch, Check Read Interruption


          (53)=(53/599)
          (62)=(62/599)
          (72)=*6004 2200                       [V store address
          (78)=(5/599)10.4                      [Private store address

   1)     101,     111,     123,     -4.0(78)   [Pick up private store address
          101,     113,     111,     (62)*7     [Pick up buffer address
          121,     118,     0,       0
          211,     125,     113,     (2)        [No check as first card
          122,     113,     0,       28.0

          101,     117,     113,     *7         [Pick up ½word from store
          165,     118,     117,     6.0
          106,     117,     123,     -4.0(72)
          127,     117,     0,       *7760      [And out required bits.
          147,     117,     111,     (53)*7

          101,     114,     123,     (72)
          106,     114,     113,     0.4*7
          167,     117,     114,     0

          101,     114,     123,     4.0(72)
          106,     114,     113,     1.0*7
          167,     117,     114,     0

          101,     114,     123,     8.0(72)
          106,     114,     113,     1.4*7
          167,     117,     114,     0
          113,     117,     111,     (53)*7
   2)     121,     116,     0,       *0010 0000
          113,     116,     123,     -4.0(72)   [Put out L.A.M.
          214,     125,     118,     (1/500)    [Exit to M/E Unless error
          121,     125,     0,       (14/578)



                                                      2.9.63
```

```
        R578                                           [Card Punch end of card interruption

        (78)     =      (5/599)10.4
        (72)     =      *6004 2200
        (53)     =      (53/599)
        (62)     =      (62/599)
        (51)     =      (51/599)
        (65)     =      (65/599)

1)      101,     111,   123,     (78)                  [Pick up private store address
        101,     113,   111,     (62)*7                [Pick up next buffer address
        121,     114,   113,     2.0
        101,     117,   113,     *7
        127,     117,   0,       6.0
        122,     117,   0,       6.0
        214,     125,   117,     (15)                  [Jump if end of card

14)     121,     112,   0,       0.4                   [Not at end of card
        113,     112,   111,     (65)*7
13)     121,     112,   0,       *0010 0022            [Stop and put out all LAM's
        113,     112,   115,     (72)
        121,     112,   0,       2.4
        113,     112,   111,     (51)*7                [Mark for check fail
        121,     112,   0,       (15/575)
        121,     125,   0,       (4/201)

15)     211,     125,   113,     (12)                  [Jump if first card
        101,     112,   111,     (53)*7
        215,     125,   112,     (13)                  [Jump if check fail
        113,     0,     111,     (65)*7                [Clear for next card
        121,     125,   0,       (11)

12)     121,     112,   0,       -4.0
        167,     114,   0,       0.1
11)     113,     114,   111,     (62)*7
        104,     112,   113,     *7                    [Pick up last ½ word of card
        113,     112,   123,     (72)                  [P O LAM do not offset
        127,     112,   0,       0.2
        214,     125,   112,     (1/500)               [Exit to M/E unless stopping
        121,     112,   0,       -0.4
        113,     112,   111,     (51)*7                [Set M
        121,     112,   0,       (1/579)               [Call PER to queue
        121,     125,   0,       (4/201)
```

2.9.63

```
            R579                                          [R579.1

                                                          [Extracode for card punch


        (60) =(60/599)
        (65) =(65/599)
        (53)= (53/599)
        (62) =(62/599)
        (56)=(56/599)
        (50)=(50/599)
        (61)=(61/599)
        (67)=(67/599)
        (52)=(52/599)
        (80)=(80/599)
        (81)=(81/599)
        (90)=(90/599)
        (99)=(99/599)


1)      101,    109,    100,    (62)*7                    [Transfer last complete card and
        121,    105,    0,      51.4                      [incomplete card, if any, to beginning
        122,    109,    0,      26.0                      [of buffer
        101,    107,    100,    (60)*7
        101,    106,    109,     *7
        113,    106,    107,     *7
        124,    109,    0,       0.4
        124,    107,    0,       0.4
        202,    126,    105,    -4.0(0)


        101,    107,    100,    (67)*7
        122,    107,    0,       40.0
        216,    126,    107,    -1(0)
        124,    107,    0,       40.0
        113,    107,    100,    (67)*7                    [Reset(67) to beginning of buffer
        101,    102,    100,    (56)*7                    [characters left in present record
        104,    102,    100,    (50)*7                    [carriage control characters
        214,    126,    102,    (1/514)                   [exit if output finished
2)      121,    110,    0,       (3)
        121,    126,    0,      (1/520)                   [Bring block to cores
3)      121,    110,    0,      2(0)
        121,    126,    0,      (2/520)
        113,    102,    0,      (80)

        101,    103,    100,    (60)*7
        124,    103,    0,      26.0
        121,    105,    103,    -2.0
        121,    107,    0,      6.0
        113,    107,    105,    *7                        [Mark end of first card in buffer
        101,    104,    100,    (67)*7
        121,    105,    104,     0
        215,    126,    105,    (7)                       [Jump if card still in buffer
```

2.9.63

```
4)  121,   102,   0,     *001        [Mark for column of card
    121,   109,   0,     1.4         [Mark for ½ word of card
    113,   109,   0,     (81)
    121,   109,   103,   25.4
    102,   109,   100,   (61)*7
    216,   126,   109,   (30)        [Jump if buffer full
    121,   109,   0,     25.4
    113,   0,     103,   *7          [Clear buffer for next card
    124,   103,   0,     0.4
    202,   126,   109,   -2(0)
    121,   109,   0,     6.0
    122,   103,   0,     2.0
    113,   109,   103,   *7
    122,   103,   0,     24.0
    121,   126,   0,     (5)

7)  121,   109,   0,     1.4         [VE MARK TO CORRECT POSITION
    124,   104,   0,     0.4         [  along partial card
    122,   105,   0,     4.0
    121,   102,   0,     *4
    217,   126,   105,   (8)
9)  122,   105,   0,     12.0
    124,   103,   0,     0.4         [Increase to next ½ word of buffer
    122,   109,   0,     0.4
    216,   126,   105,   (9)
8)  124,   105,   0,     0.4
    214,   126,   105,   3(0)

    163,   102,   0,     0
    121,   126,   0,     (8)
    113,   109,   0,     (81)
5)  101,   105,   100,   (56)*7
    124,   105,   105,   0
    124,   105,   105,   0           [No of characters to punch
    121,   110,   0,     2(0)        [Set code conversion parameters
    121,   126,   0,     (1/516)
    121,   110,   0,     (16)
    210,   110,   109,   (27)        [Binary
    164,   110,   101,   0.3
    101,   107,   101,   *7          [First half word
    211,   126,   109,   (2/512)     [Shift up first half word (int code)
    121,   126,   0,     (1/512)     [Ditto                    (Binary)
```

```
21)  124,   101,   0,     0.1            [Advance main store address
16)  202,   126,   106,   (18)           [Jump unless beginning ½ word

     101,   109,   101,   *7
     121,   106,   0,     1.4
     165,   107,   109,   *0303 0303
     125,   107,   0,     0
     164,   107,   109,   *7474 7474

18)  125,   107,   0,     0
     165,   109,   107,   *0000 0374
     202,   126,   105,   (2/517)        [Return to (20) or (21)
     113,   0,     100,   (56)*7         [End of characters
     121,   126,   0,     (1/527)        [Go to find carriage control character
                                         [ return to (31) (32) or (36)

19)  113,   110,   0,     (81)           [Store mark for ½word in buffer
24)  125,   109,   0,     0
23)  125,   109,   0,     0
     165,   110,   109,   *0000 0760     [And out row of 1st hole
     214,   126,   110,   (21)           [Exit if no more holes
     124,   110,   103,   -2.0           [Add on buffer address
     114,   102,   110,   *7             [Add mark into store
     211,   126,   126,   (23)0.1        [Repeat for 2nd hole
     125,   109,   0,     0
22)  127,   109,   0,     *0000 1777     [And out bits for 10 remaining holes
     214,   126,   109,   (21)           [Exit if no more holes
     124,   110,   0,     2.0            [Increase address
     211,   126,   109,   2(0)           [Jump if no hole
     114,   102,   110,   *7             [Add into buffer
     163,   109,   0,     0              [Shift down one place
     121,   126,   0,     (22)

20)  124,   102,   102,   0              [Shift mark
     124,   104,   0,     0.4            [Increase count of characters
     215,   126,   102,   (24)           [Plant character if room in ½word
     124,   103,   0,     0.4
     121,   102,   0,     0.1            [Reset for next ½word of card
     101,   110,   0,     (81)
     202,   126,   110,   (19)           [Jump unless card full
25)  124,   103,   0,     24.0           [Increase buffer to beginning
                                         [  of next card

     122,   104,   0,     0.4            [Correct character count
     163,   105,   0,     0
     163,   105,   0,     -0.1
     113,   105,   100,   (56)*7         [Store character count
     121,   126,   0,     (4)            [Jump to test if room for next card
```

```
39)  (10)    /      (11)
     (99)    /      (99)
     (20)    /      (99)
     (21)    /      (21)
     (21)    /      (99)
     (12)    /      (31)
     (31)    /      (32)
     (36)    /      (99)

                                    [Binary
40)  113,    110,   0,      (81)    [Mark for ½word of card
28)  121,    110,   103,    0       [Buffer address
     216,    126,   109,    2(0)
     114,    102,   110,    *7      [Store in buffer
     124,    110,   0,      2.0
     124,    109,   109,    0
     215,    126,   109,    1(28)   [Jump unless blank column
     124,    101,   0,      0.2     [Increase store address

27)  203,    126,   106,    (26)
     101,    107,   101,    *7
     121,    106,   0,      1.0
     121,    126,   0,      2(26)

26)  125,    107,   0,      0
     125,    107,   0,      0
     165,    109,   107,    *7777   [And out characters
     203,    126,   105,    (29)
     121,    126,   0,      3(18)   [End of record

29)  124,    104,   0,      0.4
     124,    102,   102,    0
     215,    126,   102,    (28)
     124,    103,   0,      0.4     [Increase to next ½word of card
     121,    102,   0,      0.1
     101,    110,   0,      (81)
     202,    126,   110,    (40)
     124,    105,   0,      0.4
     121,    126,   0,      (25)    [Jump to test if room for next card
```

```
31)   101,   109,   0,     (81)        [Card feed caused by end of record
      124,   103,   109,   24.4        [Increase buffer to end of card
      104,   104,   109,   (37)
      124,   104,   0,     0.4         [Increase buffer counter to end of card
      124,   102,   102,   0
      215,   126,   102,   -2(0)
      121,   126,   0,     (4)         [Test if room for next card


32)   124,   101,   0,     0.1         [Increase main store address
36)   121,   110,   0,     (33)
      121,   126,   0,     (1/521)     [Go to read next separator
33)   215,   126,   108,   (5)         [Return unless zero separator
      122,   102,   0,     *001
      215,   126,   102,   4(0)
      121,   105,   0,     1.4
      102,   105,   9,     (81)
      214,   126,   105,   (30)
      122,   104,   0,     0.4
      101,   109,   0,     (81)
      102,   103,   109,   (45)
30)   122,   103,   0,     2.0
38)   121,   110,   0,     2(0)
      121,   126,   0,     (1/518)     [Go to store code parameters
      113,   104,   100,   (67)*7      [Store no of characters to punch
      122,   104,   0,     40.0
      101,   102,   0,     (80)
46)   121,   110,   0,     2(0)
      121,   126,   0,     (4/523)     [Remove reserved block label
      217,   126,   104,   (1/514)     [Return to output master routine
                                       [if no characters to punch
      121,   104,   0,     6.2
      113,   104,   103,   *7          [Stop bit at end of buffer
      101,   109,   100,   (62)*7
      127,   109,   0,     0.1
      124,   109,   0,     26.0
      104,   109,   100,   (60)*7
      113,   109,   100,   (62)*7
47)   113,   0,     100,   (65)*7
48)   121,   109,   0,     *00160001
      113,   0,     100,   (53)*7
      121,   126,   0,     (2/515)     [Start punch

37)   -0.4   /      11.4
      23.4   /      35.4

45)   1.4    /      1.0
      0.4    /      0
```

```
                                                  [R579.6

10)            (98)     /       *3000 0000        [00 / 04
        *4061 6010      /       *4042 0004        [10 / 14
        *4060 0000      /       *4160 0000        [20 / 24
        *4260 0000      /       *4142 6000        [30 / 34
        *4162 6000      /       *4021 6000        [40 / 44
        *4022 6000      /       *4041 4000        [50 / 54
        *4042 4000      /       *4061 4000        [60 / 64
        *4062 4000      /       (98)              [70 / 74

        *4000 0000      /       *2000 0000        [01 / 05
        *4021 6010      /       *4202 6000        [11 / 15
        *4100 0000      /       *4200 0000        [21 / 25
        *4300 0000      /       *4020 0000        [31 / 35
        *4021 0000      /       *4022 0000        [41 / 45
        *4023 0000      /       *4041 6000        [51 / 55
        *4042 6000      /       *4061 6000        [61 / 65
        *4062 6000      /       (98)              [71 / 75

        *4000 0000      /       0                 [02 / 06
        *4061 4020      /       *4041 6010        [12 / 16
        *4120 0000      /       *4220 0000        [22 / 26
        *4062 0004      /       *4040 0000        [32 / 36
        *4021 2000      /       *4022 2000        [42 / 46
        *4041 0000      /       *4042 0000        [52 / 56
        *4043 0000      /       *4062 0000        [62 / 66
        *4063 0000      /       (98)              [72 / 76

               (98)     /       0                 [03 / 07
        *4041 4020      /       *4061 0000        [13 / 17
        *4140 0000      /       *4240 0000        [23 / 27
        *4022 0004      /       *4021 4020        [33 / 37
        *4021 4000      /       *4022 4000        [43 / 47
        *4041 2000      /       *4042 2000        [53 / 57
        *4061 2000      /       *4062 2000        [63 / 67
               (98)     /       (98)              [73 / 77

               (98)     =       *4021 4020        [Fullstop
```

2.9.63

```
11)     (98)                /     *3000 0000  [00 / 04
        (98)                /     (98)        [10 / 14
        (98)                /     (98)        [20 / 24
        (98)                /     (98)        [30 / 34
        (98)                /     *4021 6000  [40 / 44
        *4022 6000          /     *4041 4000  [50 / 54
        *4042 4000          /     *4061 4000  [60 / 64
        *4062 4000          /     (98)        [70 / 74

        *4000 0000          /     *2000 0000  [01 / 05
        (98)                /     (98)        [11 / 15
        *4042 4001          /     (98)        [21 / 25
        (98)                /     (98)        [31 / 35
        *4021 0000          /     *4022 0000  [41 / 45
        *4023 0000          /     *4041 6000  [51 / 55
        *4042 6000          /     *4061 6000  [61 / 65
        *4062 6000          /     (98)        [71 / 75

        (98)                /     0           [02 / 06
        (98)                /     (98)        [12 / 16
        *4042 2002          /     *4022 2002  [22 / 26
        (98)                /     (98)        [32 / 36
        *4021 2000          /     *4022 2000  [42 / 46
        *4041 0000          /     *4042 0000  [52 / 56
        *4043 0000          /     *4062 0000  [62 / 66
        *4063 0000          /     (98)        [72 / 76

        (98)                /     0           [03 / 07
        (98)                /     *4222 6000  [13 / 17
        (98)                /     *4022 4001  [23 / 27
        (98)                /     (98)        [33 / 37
        *4021 4000          /     *4022 4000  [43 / 47
        *4041 2000          /     *4042 2000  [53 / 57
        *4061 2000          /     *4062 2000  [63 / 67
        (98)                /     (98)        [73 / 77
```

23.10.63

```
                                               [R579.8
                                               [Carriage control table
12)    0       /       *6004                   [ 00 / 01
     *6001     /       *6002                   [ 02 / 03
     *6003     /       *6004                   [ 04 / 05
     *6005     /       *6006                   [ 06 / 07
     *6007     /       *6010                   [ 10 / 11
     *6011     /       *6012                   [ 12 / 13
     *6013     /       *6014                   [ 14 / 15
     *6015     /       *6016                   [ 16 / 17

       0       /       *6000                   [ 20 / 21
     *6021     /       *6022                   [ 22 / 23
     *6023     /       *6024                   [ 24 / 25
     *6025     /       *6026                   [ 26 / 27
     *6027     /       *6030                   [ 30 / 31
     *6031     /       *6032                   [ 32 / 33
     *6033     /       *6034                   [ 34 / 35
     *6035     /       *6036                   [ 36 / 37

     *6000     /       *6000                   [ 40 / 41
     *6000     /       *6000                   [ 42 / 43
     *6000     /       *6000                   [ 44 / 45
     *6000     /       *6000                   [ 46 / 47
     *6000     /       *6000                   [ 50 / 51
     *6000     /       *6000                   [ 52 / 53
     *6000     /       *6000                   [ 54 / 55
     *6000     /       *6000                   [ 56 / 57

     0/0                                       [ 60 / 61
     0/0                                       [ 62 / 63
     0/0                                       [ 64 / 65
     0/0                                       [ 66 / 67
     0/0                                       [ 70 / 71
     0/0                                       [ 72 / 73
     0/0                                       [ 74 / 75
     0/0                                       [ 76 / 77
```

2.9.63

[R585

R585

[Teleprinter fault testing routine

(1)=(1/570)

2.9.63

```
                                                  [R586

      R586                                        [Teleprinter interruption

      (74)=(5/599)13.0                            [Subsidiary store address
      (62)=(62/599)                               [     of teleprinter 0
      (51)=(51/599)

1)    101,    123,    0,      *6004 3410          [Which teleprinter
      101,    111,    123,    (74)                [Find private store
      101,    113,    111,    (62)*7              [Address of this character
      121,    114,    113,    0.4                 [Address of next character
      113,    114,    111,    (62)*7
      101,    112,    113,    *7                  [Pick up character and POLAM
      113,    112,    123,    *6004 2600
      127,    112,    0,      0.2
      214,    125,    112,    (1/500)             [Exit unless stopping
      121,    112,    0,      -0.4
      113,    112,    111,    (51)*7
      121,    112,    0,      (1/573)
      121,    125,    0,      (4/201)             [Call PER to queue
```

2.9.63

```
R590                                                    |Peripheral extracode linkage

        (3) = (4/247)                                   |Exit to main
        (4) = (1/201)                                   |Enter sup.
        (5) = (1/591)                                   |Select input/output
        (6) = (1/215)                                   |Set fr switch
        (7) = (24/227)                                  |Set extracode PAR
        (8) = (35/203)                                  |Block location table
        (9) = (2/318)                                   |Call to cores
        (12) = (12/227)                                 |Input/Output records
        (13) = *3665                                    |Input label
        (14) = *3660                                    |Output label
        (15) = (6/203)                                  |Page directory
        (16) = (19/591)                                 |Select first stream

1)      113,    0,      0,      *7(10/201)
        113,    97,     0,      (6/201)
        127,    97,     0,      -0.2
        121,    96,     126,    1.0
        121,    126,    0,      3(4)                    |Enter Supervisor
        121,    101,    0,      (5)
        211,    101,    96,     (3)                     |Set link
2)      165,    100,    96,     0.4                     |SR Call block to cores
        121,    109,    100,    2.4(0)
        121,    126,    0,      (6)                     |Set fr switch
        121,    108,    0,      (13)*4
        215,    108,    100,    (14)*4
        121,    109,    0,      2(0)
        121,    126,    0,      (7)                     |Check & set PAR
        217,    126,    107,    (16)                    |Exit if more selected
        216,    126,    106,    6(0)
        113,    107,    0,      (8)
        125,    110,    0,      *01
        105,    110,    100,    (12)
        165,    108,    110,    *07757775
        121,    126,    0,      (9)                     |Call to cores
        121,    109,    0,      0.1
        147,    109,    106,    (15)                    |Remove
        113,    109,    106,    (15)                    |Lock out Page Directory
        121,    126,    101,    0                       |Exit form subroutine

                                                        |11/7/63.
```

```
                                        [R595.I

R595                                    [Input extracodes

(70) = 8.0*7
(7I) = I(70)
(72) = 2(70)
(73) = 3(70)
(74) = 4&70)
(75) = 5(70)
(76) = 6(70)
(78) = *7
```

2.9.63

```
                                                    [R595.2

         (0) = *4000 0540

 1)      101,    126,    0,      (72)               [EXTRACODE 1054  Enter at (1)
                                                    [Jump to (2),(2)1,(11), or (14)
         (0) = 55*4010

 2)      121,    126,    0,      (5)0.1             [Jump to find next half word
         121,    94,     0,      0.1                [Subtract 0.1 from multiway
         110,    94,     0,      (72)               [    jump address
 3)      101,    94,     0,      (71)               [Half word
 4)      125,    94,     0,      0                  [Shift up
         113,    94,     0,      (71)               [Preserve for next time
         217,    126,    127,    (17)               [Exit on extracode control
         565,    122,    94,     7.7

 5)      101,    94,     0,      (75)               [Present address
         121,    95,     94,     0.4
         113,    95,     0,      (75)               [Next address
         102,    95,     0,      (76)               [Next address - end address
         121,    96,     0,      (2)1.2
         113,    96,     0,      (72)
 6)      113,    126,    0,      3*6                [INHIBIT INTERRUPTIONS
         121,    125,    0,      (8)0.2
         113,    125,    0,      3*6
         121,    97,     0,      (6)0.1
         121,    126,    0,      (1/590)0.0         [Call current input block

 8)      113,    126,    0,      (79)               [Preserve B126 in case of
                                                    [   non-equivalence
         101,    94,     94,     0                  [Pick up half word
         121,    126,    0,      (4)
         216,    126,    95,     (9)                [Return to (4) unless last
         113,    0,      0,      3*6                [    half word in record

 9)      122,    95,     0,      0.4                [If no character in record at all
         216,    126,    95,     (14)               [    get next record
         113,    94,     0,      (71)
         165,    94,     95,     0.3
         120,    94,     0,      (11)0.3            [Reset multiway jump address
         113,    94,     0,      (72)               [    including l.s. bits
         121,    126,    94,     0
11)      165,    94,     126,    0.3
         215,    126,    94,     (2)1               [Transfer remaining characters
         101,    126,    0,      (73)               [Jump to (12) if true end of record
                                                    [         (13) if record continues
12)      217,    126,    127,    (15)               [SPECIAL EXIT ON EXTRACODE CONTROL
         121,    127,    119,    0                  [c'=n
13)      121,    94,     0,      (14)0.7            [Arrange to read separator
         113,    94,     0,      (72)               [     next time
         121,    126,    0,      (3)                [Transfer last character

14)      121,    94,     0,      (5)0.1
         121,    126,    0,      (50)0.1

15)      121,    94,     0,      (14)0.7
         113,    94,     0,      (72)
         101,    94,     0,      (71)
         125,    94,     0,      0
         127,    94,     0,      7.7
16)      121,    126,    91,     0                  [Exit on extracode control

17)      127,    94,     0,      7.7
18)      121,    126,    127,    0                  [Exit on extracode control
                                                                 2.9.63
```

```
                                                 [R595.3

                                                 [EXTRACODE 1056 Enter at (20)
20)   113,   122,   0,    (78)
      101,   92,    0,    (78)
      127,   92,    0,    *3777 7777             [Number of characters required
      121,   126,   0,    (22)
                                                 [EXTRACODE 1057 Enter at (21)
21)   121,   92,    0,    *3777 7774
22)   127,   119,   0,    *7777 7774             [Address in programmer's store
      121,   93,    0,    0                      [To accumulate total transferred

23)   101,   91,    0,    (75)
      101,   94,    0,    (76)
      120,   94,    91,   0                      [Length of record (negative)
      101,   95,    0,    (72)
      127,   95,    0,    0.3                    [Set B95 non zero if half
      126,   95,    0,    0.3                    [    word already started
      217,   126,   94,   (26)                   [Jump provided record not finished
      215,   126,   95,   (26)
      101,   95,    0,    (73)
      122,   95,    0,    (12)0.4
      216,   126,   95,   (25)                   [Jump if not true end of record
      214,   126,   93,   (25)                   [   or record not started yet
24)   521,   122,   93,   0                      [End of record EXIT

25)   121,   94,    0,    (23)
      121,   126,   0,    (50)0.1

26)   215,   126,   92,   (27)
      521,   122,   93,   *4                     [If no characters required EXIT

27)   215,   126,   95,   (60)                   [Transfer character by character if
                                                 [    NOT BEGINNING HALF WORD
      123,   95,    94,   0                      [Length remaining
      124,   94,    92,   0                      [B94 contains number of characters
      217,   94,    94,   0                      [      required, or number available
      120,   94,    92,   0                      [    whichever is smaller
      127,   94,    0,    *0000 7774             [Number of half words to be transferr
      214,   126,   94,   (60)                   [Jump if no complete half words
      122,   95,    94,   0
      215,   126,   95,   (28)                   [If record will be exactly
      121,   95,    0,    (14)0.7                [     finished, reset jump for
      113,   95,    0,    (72)                   [     extracode 1054
28)   124,   93,    94,   0                      [Add to total
      122,   92,    94,   0                      [Subtract from number required
      121,   126,   0,    (39)                   [Transfer half words. Return to (23)
```

2.9.63

```
                                                    [R595.4

30)     I2I,    95,     94,     -3.4               [B95 contains B94 or 3.4
        2I6,    95,     95,     0                  [   whichever is the smaller
        I24,    95,     0,      3.4
        I65,    96,     95,     3.4                [Copy to B96
        II3,    I26,    0,      3*6                [INHIBIT INTERRUPTIONS
        I2I,    I25,    0,      (32)0.2
        III,    I25,    0,      3*6                [Interrupt control
        I2I,    97,     0,      (30)0.I
        I2I,    I26,    0,      (I/590)0.0         [Call current input block

32)     II3,    I26,    0,      (79)               [Preserve BI26 in case of
                                                   [  non-equivalence
        I2I,    I25,    0,      (35)

34)     I0I,    97,     9I,     0                  [Transfer from supervisor to
        II3,    97,     95,     0.4(78)            [    subsidiary store
        I24,    9I,     0,      0.4
35)     202,    I25,    95,     (34)
        I22,    94,     96,     0                  [Subtract from number remaining
        I2I,    I26,    0,      (38)

        II3,    0,      0,      3*6                [Resume extracode control

37)     I0I,    97,     96,     0.4(78)            [Transfer from subsidiary store
        II3,    97,     II9,    0                  [    to programme store
        I24,    II9,    0,      0.4
38)     202,    I26,    96,     (37)

39)     2I5,    I26,    94,     (30)0.I
        II3,    9I,     0,      (75)
        I2I,    I26,    0,      (23)
```

```
                                                  [R595.5

40)    165,    91,     119,    *7777 7770         [EXTRACODE 1050  Enter at (40)
41)    101,    95,     0,      (70)
       122,    95,     91,     0
       121,    97,     0,      (41)
       215,    126,    95,     (1/591)            [Select new input stream
42)    121,    126,    0,      (4/596)

       (0) = *4000 0510

43)    501,    122,    0,      (70)               [EXTRACODE 1051   Enter at (43)

       (0) = *4000 0520

44)    501,    122,    0,      (74)               [EXTRACODE 1052  Enter at (44)

       (0) = 1(42)

45)    101,    95,     0,      (72)               [EXTRACODE 1053   Enter at (45)
       127,    95,     0,      0.3
       126,    95,     0,      0.3                [Test if any characters left
       215,    126,    95,     (46)               [    in half word
       101,    95,     0,      (75)               [If none, test if any half
       102,    95,     0,      (76)               [    words left in record
       121,    94,     0,      (45)
       216,    126,    95,     (50)0.1            [If none, get next record
46)    101,    94,     0,      (73)
       211,    126,    94,     (4/596)            [Test binary / internal code
       217,    126,    127,    (16)               [Exit on extracode control
       521,    122,    119,    0                  [Binary: ba' = n
```

2.9.63

```
51)    I2I,    97,    0,     *7777 7774        [Return with new block: clear
       II7,    97,    0,     (76)              [    l.s bits
                                               [Find next record. Enter at (50)0.I
50)    I0I,    97,    0,     (76)              [End address of previous record
       I24,    97,    0,     0.3               [Round up: equals address of
       I27,    97,    0,     *7777 7774        [    next separator
       II3,    I26,   0,     3*6               [INHIBIT INTERRUPTIONS
       I2I,    I25,   0,     (53)0.2
       III,    I25,   0,     3*6               [Interrupt control

52)    I2I,    97,    0,     (50)0.I
       I2I,    I26,   0,     (I/590)0.0        [Call current input block

53)    II3,    I26,   0,     (79)              [Preserve BI26 in case of
                                               [  non-equivalence

       I0I,    96,    97,    0                 [Pick up separator
       I2I,    I26,   0,     (54)
       II3,    0,     0,     3*6               [Resume extracode control

54)    2I5,    I26,   96,    (56)              [Jump unless zero separator
       I2I,    97,    0,     (5I)0.I
       I2I,    I26,   0,     (I/592)           [Advance to next input block

56)    I65,    95,    96,    *0004
       2I4,    I26,   95,    (57)              [If separator has bit I4 = I, copy
       II3,    96,    0,     (74)              [   to (74) and ignore character
       I2I,    96,    0,     *3                [   count

57)    I65,    95,    97,    *0000 7777        [Monitor if next separator is
       I64,    95,    96,    *0000 7777        [   not in the same block
       I24,    95,    0,     *7777 0007
       2I6,    I26,   95,    (99)
       I24,    97,    0,     0.4
       II3,    97,    0,     (75)              [Next address
       I64,    97,    96,    *0000 7777
       II3,    97,    0,     (76)              [End address
       I65,    97,    96,    *I
       2I5,    97,    97,    (I3)              [Record continues
       2I4,    97,    97,    (I2)              [True end of record
       2I6,    96,    96,    0
       2I7,    96,    96,    0.I
       I24,    96,    97,    0
       II3,    96,    0,     (73)              [Set jump addresses
       I2I,    97,    0,     (2)0.7
       II3,    97,    0,     (72)

       I2I,    I26,   94,    0                 [Return
```

```
                                                          [R595.7

60)      II3,     I27,     0,       (78)                  [Preserve main control
         I2I,     9I,      0,       (62)0.I               [Return addrress at end of record
         I2I,     I27,     0,       (62)                  [Return for address for character
         I0I,     I26,     0,       (72)
62)      I0I,     95,      II9,     0
         I27,     95,      0,       *0077 7777
         I25,     95,      94,      0                     [Send characterto
         II3,     95,      II9,     0                     [    programme store
         I22,     92,      0,       0.I                   [Count down number required
         I24,     93,      0,       0.I                   [Count number sent
         I24,     II9,     0,       0.I                   [Advance address
         I65,     96,      II9,     0.3
         2I0,     I26,     I26,     (66)                  [Exit at end of record
         2I4,     I26,     92,      (66)                  [Exit after last character
         2I4,     I26,     96,      (67)                  [Exit if half word is full
         I0I,     I26,     0,       (72)                  [Enter I054

65)      I0I,     95,      II9,     0
         I25,     95,      0,       0
         II3,     95,      II9,     0
         I24,     96,      0,       0.I
         I27,     96,      0,       0.3
66)      2I5,     I26,     96,      (65)                  [Shift up odd characters
67)      I0I,     I27,     0,       (78)                  [recover main control
         I2I,     I26,     0,       (23)
```

2.9.63

```
                                                          [R596.I

        R596                                              [Output extracodes

        (70) = 8.4*7
        (7I) = I(70)
        (72) = 2(70)
        (73) = 3(70)
        (74) = 4(70)
        (75) = 5(70)
        (76) = 6(70)
        (78) = *7

        (0) = *40I7

I)      165,    94,     II9,    7.7              [EXTRACODE I064 Enter at (I)
2)      IOI,    95,     0,      (7I)
        2I7,    I26,    95,     (5)0.I           [Jump if half word will be full
        I25,    95,     94,     0                [Add character in
3)      II3,    95,     0,      (7I)
4)      2I7,    I26,    I27,    (I8/595)         [Exit under extracode control
        52I,    0,      0,      0                [Exit to main control

5)      I25,    95,     94,     -4.0             [Shift up and remove old sign bit
        IOI,    94,     0,      (75)             [present address

6)      II3,    I26,    0,      3*6              [INHIBIT INTERRUPTIONS
        I2I,    I25,    0,      (7)0.2
        II3,    I25,    0,      3*6
        I2I,    97,     0,      (6)0.I
        I2I,    I26,    0,      (I/590)0.4       [Call output block

7)      II3,    I26,    0,      (79)             [Preserve BI26 in case of
                                                 [   non-equivalence
        II3,    95,     94,     0                [Half word to store
        I2I,    I26,    0,      (8)
        II3,    0,      0,      3*6

8)      I24,    94,     0,      0.4
        II3,    94,     0,      (75)             [Next address
9)      I02,    94,     0,      (76)             [Next - end address
        I2I,    95,     0,      4.0
        2I7,    I26,    94,     (3)              [Return unless end of store reached
        I2I,    94,     0,      (I0)
        I2I,    I26,    0,      (50)             [End of store Insert separator

I0)     IOI,    94,     0,      (75)
        I2I,    I26,    0,      (9)
```

2.9.63

```
20)    121,   92,    0,    *4           [EXTRACODE 1066 Enter at (20)
       121,   126,   0,    (22)

21)    121,   92,    0,    0            [EXTRACODE 1067 Enter at (21)
22)    113,   122,   0,    (78)
       101,   93,    0,    (78)
       127,   92,    93,   0            [Bit 23 = 0 if record ends
       127,   93,    0,    *3777 7777   [Number of characters
       214,   126,   93,   (27)         [Exit if no characters
       101,   91,    0,    (71)         [If not beginning new half
       122,   91,    0,    4.0          [    word, send character
       215,   126,   91,   (80)         [    by character

24)    101,   91,    0,    (75)
       101,   94,    0,    (76)
       122,   94,    91,   0            [Length of store available
       122,   94,    93,   0            [Length  available or length
       216,   94,    94,   0            [    required: whichever
       124,   94,    93,   0            [    is smaller
       127,   94,    0,    *0000 7774   [Numbers of half words to be
                                        [  transferred
       122,   93,    94,   0            [Subtract from number required
       114,   94,    0,    (75)         [Address beyond last character
       121,   126,   0,    (39)         [Transfer: Return to (26)

26)    121,   94,    0,    (24)
       215,   126,   93,   (27)         [Jump if some characters remain
       217,   126,   92,   (27)         [If no characters remain exit
       121,   92,    0,    *4           [    or terminate record
       121,   126,   0,    (51)

27)    102,   91,    0,    (76)         [If block is full, insert
       216,   126,   91,   (50)         [    separator
       215,   126,   93,   (80)         [Transfer remaining characters
       521,   0,     0,    0            [End of extracode
```

```
        [Enter with B91 B94 B119

39)     214,    126,    94,     (26)
30)     121,    95,     94,     -3.4          [B95 Contains B94 or 3.4
        216,    95,     95,     0             [    whichever is smaller
        124,    95,     0,      3.0
        165,    96,     95,     3.4           [Copy to B96
        122,    94,     96,     0.4
        121,    126,    0,      (32)0.1
32)     101,    97,     119,    0             [Transfer from programmers store
        113,    97,     96,     (78)          [   to subsidiary store
        124,    119,    0,      0.4
33)     202,    126,    96,     (32)0.1
31)     113,    126,    0,      3*6           [INHIBIT INTERRUPTIONS
        121,    125,    0,      (35)0.2
        111,    125,    0,      3*6           [Interrupt control

34)     121,    97,     0,      (31)0.1
      \\121,    126,    0,      (1/590)0.4    [Call current output block
        113,    126,    0,      (79)          [Unsued now

35)     113,    126,    0,      (79)          [Preserve B126 in case of non
                                              [equivalence

36)     101,    97,     95,     (78)          [Transfer from subsidiary store
        113,    97,     91,     0             [  to supervisor
        124,    91,     0,      0.4
37)     202,    125,    95,     (36)
        121,    126,    0,      (39)
38)     113,    0,      0,      3*6           [Resume extracode control
```

```
        (0) = *4000 0610
43)     501,    122,    0,      (70)            [EXTRACODE 1061 Enter at (43)

        (0) = *4000 0620
44)     501,    122,    0,      (74)            [EXTRACODE 1062 Enter at (44)

        (0) = 1(38)

                                                [EXTRACODE 1065 Enter at (45)
45)     165,    94,     119,    7.7             [6-bit character
42)     101,    95,     0,      (71)
        121,    96,     0,      0.4
46)     217,    126,    95,     (47)0.1         [Jump if half word will be full
        125,    95,     94,     0
        121,    94,     0,      0               [Insert character shift up
        122,    96,     0,      0.1             [   and find end address
        121,    126,    0,      (46)

47)     125,    95,     94,     -4.0            [Add in and remove old sign bit
        101,    94,     0,      (75)
        113,    126,    0,      3*6             [INHIBIT INTERRUPTIONS
        121,    125,    0,      (48)0.2
        111,    125,    0,      3*6
        121,    97,     0,      3.1(47)
        121,    126,    0,      (1/590)0.4      [Call current output block

48)     113,    126,    0,      (79)            [Preserve B126 in case of
        113,    95,     94,     0               [   non-equivalence
        121,    126,    0,      2(0)
        113,    0,      0,      3*6             [Resume extracode control

        121,    94,     0,      (10)
        121,    126,    0,      (51)            [Insert separator
```

```
                                                          [R596.5

                                                          [Insert Separator. Enter with B94
50)     I2I,    95,     0,      0.I                       [Entry when store is full
        I2I,    I26,    0,      (52)0.I                   [   (record continues)

5I)     I2I,    95,     0,      I,0                       [Entry at end of record
        II4,    95,     0,      (73)                      [Count of records
        I2I,    95,     0,      0
        I2I,    I26,    0,      (52)0.I

56)     I27,    95,     0,      0.I                       [Re-entry after obtaining
                                                          [  new block
57)     I2I,    96,     0,      *7777 7774
        II7,    96,     0,      (76)                      [Ensure bits 0, I are zero
        I07,    96,     0,      (75)
        II3,    96,     0,      (75)
        II3,    96,     0,      (72)
52)     I0I,    96,     0,      (75)
        I24,    96,     0,      0.3
        I64,    95,     96,     *7777 7774                [Address for next separator

53)     I2I,    97,     0,      0
        2I0,    97,     95,     *I
        I0I,    96,     0,      (70)
        2II,    I26,    96,     (54)0.I
        I24,    97,     0,      *4
54)     I0I,    96,     0,      (72)
        2II,    I26,    96,     (55)0.I
        I24,    97,     0,      0.I*2
55)     I22,    97,     96,     0.4
        I04,    97,     0,      (75)                      [Separator

        II3,    I26,    0,      3*6                       [INHIBIT INTERRUPTIONS
        I2I,    I25,    0,      (58)0.2
        III,    I25,    0,      3*6                       [Interrupt control
        I2I,    97,     0,      (53)0.I
        I2I,    I26,    0,      (I/590)0.4                [Call output block

58)     II3,    I26,    0,      (79)                      [Preserve BI26 in case of
        II3,    97,     96,     0                         [non equivalence
        II3,    0,      95,     0                         [Insert separator and clear
        I2I,    I26,    0,      (59)                      [   space for next one
        II3,    0,      0,      3*6                       [Resume extracode control

59)     II3,    95,     0,      (72)                      [Address for next separator
        I65,    96,     95,     *7777 7774
        I24,    96,     0,      0.4
        II3,    96,     0,      (75)                      [Address for next character
        I2I,    97,     0,      (56)0.I
        I00,    96,     0,      (76)                      [End address - next character
                                                          [                address
        2I7,    I26,    96,     (I/594)                   [Next block if required
        I2I,    95,     0,      4.0
        II3,    95,     0,      (7I)                      [Set (7I)

        I2I,    I26,    94,     0                         [Return
```

2.9.63

```
                                              [R596.6

                                              [EXTRACODE 1060 Enter at (60)

60)   121,   91,   119,   0
61)   165,   94,   91,    *7777 7770          [Currently selected input
      101,   95,   0,     (70)
      165,   96,   95,    *7777 7770
      122,   96,   94,    0
      121,   97,   0,     (61)
      215,   126,  96,    (1/593)             [Select new output stream if
      101,   96,   0,     (75)                [   required
      102,   96,   0,     (72)                [Jump unless this is first
      215,   126,  96,    (63)                [   time output stream selected
      121,   94,   0,     (66)
      121,   95,   0,     0
      121,   126,  0,     (57)0.1             [Prepare first block

63)   124,   95,   91,    0                   [Test whether changing Bin/I.C.
      211,   126,  95,    (66)
      122,   96,   0,     0.4
      215,   126,  96,    (64)
      101,   96,   0,     (71)                [Exit if no characters have been
      122,   96,   0,     4.0                 [   sent to previous record
      214,   126,  96,    (66)

64)   113,   127,  0,     (78)                [Preserve main control
      121,   94,   0,     0
      121,   127,  0,     (65)                [Enter 1065 to end
      121,   126,  0,     (42)                [   previous record
65)   101,   127,  0,     (78)                [Recover main control

66)   113,   91,   0,     (70)
      121,   126,  0,     (4)
```

```
                                        [R596.7

80)     113,     127,     0,      (78)             [Preserve main control
        127,     119,     0,      *7777 7774
        121,     127,     0,      (82)             [Return address after extracode

82)     165,     97,      119,    0.3
        215,     126,     97,     (84)
        101,     91,      119,    0                [Pick up new half word
84)     125,     91,      0,      0
        124,     119,     0,      0.1
        122,     93,      0,      0.1
        165,     94,      92,     7.7              [Pick out character
        215,     126,     92,     (2)              [Enter extracode 1064

        101,     127,     0,      (78)             [Recover main control
        217,     126,     92,     (2)
        121,     126,     0,      (42)             [Enter extracode 1064 or 1065
                                                   [for last character
```

2.9.63

```
        T
        FIXED STORE COLUMN 4003

        (0) = 223*4003


        R659                                               |Activate scheduler



        (2) = (1/660)                                      |Main store scheduler
1)      101,    108,    0,      (3/660)
        167,    109,    108,    0
        113,    109,    0,      (3/660)                    |Force marker
        215,    126,    108,    7(1/312)                   |Exit if active
        121,    109,    0,      (2)
        121,    126,    0,      7(3/230)                   |Insert scheduler to queue
```

| 14/6/63.

```
                                                            |R630
        R630

        (30) = (10/205)
        (31) = 249.4*7
        (32) = (2/203)
        (33) = (1/649)
        (43) = (4/203)
        (44) = (3/203)

1)      103,    109,    0,      (30)
        217,    126,    109,    4(0)
        211,    126,    101,    (33)
        172,    109,    0,      6                           |MANCHESTER has 152, 109, 0, (31)
        226,    126,    0,      (33)
        124,    109,    0,      1.0
        111,    109,    0,      (30)
        121,    105,    0,      0.1
        101,    109,    105,    (32)
        211,    126,    109,    (9)
        200,    126,    105,    -2(0)
2)      165,    105,    108,    *3777
        101,    106,    0,      (43)
        163,    106,    105,    0
        216,    126,    106,    7(0)
        121,    105,    0,      0.4
        101,    106,    105,    (32)
        126,    106,    108,    0
        127,    106,    0,      *3777
        214,    126,    106,    5(0)
        200,    126,    105,    -4(0)
        125,    105,    0,      0
        125,    105,    0,      0
        121,    126,    0,      (2/399)                     |Blister to shift up 2 more
        101,    106,    105,    (44)
        121,    126,    110,    0
9)      121,    109,    105,    0
        125,    109,    0,      0
        125,    109,    0,      0
        163,    109,    0,      0
        163,    109,    0,      *3777 2001
        113,    109,    105,    (32)
        121,    126,    110,    0

                                                            [1/9/64
```

```
T
FIXED STORE COLUMN 40034

(0) = 210*40034



R704                        |Instruction counter monitor


(11) = 1000                                         |4 sec. extra time
(12) = *6                                           |Lowest block timer
(13) = *01                                          |Local time
(14) = *1                                           |Total time

(2)  = 9(3/303)                                     |Return to update timers
(3)  = (2/700)                                      |Montior
(4)  = (21/303)                                     |Check time
(5)  = (20/303)                                     |Timers
(6)  = (9/205)                                      |Program in store
(9)  = (4/203)                                      |Store directory
(10) = (3/203)                                      |Block timers

1)     101,    100,    0,      (4)                  |<- ENTRY
       211,    126,    100,    7(0)                 |JUMP IF NOT LOCAL CHECK
       121,    100,    0,      (13)                 |SET MARKER LOCAL TIME EXCEEDED
       101,    101,    0,      (7)
       113,    0,      0,      (7)                  |OVERFLOW CHECK TO ZERO
       113,    101,    0,      (4)                  |RESET CHECK TIMER
       121,    102,    0,      (2)
       121,    126,    0,      0.1(3)               |)EXIT TO R700 AND THENCE TO R303
       101,    101,    0,      (8)
       165,    102,    101,    *777                 |)JUMP IF COUNTER NOT EXHAUSTED
       215,    126,    102,    6(0)                 |)
       124,    101,    0,      0.1
       113,    101,    0,      (8)                  |)ACCUMMULATE ADDITIVE COUNT
       121,    101,    100,    (11)                 |MODIFIED CHECK TIMER
       121,    100,    0,      (14)                 |MARKER OVERALL TIME EXCEEDED
       121,    126,    0,      4(1)                 |TO MONITOR
       124,    101,    0,      1*777
       113,    101,    0,      (8)                  |)STEP TO NEXT SECTION
       101,    101,    0,      (6)
       101,    102,    101,    (9)                  |)SET UP COUNTERS TO SCAN BLOCK TIMERS
       165,    103,    102,    1023.4               |)
       127,    102,    0,      *7776                |)
       101,    105,    103,    (10)
       122,    105,    100,    0                    |SUBTRACT CHECK TIMER
       170,    105,    0,      (12)
       227,    126,    0,      3(0)                 |)IF TOO SMALL REPLACE BY STANDARD
       127,    105,    0,      0.7                  |)
       124,    105,    0,      (12)                 |)
       113,    105,    103,    (10)
       124,    103,    0,      0.4                  |)STEP THROUGH TIMERS
       122,    102,    0,      1024                 |)
       215,    126,    102,    -9(0)                |)
       110,    100,    101,    (5)                  |FORM NEW NUMBER OBEYED
       101,    101,    0,      (7)
       214,    126,    101,    4(0)                 |)FORM NEW CHECK AND OVERFLOW CHECK
       124,    101,    0,      0.1                  |)
       113,    101,    0,      (4)                  |)
       113,    100,    0,      (7)                  |)
       121,    126,    0,      (2)                  |RETURN TO R303

                                                          |18/3/63
```

```
T
FIXED STORE COLUMN 40014

(0) = 159*40014




R703                                         |Block monitor


(3)  = 2,0                                   |Monitor mark-store exceed
(6)  = *03                                   |Compile & supervisor bits
(4)  = (1/708)                               |exit for extra block
(5)  = (15/204)                              |Switch directory
(7)  = (4/204)                               |≠ marker
(9)  = (27/205)                              |Free program
(10) = (14/700)                              |Monitor
(11) = (9/205)                               |Current program in store
(12) = (7/202)                               |Branch in dicator
(13) = (1/232)                               |Branch block monitor
(14) = (1/202)                               |Program scan
(15) = (1/215)                               |Set full recover switch
(16) = (3/204)                               |Main programs
(17) = (7/204)                               |Short dumps
(18) = (9/204)                               |Status directory



1)      121,    100,    0,      (3)          |ENTRY FOR STORE EXCEED
        101,    101,    108,    (5)
        164,    100,    101,    (6)          |)COLLECT COMPILE SUPERVISE BITS
2)      101,    101,    0,      (7)          |ENTRY FOR LABEL EXCEED
        165,    106,    108,    63.4         |)EXIT IF NON EQUIVALENCE
        210,    126,    101,    (20)         |)
        152,    106,    0,      (11)
        225,    126,    0,      (19)         |)EXIT IF PROGRAM NOT IN STORE CONTROL
        101,    102,    0,      (12)
        216,    126,    102,    (13)         |)EXIT IF PROGRAM BRANCHING
        210,    126,    101,    2(0)
        122,    127,    0,      1            |)REDUCE M BY 1 IF NOT =/
        165,    101,    100,    (6)
        215,    126,    101,    (4)          |)EXIT IF EXTRA BLOCK PERMITTED
        121,    109,    0,      0.1(10)
        121,    126,    0,      5(15)        |)RESET FULL RECOVER SWITCH AND EXIT TO

                                             | R700 AND THEN PROGRAM SCAN
19)     113,    100,    106,    (17)
        113,    101,    106,    (16)(17)     |)SET UP PARAMETERS FOR RESUMPTION
        121,    100,    0,      5(2)         |)
        113,    100,    106,    (16)(16)(16)(16)(16)(17)
        121,    108,    0,      64.2         |)
        147,    108,    106,    (18)         |)RESUME IN STORE CONTROL AND IN SUPERVISER
        113,    108,    106,    (18)
        121,    126,    0,      (14)         |)
20)     113,    0,      0,      (7)          |RESET =/ SWITCH
        121,    108,    109,    0
        167,    108,    0,      *4           |B108 = BLOCK LABEL p23=1
        121,    110,    0,      3(2)         |FREE PRODRAM
        121,    126,    0,      (9)

                                             |30/8/63.
```

Section 9 Extracodes I200 - I777

This section contains a print-out of the Atlas Extracode programs from I200 upwards in Intermediate Input. These are, apart from residual errors and amendments which may from time to time prove desirable, in the form in which they will be loaded into the London and Harwell Fixed Stores. They are not an exact print-out of what is loaded in the Fixed Store of MUSE. Errors were found and improvement devised in a number of the extracodes in MUSE after they had been loaded, and the necessary changes were made making as few changes to the ''hairbrushes'' as possible. In some cases this involved inserting additional jump instructions. For the London and Harwell machines the instructions have to some extent been re-ordered to reduce the number of jumps and generally to tidy up the routines. At some date it may be possible to change the MUSE Fixed Store to render at least the arithmetic extracode part identical in all machines. However, the basic arithmetic is the same in all the computers.

Sub-section 9.I contains a description of the linking sytem for the functional extracode subroutines. This has been written in ABL. A description giving further information about the methods of the extracodes, particularly the functional ones, will be issued later in the volume containing the routine specifications.

9.I The Interlinking of the Functional Extracodes

The routines for implementing the following extracodes are all interconnected:-

I400	ca' = log s:
I402	ca' = exp s:
I4I0	ca' = sq.rt. s:
I4II	am' = arg s:
I4I2	am' = mod. s:
I4I3	ca' = s cos s*, s sin s*
I700	am' = log s
I70I	am' = log aq
I702	am' = exp s
I703	am' = exp aq
I7I0	am' = sq.rt. s
I7II	am' = sq.rt. aq
I7I2	am' = sq.rt. $(aq^{2}+s^{2})$
I7I3	am' = am^{n}
I720	am' = arcsin s
I72I	am' = arcsin aq
I722	am' = arccos s
I723	am' = arccos aq
I724	am' = arctan s
I725	am' = arctan aq
I726	am' = arctan (aq/s)
I730	am' = sin s
I73I	am' = sin aq
I732	am' = cos s
I733	am' = cos aq

These extracodes use five basic subroutines, namely:-

I. Square Root

2. Arctan, arccot

3. Log

4. Exp

5. sin, cos

In the cases of 2 and 5 the required function is indicated by means of markers in various B-lines.

All these subroutines are closed, i.e. exit is by means of a link-setting. Links are carried in B97.

Thus a simple exit is I2I I26 97 0

For a simple extracode which only requires the use of a single subroutine, for example, I720 to I725, this exit will be to a 'dummy exit' instruction 52I 0 0 0. In other cases, however, where operations are required afterwards, the exit will be to other routines. For example, I7I3 (am^{s}) requires first that log am be formed, then the result multiplied by s, and finally the exponential of this product formed. In such a case, to save instructions to reset links, a system is used whereby a single setting of B97 will normally cause the correct exit through all relevant routines.

The following is an outline of the complete system (in ABL notation). The entry points for all the extracodes are indicated, and all the link-setting and link-implementing instructions are shown. Also, an indication is given of the formulae used for the extracodes which call for more than one subroutine.

These routines are not listed in the order in which they occur in the store; they are listed in an order which seems logical in order to expound the system of interlinking. Since the labelling system used is sequential throughout the extracodes it is easy to discover the absolute position of each group of instructions.

In the annotations, the following notation is sometimes used for convenience, in addition to the standard Atlas notation:

x and y for s and s* (i.e. the real and imaginary parts of s:)

u and v for c(ba) and c(ba+I) (i.e. the real and imaginary parts of the complex accumulator Ca.)

9.I continued

```
                                JUMP TABLES
I2I   I26   0    A670      I400   ca' = log s:   u = log√(x^2+y^2),  v = arctan (y/x)
I2I   I26   0    A469      I402   ca' = exp s:   u = exp x cos y,   v = exp x sin y
I2I   I26   0    A625      I4I0   ca' = sq.rt.s: u =√(½√(x^2+ y^2)+x)),  v = y/2u
I2I   I26   0    0.IA662   I4II   am' = arg s:   am' = arctan (y/x)
I2I   I26   0    0.IA626   I4I2   am' = mod s:   am' =√(x^2+y^2)
I2I   I26   0    0.IA469   I4I3   ca' = s cos s*, s sin s*

324   0     II9  0         I700   am' = log s    Set aq' = s
I2I   I26   0    A587      I70I   am' = log aq.
324   0     II9  0         I702   am' = exp s    Set aq' = s
I2I   I26   0    A364      I703   am' = exp aq
324   0     II9  0         I7I0   am' =√s        Set aq' = s
I2I   I26   0    0.IA629   I7II   am' =√aq
I2I   I26   0    0.IA627   I7I2   am' =√(aq^2+s^2)
I2I   I26   0    A56I      I7I3   am' = am^s         am' = exp (s log am)
324   0     II9  0         I720   am' = arcsin s     Set aq' = s
I2I   I26   0    A678      I72I   am' = arcsin aq    am' = arctan (s/√(I-x^2))
324   0     II9  0         I722   am' = arccos s     Set aq' = s
I2I   I26   0    0.7A678   I723   am' = arccos aq    am' = arccot (s/√(I-x^2))
324   0     II9  0         I724   am' = arctan s     Set aq' = s
I2I   I26   0    A650      I725   am' = arctan aq
I2I   I26   0    0.IA663   I726   am' = arctan (aq/s)
324   0     II9  0         I730   am' = sin s    Set aq' = s
I2I   I26   0    0.IA544   I73I   am' = sin aq
324   0     II9  0         I732   am' = cos s    Set aq' = s
I2I   I26   0    A544      I733   am' = cos aq


                                    ROUTINES
625)  I2I   97   0   -3A640   (I4I0)   Set link to exit to A640
      - - - - - - - - - - - -
626)  - - - - - - - - - - - - (I4I2)   Set s* in A to form (s^2+s*^2) in A
      - - - - - - - - - - - -
627)  - - - - - - - - - - - - (I7I2)   Form (a^2+s^2) in A
      - - - - - - - - - - - -
629)  2I0   97   I26 -3A669   (I7I0/I) Set link to exit to A669 if not already set
630)  - - - - - - - - - - - -
63I)  - - - - - - - - - - - -          SQUARE ROOT
      - - - - - - - - - - - -          Form √a in A
      I2I   I26  97    3               Exit to b97+3
```

9.I continued

```
640)  - - - - - - - - - - - - -    (I4I0 continued)
      - - - - - - - - - - - - -    Add x, multiply by I/2
      - - - - - - - - - - - - -
      I2I   97    0    -3A643      Set link
      I2I   I26   0    A63I        Jump to form $\sqrt{(\frac{1}{2}(\sqrt{(x^2+y^2)}+x))}$, exit to A643
643)  - - - - - - - - - - - - -
      - - - - - - - - - - - - -    Store as u, form y/2u, store as v
      756   I22   95   0           EXIT from I4I0

650)  - - - - - - - - - - - - -    (I724/5)
      I2I   96    0    0           Set marker for arcTAN
      I2I   97    0    A669        Set link to exit to A669
652)  - - - - - - - - - - - - -
      - - - - - - - - - - - - -    ARCTAN/COT
      - - - - - - - - - - - - -    Form arctan/cot a in A
      I2I   I26   97   0           EXIT

662)  334   0     II9  I           (I4II)  Set s* in a to form s*/s
663)  2I0   97    I26  A669        (I726)  Set link to exit to A669 if not already set
      - - - - - - - - - - - - -
      - - - - - - - - - - - - -    Form am/s
      I2I   I26   0    A652        Jump to form arctan

669)  52I   0     0    0           EXIT
670)  I2I   97    0    A68I        (I400)  Set link
      I2I   I26   0    A626        Jump to form $\sqrt{(x^2+y^2)}$, exit to 3A68I = A683
672)  235   I26   0    A676        (I720, I, 2, 3 continued)  Jump if $\sqrt{(I-x^2)} \neq 0$
      - - - - - - - - - - - - -    If = I, jump direct to arctan
      I2I   I26   0    A652        exit to A669

676)  - - - - - - - - - - - - -
      - - - - - - - - - - - - -    (I720, I, 2, 3 continued) if $\sqrt{(I-x^2)} \neq 0$,
      - - - - - - - - - - - - -    form $x/\sqrt{(I-x^2)}$
      I2I   I26   0    A652        Then jump to form arctan.  Exit to A669
```

9.1 continued

```
678)  - - - - - - - - - - - - -           (1720, 1, 2, 3)
      - - - - - - - - - - - - -           Form 1-x^2
      121   97   126   A669-(*+1)         Set b97 = A669, preserving marker from b126
      165   96   97    0.2                Set marker for arcsin/arccos
      236   126  0     A630               Jump to form √(1-x^2) if 1-x^2 ≥ 0. Exit to
                                                                 3A669=A672

      Error exit if 1-x^2 < 0 i.e. x^2 > 1


681)  356   122  0     1                  (1300 continued) Store arctan (y/x) as v
      334   0    0     4J7                Recover √(x^2 +y^2)
      121   126  0     A589               Jump to form log √(x^2 +y^2).  Exit to A414
683)  356   0    0     4J7                (1400 continued) Store √(x^2 +y^2)
      121   126  0     A662               Jump to form arctan y/x.  Exit to A681

414)  756   122  0     0                  (End of 1400) Store log √(x^2 +y^2) as u. EXIT

587)  121   97   0     A537-A414+A681     (1700/1)  Set link to exit to A537
588)  - - - - - - - - - - - - -           Standardize
589)  - - - - - - - - - - - - -           LOG
      - - - - - - - - - - - - -           Form log a in A
      121   126  97    A414-A681          EXIT

537)  521   0    0     0                  EXIT

561)  121   97   0     A563-A414+A681     (1713)  Set link
      235   126  0     A588               Jump to form log am if a ≠ 0, exit to A563
      - - - - - - - - - - - - -
      - - - - - - - - - - - - -           If a = 0, prepare to set am' = 0 or EO
      - - - - - - - - - - - - -           depending on whether s ≥ or <0
      121   126  0     A575               Jump to complete, exit to A537

563)  342   0    119   0                  (1713 continued)  Form s x log am
564)  121   97   0     A563-A414+A681     (1702, 3 join)  Set link to exit to A537
565)  - - - - - - - - - - - - -
      - - - - - - - - - - - - -           EXP
      - - - - - - - - - - - - -           Form exp a in A
575)  - - - - - - - - - - - - -
      - - - - - - - - - - - - -           (tail) set EO if out of range
      121   126  97    A537-A563+A414-A681  Exit
```

9.I continued

```
469)  324   0     119   0              (I402, I4I3)  Set x in Am
      I2I   97    0     A47I-A537+A563-A4I4+A68I    Set link
      2II   I26   I26   A565           Jump if I402 to form exp x.  Exit to A47I
47I)  356   0     0     2J7            (Here directly with x in Am if I4I3; exp x if I402)Sto

      324   0     II9   I              Set s* in Am
      I2I   I26   0     A545           Jump to form cos s*.  Exit to A584

544)  I2I   97    I26   -(*+I)-A584+A47I+A563-A4I4+A68I
                                       Set link to exit to A537.  Preserve marker in bI26
545)  - - - - - - - - - - - - -        SIN/COS
      - - - - - - - - - - - - -        Form sin a in A if b97 odd, cos a if even
      I2I   I26   97    A584-A47I+A537-A563+A4I4-A68I   Exit

584)  362   0     0     2J7            (I402, I3 continued)  Multiply by x (I4I3) or
                                                                    exp x (I402)
      356   I22   0     0              Store as u
      324   0     II9   I              Bring out y
      I2I   97    0     2.I*-A584+A47I-A537+A563-A4I4+A68I   Set link to exit to 2*
      I2I   I26   0     A545           Jump to form sin y, exit to I*
      362   0     0     2J7            Multiply by x (I4I3) or exp x (I402)
      756   I22   0     I              Store as v and exit
```

9.2 the I200 Extracodes

```
     (0)=5I2*4
     2II,   I26,   I24,   (I)       |I200 ba'=n if AO set, clear AO. Jump when Acc. free
     2II,   I26,   I24,   2(I)      |I20I ba'=n if AO not set, clear AO. Ditto
             +0     /     0
 96)         *4     /     0         |Floating-point zero
     I2I,   I26,   0      (4)       |I204 ba'=no. of identical chars. from m.s. end of g and s
             +0     /     0
     I65,   9I,    98,    *77       |I206 ba'=n if m.s. char. of g=0. Extract m.s. char.
     2I5,   I2I,   I,     0         |)Set ba'=n if = 0, otherwise 'set BO' Exit
     52I,   I22,   II9,   0,        |)
 I6) I02,   II9,   I27,   -0.4      |(I2I6) Subtract N From BII9, i.e. bII9'=bm
     2I7,   I26,   II9,   (97)      |)Jump to exit if bm ≤ 0
     2I4,   I26,   II9,   (97)      |)
 I7) 50I,   I22,   I27,   -0.4      |Otherwise set ba=N and exit
             +0     /     0
     I2I,   I26,   0,     (I6)      |I2I6 ba'=n if bm > 0
     I02,   II9,   I27,   -0.4      |I2I7 ba'=n if bm ≤ 0. bII9'=bm
     2I7,   I26,   II9,   (I7)      |)Jump to set ba=N and exit if bm≤ 0
     2I4,   I26,   II9,   (I7)      |)
     52I,   0,     0,     0         |Otherwise exit
     I0I,   9I,    0,     6*6       |I223 ba'=n if Bcarry = I. Extract V6
     2II,   I2I,   9I,    0         |)Set ba'=n if l.s. digit = I. Exit
     52I,   I22,   II9,   0
     I24,   I26,   0,     0.5       |I226 ba'=n if bt>0. Add 0.5 to BI26
     227,   I26,   I26,   2.3       |I227 ba'=n if bt ≤ 0. Jump 3 or 4 if bt < 0
     224,   I26,   I26,   I.3       |Jump 2 or 3 if bt=0
     2II,   I2I,   I26,   0         |Set bI2I=0 if I227
     52I,   I22,   II9,   0         |Set ba'=n if ≤ 0 (I227), >0 (I226). Exit
 97) 52I,   0,     0,     0         |(I2I6 and I226 if bt ≤ 0), Exit
     I2I,   I26,   0,     0.I(34)   |I234 c'=c+2 if am approx = s
     I2I,   I26,   0,     (34)      |I235 c'=c+2 if am not approx=s
     I24,   I26,   0,     0.5       |I236 ba'=n if am>0. Add 0.5 to BI26
     237,   I26,   I26,   2.3       |I237 ba'=n if am≤0. Jump 3 or 4 if am < 0
     234,   I26,   I26,   I.3       |Jump 2 or 3 if am=0
     2II,   I2I,   I26,   0         |Set bI2I=0 if I237
     52I,   I22,   II9,   0         |Set ba'=n if ≤ 0(I237) >0(I236). Exit
     52I,   0,     0,     0         |(I236 if am ≤ 0) Exit
 77)        *77    /      *0077     |Character  masks
            *000077  /    7.7
 54)         +0     /     7.7
     *00007777  /     *00777777
     I2I,   I26,   0,     (50)      |I250 ba'=char. s in bits 0.5
     I2I,   I26,   0,     (5I)      |I25I s'=char. in bits 0.5 of ba
     203,   I26,   II9,   (52)      |I252 Unpack n chars. Jump if n ≠ 0, reduce n by I
     203,   I26,   II9,   0.I(52)   |I253 Pack n chars. Ditto
     52I,   0,     0,     0         |Exit if n = 0
     356,   0,     0,     (99)      |I255 ba' = n if m ≠ 0 or I's. Store m
     357,   0,     0,     I(99)     |Store 1
     I2I,   I24,   0,     *0I4      |Set exponent = I2
     340,   0,     0,     (0)       |Standardize, i.e. shift up 39 or more if m = 0 or all I's
     2I7,   I2I,   I24,   0         |Set bI2I=0 if shifted 39 or more places
     334,   0,     0,     (99)      |Recover m
     344,   0,     0,     I(99)     |Recover 1
     52I,   I22,   II9,   0         |Set ba'=n if shifted 39 or more otherwise 'set BO'Exit
     I25,   98,    0,     0         |I265g'=(64)g+n,ba'= overflow. Shift ms½
     I25,   99,    0,     0         |Shift l.s.½
     I65,   9I,    987.7            |Extract formertopchar.fromB98(overflow)
     I27,   98,    0,     *777777   |Remove it from bottom of B98
     I64,   98,    99,    7.7       |Add former top char. of B99 into bottom of B98
     I27,   99,    0,     *777777   |Remove it from bottom of B99
     I24,   99,    II9,   0         |Add in n to B99
     I0I,   92,    0,     6*6       |)Add in I at bottom of B98 if adding in n
     I64,   98,    92,    0.I       |)                                        set Bcarry
     52I,   I22,   9I,    0         |Exit putting overflow in ba.
             +0     /     0         |Fixed point zero
```

30.8.63

```
 I)  I2I,  9I,   0,    I.0       |(I200)
     II6,  9I,   0,    6*6       |Reverse A0 setting for I200
     I0I,  9I,   0,    6*6       |(I20Ijoins) Extract V6
     I27,  9I,   0,    I.0       |Mask out A0 digit
     II6,  9I,   0,    6*6       |Clear A0
     2I5,  I2I,  9I,   0         |Set bI2I=0 if A0 digit = 0 (I200),=I(I20I)
     52I,  I22,  II9,  0         |Set ba or b0=n. Exit
 4)  I2I,  92,   0,    -7        |(I204) Set count
     I2I,  9I,   98,   0         |Copy m.s. ½ of g to B9I
     I06,  9I,   II9,  0         |Non-equivalent with m.s. ½ of s
     2I5,  I26,  9I,   4(0)      |Jump if different
     I2I,  9I,   99,   0         |If same, copy l.s. ½ of g to B9I
     I06,  9I,   II9,  0.4       |Non-equivalent with l.s. ½ of s
     I2I,  92,   0,    -3        |Set counter
     I65,  93,   9I,   *77       |Extract m.s. character
     2I5,  I26,  93,   4(0)      |Jump if non-zero
     I25,  9I,   0,    0         |Shift round
     20I,  I26,  92,   -3(0)     |Cycle back reducing counter
     I2I,  92,   0,    I         |Set b92 =I when all 8 chars. same
     52I,  I22,  92,   7         |Exit setting up ba
50)  I0I,  9I,   II9,  0         |(I250) Extract s
     2I0,  I26,  II9,  2(0)      |Jump if k=I or 3
     I25,  9I,   0,    0         |Shift round if k=0 or 2
     I63,  II9,  II9,  0         |bII9'=½s-s
     2II,  I26,  II9,  3(0)      |Jump if even, i.e. last 2 digits of s same
     I25,  9I,   0,    0         |)Shift twice more, i.e. 3 in all for k=2
     I25,  9I,   0,    0,        |)                  and 2 in all for k=I
     565,  I22,  9I,   7.7       |Extract bottom char. which is required one and exit
51)  II3,  I22,  0,    (99)      |(I25I) Store ba
     I0I,  9I,   0,    (99)      |Set into B9I
     I65,  92,   II9,  0.3       |Extract k
     I24,  92,   92,   0         |)Shift up twice and subtract I.4
     I24,  92,   92,   -I.4      |)i.e. b92=0,-0.4,-I.0,-I.4, for k=3,2,I,0
     2I4,  I26,  92,   4(0)      |Jump if = 0, i.e. k=3 (no shifting needed)
     I23,  93,   92,   0.4       |Set b93=-(b92+0.4)=0,0.4,I.0, for k=2,I,0
     I25,  9I,   0,    0         |Shift round ba     ) shift I,2,3
     202,  I26,  93,   -I(0)     |Cycle, counting in b93    ) for k=0,I,2
     I06,  9I,   II9,  0         |Non-equivalent with s
     I07,  9I,   92,   I.4(77)   |Mask out required character position
     5I6,  9I,   II9,  0         |Non-equivalent back into s, i.e. plant new char. Exit
52)  II3,  I22,  0,    (99)      |(I252,3) Store ba
     I24,  I2I,  0,    0.4       |Step on BI2I to point at Ba*
     I0I,  9I,   0,    (99)      |Set ba in B9I
     I65,  93,   9I,   0.3       |Extract char. position of start,=k
     II3,  I22,  0,    (99)      |Store ba*
     I24,  93,   93,   0         |)
     I24,  93,   93,   -I.4      |)b93'=-I.4,-I.0,-0.4,0 as k=0,I,2,3
     I0I,  92,   0,    (99)      |Set ba* in B92
     I0I,  94,   9I,   0         |Set c(ba) in B94
     I04,  I26,  93,   *40053I24|Modified jump to shift 0,I,2,3 as k=0,I,2,3
     I25,  94,   0,    0
     I25,  94,   0,    0
     I25,  94,   0,    0
     I67,  9I,   0,    0.I       |Force marker at bottom of B9I
     2II,  I26,  I26,  (95)      |Jump if I252
     I07,  94,   93,   I.4(54)   |(I253 continued) Remove characters to be replaced
     I05,  94,   92,   0         |Shift required char. posn. to bottom add c(ba*)
     I24,  92,   0,    0.4       |Step on ba*
     200,  I26,  93,   4(0)      |Jump if b93 ≠ 0 (not last char. of ½ word), add 0.4.
     II3,  94,   9I,   0         |If end of ½ word, store b94 back in C(ba)
     200,  93,   9I,   -I.4      |     step on ba and reset count,
     I2I,  94,   0,    0         |     and clear B94 for next chars.
     203,  I26,  II9,  -5(0)     |Cycle, counting characters              30.8.63
     I23,  95,   93,   I.4       |b95'=-(b93+I.4)
```

```
                                      |I2CO Extracodes, Page 3
      2I4,   I26,   95,   (97)      |(I253 continued) Jump to exit if last char. fills ½ word
      IOI,   96,    95,   2(54)     |Extract mask
      II7,   96,    9I,   0         |Clear required char. positions in c(ba)
      IO4,   I26,   95,   *40053I3  |Modified jump to shift 3,2 or I as final k=0,I or 2
      I25,   94,    0,    0
      I25,   94,    0,    0
      I25,   94,    0,    0
      5I6,   94,    9I,   0         |Plant into c(ba) and exit
      200,   I26,   93,   3(0)      |(I252) Jump if b93≠0 (not last char. of ½ word), subtr.I
      200,   93,    9I,   -I.4      |if end of ½ word, step on ba and reset counter
      IOI,   94,    9I,   0         |Extract new c(ba)
95)   I25,   94,    0,    0         |(Entry) Shift char.to foot of B94
      I65,   95,    94,   7.7       |Extract character
      II3,   95,    92,   0         |Store in c(ba*)
      I24,   92,    0,    0.4       |Step on ba*
      203,   I26,   II9,  -7(0)     |Cycle, counting characters
      52I,   0,     0,    0         |Exit
             +0     /     0
34)   356,   0,     0,    (99)      |(I234,5) Store am
      234,   I26,   I26,  5.0       |Jump 6 places if am=0
      32I,   0,     II9,  0         |Subtract s
      374,   0,     0,    (99)      |Divide by 'am'
      366,   0,     0,    (0)       |Take modulus
      32I,   I22,   0,    0         |Subtract C(ba)
      237,   I26,   I26,  0.I       |Add 0.I to control if <0
      2I0,   I26,   I26,  2(0)      |Jump if BI26 odd, i.e. approx = for I235, not for I234
      I24,   I27,   0,    I.0       |Add I to BI27 otherwise
      734,   0,     0,    (99)      |Recover am and exit
```

```
    (0)=768*4
5)  334,  0,    119,  0         |1300 ba'=int.pt.s, am'= frac.pt.s. Put s in A
    121,  126,  0,    (1)       |1301 ba'=int.pt.am, am'=frac.pt.am. Jump
    120,  119,  0,    0         |1302 ba'=ba.n.  Set b119'=-n
    121,  126,  0,    (3)       |1303 ba'=-ba.n
    215,  126,  119,  0.5(3)    |1304 ba'=int.pt(ba/n), b97'=rem. Jump if n≠0
    376,  0,    0,    (5)       |Cause DO interrupt if n=0
          0     /     0
          0     /     0
          0     /     0
          0     /     0
    120,  119,  0,    0         |1312 ba'=ba,n (24 bit integers) Set b119'=-n
    121,  126,  0,    0.1(3)    |1313 ba'=-ba.n (ditto)
    215,  126,  119,  0.4(3)    |1314 ba'=int.pt.(ba/n), b97'=rem. (24 bit int.) Jump n≠0
    376,  0,    0,    (5)       |Cause DO interrupt if n=0
16) 335,  0,    0,    (99)      |(1302,3,12,13 continued) Set am'=-n
    352,  0,    0,    1(99)     |Multiply by ba
18) 210,  126,  95,   2(0)      |(1304,14 rejoin) Jump if 1304,12,13
    365,  0,    0,    (0)       |Shift down if 1302,3,14
    357,  0,    0,    (99)      |Store
    216,  126,  119,  2(0)      |Jump if ba and n are same sign
    112,  0,    0,    0.4(99)   |Negate answer if opposite
    334,  0,    0,    2(99)     |)Restore A
    344,  0,    0,    3(99)     |)
    501,  122,  0,    0.4(99)   |Set result in ba and exit
95)       *06404 /    0         |Mantissa 1/16,exponent 26
1)  356,  0,    0,    (99)      |(1300,1) Store
    217,  124,  124,  0         |ay'=0 if ay<0
    300,  0,    0,    (95)      |Add number (m=1/16, e=26),ie. shift integer to bottom of 1,
    357,  0,    0,    1(99)     |Store al=int.pt        |then standardize, i.e. shift up
    311,  0,    0,    (95)      |Subtract number off    |one octal place, so octal fraction
    302,  0,    0,    (99)      |am'=frac.part.         |clear.
    501,  122,  0,    1.4(99)   |ba'=int.part. Exit
    120,  119,  0,    0         |1340 Shift ba down n,(arithmetic, unrounded). b119'=-n
    215,  126,  119,  (40)      |1341 Shift ba up n (arithmetic). Jump if n≠0
    120,  119,  0,    0         |1342 Shift ba down n (circular). b119'=-n
    215,  126,  119,  (42)      |1343 Shift ba up n (circular). Jump if n≠0
    120,  119,  0,    0         |1344 Shift ba down n (logical). b119'=-n
    215,  126,  119,  0.1(40)   |1345 Shift ba up n (logical). Jump if n≠0
    521,  0,    0,    0         |(1340-5) Exit if n=0
    113,  122,  0,    (99)      |1347 h'=h v ba. Store ba
    101,  91,   119,  0         |h
    147,  91,   0,    (99)      |v ba
    513,  91,   119,  0         |Store as h' and exit
    121,  126,  0,    (53)      |1353 ba' = posn of m.s. 1 bit of n.
44) 216,  126,  92,   (97)      |(Logical shift down) Jump if n>-24(i.e. -24<n<0)
94) 521,  122,  0,    0         |(Arith & log shift up, n≥ 24; log shift down, n<-24)Exit, ba=0
    101,  119,  119,  0         |1356 bt'=ba≢h. Set h in b119
    113,  122,  0,    (99)      |1357 bt'=ba≢n. Store ba
    106,  119,  0,    (99)      |h or n ≢ ba
    572,  119,  0,    0         |Set bt and exit
    121,  90,   127,  0         |1362 Fast S/R entry. Set b90'=c+1
    521,  127,  119,  0         |Set b127 and exit.
    113,  122,  0,    (99)      |1364 ba'=(ba & not n)v(bm & n), b119'=(ba≢bm)& n
    102,  119,  127,  -0.4      |Remove n from B119, i.e. b119'=bm
    106,  119,  0,    (99)      |≢ba
    107,  119,  127,  -0.4      |& n
    526,  122,  119,  0         |≢ ba i.e. ba'=((ba≢bm)& n)≢ba = required result. exit
    521,  0,    0,    0         |1371 b121=Ba, b119'=N+bm. Dummy B-type extracode.
72) 121,  92,   119,  24        |(1342,3 continued, n<0) Set b92=n+24
    216,  126,  92,   (98)      |Jump if -24 ≤n<0
    120,  119,  0,    -23.7     |If n<-24, set b119=|n| and mark odd,
    121,  126,  0,    (46)      |  and jump to reduce mod 24
    101,  119,  119,  0         |1376 bt'=ba & h. Set h in b119
    113,  122,  0,    (99)      |1377 bt'=ba & n Store ba
```

```
                                       |I300 Extracodes, Page 2
    I07,  II9,  0,    (99)             |(I376,7 continued) h or n & ba
    572,  II9,  0,    0                |Set bt and exit
3)  I2I,  95,   I26,  -(0)-I.4         |(I302,3,4,I2,I3,I4) Set mark in B95
    356,  0,    0,    2(99)            |)Preserve A
    357,  0,    0,    3(99)            |)
    II3,  II9,  0,    0.4(99)          |Store n        )
    2I6,  I26,  II9,  2(0)             |Jump if n > 0 )|n| in store
    III,  II9,  0,    0.4(99)          |Store -n       )
    I27,  II9,  0,    *4               |BII9'=*4 if n<0, =0 if n ≥0
    II3,  0,    0,    (99)             |Set exponent and top of mantissa for |n|
    II3,  I22,  0,    I.4(99)          |Store ba
    I03,  97,   0,    I.4(99)          |b97'=-ba
    2I7,  I26,  97,   3(0)             |Jump if ba ≥0
    II3,  97,   0,    I.4(99)          |Store -ba, i.e. |ba| in store
    I26,  II9,  0,    0.I*4            |bII9' odd if ba<0, -ve if ba and n are of different
    II3,  0,    0,    I(99)            |Set exponent and top of mantissa for |ba|
    2I7,  I26,  95,   (I6)             |Jump if I302,3,I2,I3
    345,  0,    0,    I(99)            |(Here if division (I304,I4)) Set |ba| in L, clear M
    375,  0,    0,    (99)             |Divide by |n|. Result in L, remainder in M
    356,  0,    0,    (99)             |Store remainder
    364,  0,    0,    (0)              |Shift up quotient
    I0I,  97,   0,    0.4(99)          |Set b97=remainder
    2II,  I26,  II9,  (I8)             |Jump to adjust answer if ba>0
    I20,  97,   0,    0                |Set remainder -ve if ba <0
    I2I,  I26,  0,    (I8)             |Jump to adjust answer
    I63,  92,   0,    0                |(Reduction loop for I342,3) If M≠0, set b92=I6M
    I63,  92,   0,    0                |b92'=8M
    I27,  II9,  0,    3I.I             |Set bII9=m
    I24,  II9,  92,   0                |bII9'=8M+m, i.e. have removed 24M
46) I65,  92,   II9,  -32              |(Enter here) Regard bII9 as 32M+m. Extract 32M
    2I5,  I26,  92,   -5(0)            |Jump back if M≠0
    I22,  II9,  0,    24               |If M now = 0, subtract 24 from m.
    2I6,  I26,  II9,  -I(0)            |Subtract a further 24 if still +ve
    2II,  I26,  II9,  2(0)             |)
    I20,  II9,  0,    -23.7            |)Set bII9 = (-24+|n| reduced) if n<0
    I2I,  I26,  II9,  (45)             |Jump to shift
97) I63,  92,   0,    0                |(Log.shift down,n>-24) Halve b92
    I07,  9I,   92,   0.2*40052I5      |Preserve ba where zeros needed
    I00,  9I,   0,    (99)             |Set zeros at foot of ba
    I2I,  I26,  II9,  (45)             |Jump to shift
4I) I2I,  92,   II9,  -24              |(Arith and logical shift up) Set b92=ba-24
    2I6,  I26,  92,   (94)             |Jump if n≥0
    I63,  II9,  0,    0                |(ba<0) Halve n as mod for engineers test constants
    I07,  9I,   II9,  *40052I5         |Set zeros at top of ba
    I2I,  I26,  92,   (45)             |Jump to shift
40) II3,  I22,  0,    (99)             |(I340,I,3,4)) Set b9I=ba
    I0I,  9I,   0,    (99)             |          )
    2I6,  I26,  II9,  (4I)             |Jump if n>0 to shift up
    I2I,  92,   II9,  23.4             |Set b92 =n+23.4
    2I0,  I26,  I26,  (44)             |Jump if logical shift down
    2I6,  I26,  9I,   (44)             |(Arithmetic shift down) Jump for logical shift if ba
    2I7,  I26,  92,   4(0)             |(Arithmetic shift down, ba<0) Jump if n≤-24
    I63,  92,   0,    0                |Halve b92 as mod for eng. tests consts.
    I47,  9I,   92,   0.2*40052I5      |''or'' ones to foot of ba
    I2I,  I26,  II9,  (45)             |Jump to shift
    52I,  I22,  0,    -0.I             |(Arithmetic shift down ba<0, n<-24) Set ba =-0.I and
42) II3,  I22,  0,    (99)             |(I342,3)) Set b9I=ba
    I0I,  9I,   0,    (99)             |        )
    2I7,  I26,  II9,  (72)             |Jump if n<0
    I22,  II9,  0,    24               |(n>0) Subtract 24
    2I6,  I26,  II9,  (46)             |Jump to reduce mod 24 if not 0<n<24
98) I2I,  I26,  II9,  (45)             |Jump to shift if 0<n<24 or -24≤n<0
53) I2I,  I22,  0,    0.I*4            |(I353) Set ba digit 23=I. Interrupt if Ba=I26
    II3,  I22,  0,    3*6              |Inhibit interrupts
                                       |
                                        |I300 Extracodes, Page 3
      I2I,  I23,  II9,  0               |(I353 continued) n -> BI23
      I2I,  I22,  I23,  0               |BI23 -> ba
      5I3,  0,    0,    3*6             |De-inhibit interrupts and exit
43)   I25,  9I,   0,    0               |(Shift table)
      I25,  9I,   0,    0
      I25,  9I,   0,    0
      52I,  I22,  9I,   0               |Exit
      I63,  9I,   0,    0               |(Here for shift up one) Shift down one
      I63,  9I,   0,    0               |(Two) down one
      I63,  9I,   0,    0               |(Three) down one
      I63,  9I,   0,    0               |(Four) down one
      I63,  9I,   0,    0               |(Five) down one
      I2I,  I26,  0,    2(43)           |(Six) Jump to shift up six and exit
      I63,  9I,   0,    0               |(Seven) Shift down one
      I63,  9I,   0,    0               |(Eight) down one
      I63,  9I,   0,    0               |(Nine) down one
      I63,  9I,   0,    0               |(Ten) down one
      I63,  9I,   0,    0               |(Eleven) down one
      I2I,  I26,  0,    I(43)           |(Twelve) Jump to shift up I2 and exit
      I63,  9I,   0,    0               |(I3) Shift down one
      I63,  9I,   0,    0               |(I4) down one
      I63,  9I,   0,    0               |(I5) down one
      I63,  9I,   0,    0               |(I6) down one
      I63,  9I,   0,    0               |(I7) down one
      I2I,  I26,  0,    (43)            |(I8) Jump to shift up I8 and exit
      I63,  9I,   0,    0               |(I9, i.e. down 5) Shift down one
      I63,  9I,   0,    0               |(u.20, i.e.d.4) down one
      I63,  9I,   0,    0               |(u.2I, i.e.d.3) down one
      I63,  9I,   0,    0               |(u.22, i.e.d.2) down one
      I63,  9I,   0,    0               |(u.23, i.e.d.I) down one
45)   52I,  I22,  9I,   0               |(u.24, i.e.d.0) Set b9I in ba and exit
```

9.4 The I400 Extracodes

```
    (0)=I024*4
    I2I,   I26,   0,     (I/I600)      |I400 ca'=log s:
           0      /      0
    I2I,   I26,   0,     (2)           |I402 ca'=exp s:
    324,   0,     II9,   0             |I403 ca'=conj s:     )
    356,   I22,   0,     0             |                     )Transfer real part
25  325,   0,     II9,   I             |(I425 continued joins))
    756,   I22,   0,     I             |                       )Negate and transfer imag. part. Exit
           0      /      0
    I2I,   I26,   0,     (I0/I600)     |I4I0 ca' =sq.rt. s:
    I2I,   I26,   0,     0.I(II/I600)  |I4II am'=arg s:
    I2I,   I26,   0,     0.I(I2/I600)  |I4I2 am'=mod s:
    I2I,   I26,   0,     0.I(2)        |I4I3 ca'=s cos s:, s sin s*
    I2I,   I26,   0,     (I4)          |I4I4 ca'=I/s:
    3I4,   0,     II9,   0             |I4I5 Pseudo-Random Number.  Set s in A
    352,   0,     II9,   I             |Multiply by s*, double-length, non-standardized
    757,   0,     II9,   0.4           |Store l.s.½ in s or s*.  Exit
    I2I,   I26,   0,     (20)          |I420 ca'=ca+s:
    324,   I22,   0,     I             |I42I ca'=ca-s:
    32I,   0,     II9,   I             |Subtract imaginary parts
    I2I,   I26,   0,     (2I)          |Jump to continue
    I2I,   I26,   0,     (24)          |I424 ca'=s:
    325,   0,     II9,   0             |I425 ca'=-s: )
    356,   I22,   0,     0             |             )Negate and transfer real part
    I2I,   I26,   0,     (25)          |Jump to negate and transfer imaginary part and exit
    203,   I26,   II9,   (30)          |I430 s(I)'=s(I)+s(2)   )
    203,   I26,   II9,   0.I(30)       |I43I s(I)'-s(2)         )Jump reducing bII9 by I
    203,   I26,   II9,   0.4(30)       |I432 s(I)'=am.s(2)      )   if n≠0
    203,   I26,   II9,   0.5(30)       |I433 s(I)'=s(I)+am.s(2))
    203,   I26,   II9,   (34)          |I434 s(I)'=s(2)         )
    52I,   0,     0,     0             |Exit if n=0
    203,   I26,   II9,   0.4(34)       |I436 am'=sum s (Ii).s(2i) )Ditto
    203,   I26,   II9,   0.5(34)       |I437 a'=sum s(Ii).s(2i)   )
    52I,   0,     0,     0             |Exit if n=0
    II3,   I22,   II9,   0.4           |I44I sx'=ba, sy'=I2.  Store ba in l.s.½ of s
    I2I,   92,    0,     *03           |Set exponent =I2
    I0I,   9I,    II9,   0.4           |Set ba in B9I
    2I7,   92,    9I,    *03I77777     |Propagate sign digit into m.s.½
    5I3,   92,    II9,   0             |Store exponent and propagated sign digit in l.s.½ of s. Exi
24) 324,   0,     II9,   I             |(I424))
    356,   I22,   0,     I             |       )Transfer imaginary part
    324,   0,     II9,   0             |)Transfer real part and exit
95) 756,   I22,   0,     0             |) (Also end of I400; store am in c(ba) and exit)
    342,   0,     II9,   0             |I452 m'=m.sx times 8 to(ya+ys-ba), ya'=ba (X)
    I2I,   I26,   0,     (47/I700)     |Multiply m by s and jump to set exponent
           0      /      0
           0      /      0
    324,   I22,   0,     0             |I456 s:'=ca )
    356,   0,     II9,   0             |            )Transfer real part
    324,   I22,   0,     I             |)
    756,   0,     II9,   I             |)Transfer imaginary part and exit
    334,   I22,   0,     0             |I462 ca'=ca.s:     )
    362,   0,     II9,   0             |                   )Form product of real parts
    356,   0,     0,     (99)          |Store
    I2I,   I26,   0,     (62)          |Jump to continue
    I2I,   I26,   0,     (66)          |I466 a'=C(s+bm+ba).C(s+bm) + a
    II3,   I22,   0,     I(99)         |I467 am'=Polynomial sum. Store ba =no. of terms
    346,   0,     0,     (99)          |Store am. Set am =0
    I0I,   9I,    0,     I(99)         |Set count in B9I
    I2I,   I26,   0,     (67)          |Jump to polynomial loop
    I2I,   I26,   0,     (4I/I700)     |I473 m'=(xa/xs) times 8 to (ya-ys-ba), ya'=ba (X)
    347,   0,     0      (0)           |I474 C(ba')= quotient (am/s),am'= remainder (X). Clear 1
    I2I,   I26,   0,     (8I/I700)     |I475 C(ba')=quotient (a/s),am'=remainder (X).
    I2I,   I26,   0,     0.I(80/I700)  |I476 C(ba')= quotient) ([am]/s),am'=remainder (X)
    I2I,   I26,   0,     0.4(77/I700)  |I477 Remainder and Adjusted Integral Quotient after divisio
```

30.8.63

```
I4)     334,  0,    II9,  0           | (I4I4) s
        362,  0,    II9,  0           |s.s
        356,  0,    0,    (99)        |Store
        334,  0,    II9,  I           |s*
        362,  0,    II9,  I           |s*.s*
        320,  0,    0,    (99)        |Add s.s.
        356,  0,    0,    I(99)       |Store
        325,  0,    II9,  I           |-s*
        374,  0,    0,    I(99)       |Divide by (s.s + s*.s*)
        356,  I22,  0,    I           |Store as imaginary part of result
        324,  0,    II9,  0           |s
        374,  0,    0,    I(99)       |Divide by (s.s + s*.s*)
        756,  I22,  0,    0           |Store as real part of result. Exit
20)     324,  I22,  0,    I           |(I420)
        320,  0,    II9,  I           |Add real parts
        356,  I22,  0,    I           |Store
        324,  I22,  0,    0
        320,  0,    II9,  0           |Add imaginary parts
        756,  I22,  0,    0           |Store and exit
2I)     356,  I22,  0,    I           |(I42I continued) Store imaginary part of result
        324,  I22,  0,    0
        32I,  0,    II9,  0           |Subtract real parts
        756,  I22,  0,    0           |Store and exit
62)     334,  I22,  0,    I
        363,  0,    II9,  I           |Negative product of imaginary parts
        320,  0,    0,    (99)        |Add product of real parts
        356,  0,    0,    I(99)       |Store temporarily (=real part of result)
        334,  I22,  0,    0           |)
        362,  0,    II9,  I           |)
        356,  0,    0,    (99)        |) Form sum of 'cross-products'.
        334,  I22,  0,    I           |)
        362,  0,    II9,  0           |)
        320,  0,    0,    (99)        |)
        356,  I22,  0,    I           |Store as imaginary part of result
        334,  0,    0,    I(99)       |)
        756,  I22,  0,    0           |)Extract and store real part and exit
66)     356,  0,    0,    (99)        |(I466) Store am
        355,  0,    0,    (0)         |Shift up 1
        356,  0,    0,    2(99)       |Store
        324,  0,    II9,  0           |C(S+bm)
        342,  I22,  II9,  0           |times C(s+bm+ba)
        I2I,  II9,  0,    0           |Set bII9=0
        I2I,  I26,  0,    (68)        |Jump to add a
30)     I2I,  92,   I26,  -I.4-(0)    |(I430,I,2,3) Set mark in b92
        II3,  I22,  0,    (99)        |Store ba
        I24,  I2I,  0,    0.4         |Step on BI2I to point at Ba*
        I0I,  9I,   0,    (99)        |Set b9I=ba
        2I6,  I26,  92,   (32)        |Jump if I432,3
        2I0,  I26,  92,   (3I)        |Jump if I43I
        324,  9I,   II9,  0           |(I430 continued) sIi
        320,  I22,  II9,  0           |+s2i
        356,  9I,   II9,  0           |Store in sIi
        203,  I26,  II9,  -3(0)       |Cycle, counting
        52I,  0,    0,    0           |Exit
3I)     324,  9I,   II9,  0           |(I43I continued) sIi
        32I,  I22,  II9,  0           |-s2i
        356,  9I,   II9,  0           |Store in sIi
        203,  I26,  II9,  -3(0)       |Cycle, counting
        52I,  0,    0,    0           |Exit
32)     356,  0,    0,    (99)        |(I432,3 continued) Store am
        2II,  I26,  92,   3(0)        |Jump if I432 into loop.
        I2I,  I26,  0,    (33)        |(I433 continued) Jump into loop
        356,  9I,   II9,  I           |(I432 loop) Store in sIi
        324,  I22,  II9,  0           |(Enter here ) s2i
```

30.8.63

```
     (0)=*4002350

50)  I02,   II9,   I27,   -0.4        |(II02) Subtract n from bII9 i.e. bII9'=bm
5I)  I2I,   9I,    I27,   0           |(II00,II0I join) Set b9I'=c+I
     I2I,   I27,   II9,   0,          |Set c'= bII9 (=s,n,bm for II00,II0I,
                                      |            II02 respectively).
     52I,   I22,   9I,    0           |Set ba'=c+I (from b9I) and exit


     (0)=*4002354

54)  5I3,   II9,   0,     6*6         |(II24)Set n in V6 and exit
55)  I07,   II9,   0,     6*6         |(II25) bII9'=v6 & n
     52I,   I22,   II9,   0           |Copy to ba and exit
57)  II3,   I22,   0,     (99)        |(II3I) Store ba
     I0I,   9I,    I27,   0           |Set C(c+I) in B9I
     I65,   92,    9I,    2047        |Set 1 (count) in B92
     I25,   9I,    0,     0           |)Shift k to integer position in B9I
     I25,   9I,    0,     0           |)
     I0I,   93,    0,     (99)        |Set ba in B93
     20I,   I26,   I27,   2(0)        |Jump into loop and increase bI27 by I
     I64,   93,    9I,    5II.4       |Add k to b93
     I0I,   94,    93,    0           |Set C(b93) in B94(first time C(ba), then C(ba+k) etc)
     I07,   94,    I27,   -0.4        |Mask b94 with m (=C(c+I.4))
     I26,   94,    II9,   0           |Non-equivalent with n
     2I4,   I26,   94,    3(0)        |Jump if zero i.e. test successful
     203,   I26,   92,    -5(0)       |Cycle if non-zero, counting from 1 till zero
     I2I,   93,    0,     *4          |If still unsuccessful after 1 cycles, prepare to set ba=*4
     52I,   I22,   93,    0           |Exit with ba=address of successful ½word (*4 otherwise)
76)  I2I,   9I,    0,     3           |Error exit (x<0) for LOG. Set marker
     I2I,   I26,   0,     *4006003    |Jump to monitor
     362,  0,    0,    (99)          |(I432 continued) Multiply by ''am''
     203,  I26,  II9,  -3(0)         |Cycle, counting
     756,  9I,   II9,  0             |Store last element and exit
     356,  9I,   II9,  I             |(I432 loop) Store in sIi
33)  324,  I22,  II9,  0             |(Enter here) s2i
     362,  0,    0,    (99)          |Multiply by ''am''
     320,  9I,   II9,  0             |Add sIi
     203,  I26,  II9,  -4(0)         |Cycle, counting
     756,  9I,   II9,  0             |Store last element and exit
34)  I2I,  92,   I26,  -I.4-(0)      |(I434,6,7) Set mark in B92
     II3,  I22,  0,    (99)          |Store ba
     I24,  I2I,  0,    0.4           |Step on BI2I to point at Ba*
     I0I,  9I,   0,    (99)          |Set ba in B9I
     2I6,  I26,  92,   (36)          |Jump if I436,7.
     II3,  I22,  0,    I(99)         |(I434 continued) Store ba*
     I02,  9I,   0,    I(99)         |ba-ba* in B9I
     2I7,  I26,  9I,   6(0)          |Jump if ba*>ba, i.e. if transfer backwards
     I0I,  9I,   0,    (99)          |Set ba in B9I
     334,  I22,  II9,  0             |(loop) Extract element from s2 starting at highest ad
     356,  9I,   II9,  0             |Store in sI
     203,  I26,  II9,  -2(0)         |Cycle, reducing modifier
     52I,  0,    0,    0             |Exit
     I23,  92,   II9,  0             |(ba*>ba) Set b92=-(n-I)
     I04,  II9,  0,    (99)          |SET ba+(n-I) in BII9
     I20,  9I,   II9,  0             |Set B9I= ba+n-I-(ba-ba*),=ba*+n-I
     334,  9I,   92,   0             |(loop) Extract element from s2, starting at lowest ad
     356,  II9,  92,   0             |Store in sI
     20I,  I26,  92,   -2(0)         |Cycle increasing modifier
96)  52I,  0,    0,    0             |Exit
36)  346,  0,    0,    (0)           |(I436,7 contiued) Set zero in A
     2I0,  I26,  92,   (37)          |Jump if I437
     356,  0,    0,    (99)          |Store partial sum (zero initially)
     324,  I22,  II9,  0             |Extract s2i
     362,  9I,   II9,  0             |Multiply by sIi
     320,  0,    0,    (99)          |Add previous partial sum, single length, QR
     203,  I26,  II9,  -4(0)         |Cycle, reducing modifier
     52I,  0,    0,    0             |Exit with result in Am
37)  356,  0,    0,    (99)          |(I437 continued) Store m.s.½ of partial sum (zero init
     355,  0,    0,    (0)
     356,  0,    0,    2(99)         |Store l.s.½ of partial sum
     324,  I22,  II9,  0             |Extract s2i
    -342,  9I,   II9,  0             |Multiply by sIi, double length, Q.
68)  356,  0,    0,    I(99)         |(I466 continued joins) Store m
     355,  0,    0,    (0)           |Shift 1 into m
     320,  0,    0,    2(99)         |Add c(2(99)) to ''1''
     356,  0,    0,    3(99)         |Store
     324,  0,    0,    (99)          |c((99))
     300,  0,    0,    I(99)         |Add ''m''
     356,  0,    0,    4(99)         |Store
     355,  0,    0,    (0)           |Shift up 1
     300,  0,    0,    3(99)         |Add sum of l.s. halves
     3I0,  0,    0,    4(99)         |Add m.s.½ of sum of m.s. halves
     203,  I26,  II9,  (37)          |Cycle, reducing modifier (not I466)
     52I,  0,    0,    0             |Exit
67)  362,  0,    0,    (99)          |Multiply by ''am''
     320,  II9,  9I,   0             |Add coefficient
     203,  I26,  9I,   -2(0)         |Cycle, reducing modifier
     52I,  0,    0,    0             |Exit
2)   324,  0,    II9,  0             |(I402,I3) Set s in Am
     I2I,  97,   0,    *4003630      |Set link
     2II,  I26,  I26,  (36/I500)     |Jump if I402
75)  356,  0,    0,    2(99)         |Preserve s
     324,  0,    II9,  I             |Set s* in Am
     I2I,  I26,  0,    (86/I500)     |Jump to form cos
```

30.8.63

9.5 The 1500 Extracodes

```
     (0)=I280*4
     I2I,  I26,  0,    (8)           |I500 a'=a+s:
     I2I,  I26,  0,    (I)           |I50I a'=a-s:
     I2I,  I26,  0,    (2)           |I502 a'=-a+s:
64)  762,  0,    0,    (99)          |(I562) Multiply by n and exit
     I2I,  I26,  0,    (4)           |I504 a'=s:
5)   325,  0,    II9,  I             |I505 a'=-s:
     70I,  0,    II9,  0
7)   362,  0,    II9,  0             |(I542,3) ''1'' xs
     356,  0,    0,    I(99)         |Store
     324,  0,    0,    (99)          |''m''
     362,  0,    II9,  I             |''m''xs*
     320,  0,    0,    I(99)         |Add ''1''xs
     356,  0,    0,    I(99)         |Store
     324,  0,    II9,  0             |s
     342,  0,    0,    (99)          |a'=''m''xs
     I2I,  I26,  0,    (80)          |Jump to add other components and exit
     I2I,  I26,  0,    I.I(25)       |I520 am'=am+n.
     I2I,  I26,  0,    0.I(25)       |I52I am'=am-n.
22)  I2I,  I26,  0,    (53)          |(I520,I continued) jump to set am=am+c(99) & exit
     734,  0,    0,    (99)          |(I534,5 continued) set am'=c(99), 1'=0 (X). exit
     I2I,  I26,  0,    2(0)          |I524 am'=n,1'=0
25)  I20,  II9,  0,    0             |I525 am'=-n,1'=0 (Also I52I,35) Set bII9'=-n
     II3,  II9,  0,    0.4(99)       |(I520,4,34 join) Store ± n
     I2I,  9I,   0,    *03           |) Set up most significant half of (99) to
     2I7,  9I,   II9,  *03I77777     |)      give floating point number from ± n
     II3,  9I,   0,    (99)          |)    with exponent I2 and sign copied up
     2II,  I26,  I26,  (60)          |Jump if bI26 even (I524,5)
     I22,  I26,  0,    9.4           |Jump back to (22) if I520,I and to I(22) if I534,5
     I2I,  I26,  0,    I.5(25)       |I534 am'=n,1'=0(X)
     I2I,  I26,  0,    0.5(25)       |I535 am'=-n,1'0(X)
65)  356,  0,    0,    (99)          |(I565) store m
     355,  0,    0,    (0)           |Shift 1 to m
     322,  0,    0,    (98)          |m'=''1''
     7II,  0,    0,    (99)          |Subtract ''m''(i.e.a'=-a) and exit
     I2I,  I26,  0,    0.I(8)        |I542 a'=a.s:
     I2I,  I26,  0,    0.I(2)        |I543  a'=-a.s:
I)   356,  0,    0,    (99)          |(I50I) Store m
     355,  0,    0,    (0)           |Shift 1 to m
     32I,  0,    II9,  I             |''1''-s*QR
     356,  0,    0,    I(99)         |Store
     324,  0,    0,    (99)          |Bring out ''m''
     30I,  0,    II9,  0             |''m''-s in A
     I2I,  I26,  0,    (80)          |Jump to add (''1''-s*) and exit
53)  356,  0,    0,    I(99)         |(I520,I continued) Store am
     334,  0,    0,    (99)          |Set c(99) in A, standardized
     720,  0,    0,    I(99)         |Add 'am' and exit
     356,  0,    II9,  0             |I556 s:'=a.    Store m in s
     355,  0,    0,    (0)           |Shift 1 to m
     356,  0,    II9,  I             |Store ''1'' in s*
     730,  0,    II9,  0             |Add back '''m'' to restore a,  exit
     I2I,  I26,  0,    0.I(62)       |I562 am'=am.n
66)  237,  I26,  0,    (65)          |(I566) Jump if ax <0
     740,  0,    0,    (0)           |If ax >0, standardise and exit
     I2I,  I26,  0,    (65)          |I565 a'=-a
     I2I,  I26,  0,    (66)          |I566 a'=|a|
     334,  0,    II9,  0             |I567 a'=|s:|.  Set am from s
     237,  I26,  0,    (5)           |Jump if ax<0 to I505 (a'=-s:)
4)   324,  0,    II9,  I             |(also I504)   Set am from s*
     700,  0,    II9,  0             |Add s (i.e. a'=s:) and exit
98)        *4    /     0             |Floating - point zero
     347,  0,    0,    (0)           |I574 am'=am/n.   Clear 1
     I2I,  I26,  0,    (75)          |I575 am'=aq/n.   Jump for I574,5
     356,  0,    0,    (99)          |I576 a'=a/s:.    Store m
     355,  0,    0,    (0)           |Shift 1 to m
     356,  0,    0,    I(99)       |(I576 continued)   Store ''1''
     324,  0,    0,    (99)        |Bring back ''m''
     374,  0,    II9,  0           |Divide by s, QR,=(am/s)R
     356,  0,    0,    2(99)       |Store
     343,  0,    II9,  0           |Multiply by 5,=-(am/s)R x s
     356,  0,    0,    3(99)       |Store m.s.½
     355,  0,    0,    (0)         |)
     356,  0,    0,    4(99)       |)Store l.s.½
     324,  0,    0,    3(99)       |Bring back m.s.½
     300,  0,    0,    (99)        |Add ''m''
     320,  0,    0,    4(99)       |Add l.s.½ of -(am/s)R x s
     320,  0,    0,    I(99)       |Add ''l''
     356,  0,    0,    3(99)       |Store, =(a-(am/s)R x s)
     324,  0,    0,    2(99)       |Bring back (am/s)R
     363,  0,    II9,  I           |Multiply by -s*
     320,  0,    0,    3(99)       |Add c(3(99)),=(a-(am/s)R x s)-(am/s)R x s*
     374,  0,    II9,  0           |Divide by s
     700,  0,    0,    2(99)       |Finally add (am/s)R and exit
75)  340,  0,    0,    (0)         |(I574,5) Standardize.
     356,  0,    0,    I(99)       |Store m
     355,  0,    0,    (0)         |)
     356,  0,    0,    3(99)       |)Store l
62)  I2I,  9I,   0,    *03         |(I562 joins)
     II3,  II9,  0,    0.4(99)     |)Set up n floating point in (99)
     2I7,  9I,   II9,  *03I77777   |)
     II3,  9I,   0,    (99)        |)
     2I0,  I26,  I26,  (64)        |Jump if bI26 odd (I562)
     324,  0,    0,    (99)        |(I574,5) n standardized in A
     I2I,  I26,  0,    6(0)        |Jump
74)  340,  0,    0,    (0)         |(I774,5) Standardize a.
     356,  0,    0,    I(99)       |Store m
     355,  0,    0,    (0)         |)Store l
     356,  0,    0,    3(99)       |)
     324,  0,    II9,  0           |s standardised in A
     356,  0,    0,    (99)        |(I574,5 rejoin)  Store standardised divisor
     334,  0,    0,    I(99)       |Bring back m.s. ½ of dividend
     774,  0,    0,    (99)        |Divide and exit
50)  52I,  0,    0,    0           |Dummy exit
2)   356,  0,    0,    (99)        |(I502,I543)  Store m
     355,  0,    0,    (0)         |Shift l to m
     322,  0,    0,    (98)        |Set m =-''1''
     3II,  0,    0,    (99)        |Subtract ''m'' (i.e.a'=-a)
8)   356,  0,    0,    (99)        |(I500,I542 join) Store m
     355,  0,    0,    (0)         |Shift l to m
     2I0,  I26,  I26,  (7)         |Jump if bI26 odd (I542,3)
     320,  0,    II9,  I           |(I500,2) Add s* to ''l''
     356,  0,    0,    I(99)       |Store
     324,  0,    0,    (99)        |Bring back ''m''
     300,  0,    II9,  0           |Add s
80)  356,  0,    0,    (99)        |(I50I joins, I542,3 rejoin)  Store m
     355,  0,    0,    (0)         |Shift l to m
     300,  0,    0,    I(99)       |Add C(I(99))
     7I0,  0,    0,    (99)        |Add ''m'' and exit
3I)  I2I,  97,   I26,  *0000672    |(I730,I,2,3) Set link for exit to (96/I400)
86)  342,  0,    0,    (97)        |SIN/COS.      Multiply by I/2π
     2I7,  I26,  I24,  3(0)        |Jump if small (<I/8, i.e.x < π /4)
     330,  0,    0,    (96)        |Fix with exponent I3 unless very large
     355,  0,    0,    (0)         |Take fractional part i.e. reduce mod 2  (zero if lar
     2I0,  I26,  97,   2(0)        |Jump if sin
     32I,  0,    0,    (94)        |If cos, subtract -I/4 (i.e. add π /2 to x)
     362,  0,    0,    (95)        |x½
     2I7,  I26,  I24,  4(0)        |Jump if exponent - ve, i.e.<+I/8
     32I,  0,    0,    (95)        |Subtract ½ (range-3/8 to I/8 , i.e. -3π /2<x<π /2)
     2I7,  I26,  I24,  2(0)        |Jump if >-I/8 (x>-π /2)
```

30.8.63

```
     322,  0,   0,   (94)      |(Sin/Cos continued) If between -3/8 and -I/8; add I/4
     356,  0,   0,   (99)      |Store as y            |(range ± I/8, or ±π/2 in X)
     362,  0',  0,   (99)      |y squared
     I2I,  9I,  0,   4         |Set count
     346,  0,   0,   I(99)     |Store y squared, clear A
     3I0,  0,   9I,  (92)      |(Loop) Add ith coefficient to a
     342,  0,   0,   I(99)     |Multiply by y squared
     203,  I26, 9I,  -2(0)     |Cycle, forming polynomial in y squared
     3I0,  0,   0,   (93)      |Add 0th coefficient, giving sin y/y
     342,  0,   0,   (99)      |Multiply by y
     I2I,  I26, 97,  *7777065  |SIN/COS EXIT
97)  *000I2I37/*I4066712       |I/2π
96)  *03200000/*00000000       |+0, with exponent I3
95)  *00040000/*00000000       |I/2
94)  *00I60000/*00000000       |-I/4
93)  *004I444I/*76652I03       |)
92)  *0072652I/*03065616       |)
     *0I050632/*740I53I3       |)Coefficients for sin/cos
     *0I354645/*664I6023       |)
     *0I252005/*0I240643       |)
     *0I306330/*74I63500       |)
9I)  *76737740/*00000000       |Constant for Log. -(256½) x8 to power -8
26)  I2I,  97,  0,   *4004I00  |(I7I3) Set link
     235,  I26, 0,   (90)      |Jump if a ≠ 0 to form log am, exit to (5I)
     325,  0,   II9, 0         |(a=0) Set -s in A
     236,  I24, 0,   *3        |If s ≤ 0, set exponent ='+I92', preparing for EO
     237,  I24, 0,   *5        |If s >0, set exponent ='-I92', preparing for exp underflow
     I2I,  I26, 0,   (89)      |Jump to tail of Exp to set EO or EU, exit to (50)
5I)  342,  0,   II9, 0         |(I7I3 continues) (Log am in A) Multiply by s
35)  I2I,  97,  0,   *4004I00  |(I702/3 join) EXPONENTIAL Set link to exit to (50)
36)  360,  0,   0,   (0)       |(I402 continued joins) Standardize (=x say)
     I2I,  9I,  0,   *4        |Set *4 in B9I
     2I7,  I26, I24, I0(0)     |Jump if exponent negative (x small)
     I2I,  92,  I24, *774      |Set b92 = exponent -4
     2I6,  I26, 92,  -9(0)     |Jump for out of range if exponent ≥ 4
     342,  0,   0,   (46)      |Multiply by log e to base 8
     330,  0,   0,   (44)      |Add ½ and fix, i.e. (Int.pt)R+ in ½ word position
     356,  0,   0,   (99)      |Store
     II3,  9I,  0,   0.4(99)   |Store *4 in l.s.½, i.e. clear frac.pt. and add ½
     3II,  0,   0,   (99)      |Subtract from a, i.e. result =x log e -(Int.pt)R+
     342,  0,   0,   (45)      |Multiply by log 8 to base e, i.e. unscale remainder
     I0I,  9I,  0,   (99)      |Set (Int.pt)R+ at bottom of B9I
     356,  0,   0,   I(99)     |Store a, =z say
     372,  0,   0,   I(99)     |z squared
     356,  0,   0,   2(99)     |Store
     330,  0,   0,   (49)      |Add p
     372,  0,   0,   I(99)     |Multiply by z
     356,  0,   0,   (99)      |Store, =(z squared + p) z, =w say
     334,  0,   0,   (47)      |q
     372,  0,   0,   2(99)     |Multiply by z squared
     I2I,  92,  0,   2.0       |Set count
     330,  0,   0,   (48)      |Add r
     30I,  0,   0,   (99)      |Subtract w
     356,  0,   0,   2(99)     |Store
     334,  0,   0,   (99)      |w
     330,  0,   0,   (99)      |2w
     374,  0,   0,   2(99)     |2w/(qzz+r-w), = exp (z/8) -I approx
     I2I,  I26, 0,   4(0)      |Jump into loop
     356,  0,   0,   (99)      |Store, = v say) Generate successively
     330,  0,   0,   (43)      |Add 2          ) exp(z/4)-I, exp(z/2)-I
     342,  0,   0,   (99)      |Multiply by v ) exp z-I, keeping accuracy
     I25,  9I,  0,   0         |)Shift b9I up 5 places each time round the loop
     I63,  9I,  0,   0         |)i.e. end up with (Int.pt)R+ in exponent position
     203,  I26, 92,  -5(0)     |Cycle
     320,  0,    0,    (42)          |(Exp continued) Add 1, i.e. result = exp z
     124,  124,  91,   *001          |Adjust exponent, adding (Int.pt)R+ +1, result =8 exp aq
 69) 365,  0,    0,    (0)           |)Shift down to ensure unstandardized) result = exp aq
     340,  0,    0,    (0)           |)and set EO or EU if appropriate    )
     121,  126,  97,   *7776445      |EXIT FROM EXPONENTIAL
 49) *01017006/*40314262             |)
 48) *01217006/*40314334             |)Coefficients for Exponential
 47) *00460021/*25613606             |)
 46) *00036616/*04734165             |Log e to base 8 ) Constants for Exponential
 45) *00220505/*31077170             |Log 8 to base e)
 44) *01200000/*40000000             |Constant for Exponent. ½,fixed with point at ½ word posn.
 43) *00220000/0                     |+2
 42) *00210000/0                     |+1
 15) 362,  0,    0,    2(99)         |(1402,13 continued) (cos s* in A) Multiply by s (1413),
     356,  122,  0,    0             |    or exp s (1402) and store as real part of ca.
     324,  0,    119,  1             |Set s* in A
     121,  97,   0,    *40036351     |Set link to exit to 2(0)
     121,  126,  0,    (86)          |Jump to form sin s*, exit to 1(0)
     362,  0,    0,    2(99)         |Multiply by s(1413) or exp s (1402)
     756,  122,  0,    1             |Store as imaginary part of ca and exit
 24) 121,  97,   0,    *4004011      |(1700,1) LOG Set link to exit to (50)
 90) 237,  126,  0,    (76/1400)     |Jump for monitor if a<0
     360,  0,    0,    (0)           |(1713 continued joins) Standardize
 27) 234,  126,  0,    (76/1400)     |(1400 continued joins) Jump for monitor if =0
     121,  91,   124,  *4            |Set b91'= exponent +256
     121,  124,  0,    0             |Set exponent =0
     320,  0,    0,    (41)          |Add p
     356,  0,    0,    1(99)         |Store x+p
     300,  0,    0,    (40)          |Add -2p, = x-p
     374,  0,    0,    1(99)         |Divide by x+p
     121,  92,   0,    6             |Set count
     113,  91,   0,    0.4(99)       |Store exponent +256 in lower half of (99)
     356,  0,    0,    1(99)         |Store (x-p)/(x+p), = z say
     362,  0,    0,    1(99)         |Square
     113,  0,    0,    (99)          |Clear top half of (99), i.e.c(99)=(exp +256) x 8 to power -8
     346,  0,    0,    2(99)         |Store z squared.
     300,  0,    92,   (37)          |(Start of loop) Add coefficient )Form
     342,  0,    0,    2(99)         |Multiply by z squared           )polynomial
     203,  126,  92,   -2(0)         |Cycle                           )in z squared
     300.  0,    0,    (38)          |Add 0th coefficient
     352,  0,    0,    1(99)         |Multiply by z, ÷ (log x+½ log 8) x 8 to power -8
     300,  0,    0,    (99)          |Add(exp +256) x 8 to power -8,double length
     310,  0,    0,    (91)          |Add -(256½) x 8 to power -8
     362,  0,    0,    (39)          |Multiply by ln8 x 8 to power 8; result = log x
     121,  126,  97,   *7776534      |EXIT FROM LOG
 41) *00026501/*17146376             |''p''               )
 40) *00122575/*41463003             |-2p                 )Constants for log
 39) *02220505/*31077170             |ln8 x 8 to power 8)
 60) 724,  0,    0,    (99)          |(1524,5 continued) Set am'=c(99) Q, and exit
     0     /     0                   |)
     0     /     0                   |)
     0     /     0                   |)
     0     /     0                   |)Spare
     0     /     0                   |)
     0     /     0                   |)
     0     /     0                   |)______
 38) *76075434/*11670327             |)
 37) *76024411/*30520752             |)
     *76014237/*13253256             |)Coefficients For Log.
     *76010630/*11271374             |)
     *75666171/*30127254             |)
     *75661015/*40021262             |)
     *75621422/*02664134             |)
     *76011735/*74545451             |)
```

9.6 The 1600 Extracodes

```
    (0)=1536*4
    +0/0                           |Unassigned
    101,  98,   119,  0            |1601 g'=s    Store m.s. ½
    501,  99,   119,  0.4          |Store l.s. ½ and exit
    +0/0                           |Unassigned
    121,  126,  0,    0.1(5)       |1604 g'=g+s    Jump with marker
    121,  126,  0,    (5)          |1605 g'=g+s with end-around-carry
    121,  126,  0,    (6)          |1606 g' =g ≠s
    107,  98,   119,  0            |1607 g' = g&s    M.s. ½
    507,  99,   119,  0.4          |L.s. ½ and exit
    126,  98,   119,  0            |1611 g' = not g.    M.s.½
    526,  99,   119,  0.4          |L.s.½ and exit
    113,  98,   119,  0            |1613 s'=g    M.s. ½
    513,  99,   119,  0.4          |L.s. ½ and exit
    113,  98,   0,    (99)         |1615 am'=g    M.s.½ to store
    113,  99,   0,    0.4(99)      |L.s.½ to store
    734,  0,    0,    (99)         |Transfer to A and exit
24) 113,  0,    0,    0.4(99)      |(1624) Clear l.s. ½ of word
    101,  91,   119,  0            |Set b91=h
    113,  91,   0,    (99)         |Store b91 in m.s. ½
    734,  0,    0,    (99)         |Set am and exit
    121,  126,  0,    (24)         |1624 am'=h
    +0/0                           |Unassigned
    121,  126,  0,    (92)         |1626 h'=am
    +0/0                           |Unassigned
    147,  98,   119,  0            |1630 g' = g & (not s)
    147,  99,   119,  0.4          |g'=gvs
 6) 106,  98,   119,  0            |(1606 joins)
    506,  99    119,  0.4          |g'=g≠ s (=g & not s for 1630) . Exit
    +0/0                           |Unassigned
    356,  0,    0,    (99)         |1635 g'=am    Store am
    101,  98,   0,    (99)         |M.s. ½ in g
    501,  99,   0,    0.4(99)      |L.s. ½ and exit
92) 356,  0,    0,    (99)         |(1626) Store am
    101,  92,   0,    0.4(99)      |Extract l.s.½
    215,  92,   92,   0.1          |Set b92 =0.1 if l.s. ≠ 0
    147,  92,   0,    (99)         |Extract m.s. ½ of g, oring 0.1 at bottom if l.s.½ ≠ 0
    513,  92,   119,  0            |(i.e. Atlas type rounding) Store in h and exit
    +0/0                           |Unassigned
    147,  98,   119,  0            |1646 g'=gvs    M.s. ½
    547,  99,   119,  0.4          |L.s. ½ and exit
    +0/0                           |Unassigned
    +0/0                           |Unassigned
    152,  98,   119,  0            |1652 bt' =g - s.  Set bt from difference of m.s. halves
    225,  126,  0,    3(0)         |Jump if non-zero to ignore l.s. halves
    152,  99,   119,  0.4          |If zero, set bt from difference of l.s. halves
    224,  126,  0,    (96)         |Jump to exit if zero
    101,  91,   0,    6*6          |Extract V6 (l.s. digit = Bcarry)
    163,  91,   0,    0            |Shift Bcarry to sign position
    572,  91,   0,    0            |Set bt from Bcarry and exit
 5) 104,  99,   119,  0.4          |(1604,5) Add l.s.½ halves
    101,  91,   0,    6*6          |Extract V6 (l.s. digit = Bcarry for l.s.½)
    104,  98,   119,  0            |Add m.s. halves
    210,  126,  126,  6(0)         |Jump if 1604
    101,  92,   0,    6*6          |Extract V6 (l.s. digit = Bcarry for m.s.½)
    164,  98,   91,   0.1          |Add carry from l.s.½ into m.s.½
    147,  92,   0,    6*6          |Set digit 23 of B92 =1 if Bcarry set by -1(0) or -4(0)
    164,  99,   92,   0.1          |Add into l.s.½
    101,  91,   0,    6*6          |Extract V6
    564,  98,   91,   0.1          |(1604 rejoins) Add final carry, if any,
73)    *00037777 / *76660000       |)           |from l.s.½ to m.s. ½, and exit
       *00037756 / *67142647       |)Coefficients
75)    *00026501 / *17146376       |)  for square root routine
76)    *01200000 / *40000000       |)
77)    *00040000 /     0           |+½
```

```
10) 165,  93,   119,  -1           |(1410) Set b93 from b119 removing octal fraction
    121,  97,   0,    -3(47)       |Set link to exit to (47)
12) 324,  0,    118,  1            |(1412 joins, B126 odd; 1400 cont.) Set a = sx, = v say
    237,  93,   93,   0.1          |Set b93 odd if v<0
14) 356,  0,    0,    (99)         |(1712 joins with B126 odd) Store v
    362,  0,    0,    (99)         |v squared
    356,  0,    0,    1(99)        |Store
    324,  0,    119,  0            |s
    362,  0,    119,  0            |s squared
    320,  0,    0,    1(99)        |Add v squared                            |for 1412,1710,12
22) 210,  97,   126,  -3(96)       |(1710,1 joins with B126 odd) Set link to exit to(96)
    360,  0,    0,    (0)          |SQUARE ROOT  Round single length, = x' say
94) 234,  126,  97,   3            |(1720,1,2,3 continued) Exit if a=0(short cut)
    237,  126,  0,    (45)         |Jump to error exit if a<0
26) 165,  91,   124,  *001         |(1410 second entry) Least sig. digit of exponent to b91
    356,  0,    0,    (99)         |Store x'
    215,  91,   91,   *00037310    |)Set 1st approximation to sqrt x' in 1(99), =y0 say
    124,  91,   124,  *00062343    |)÷ 4th root of 1/8, with ½ exp of x', if exp even
    113,  91,   0,    1(99)        |)÷(1/8) to power 3/4, with exp ½(b124+1), if exp odd
    121,  124,  0,    0            |Force b124=0, giving a'=x0, say
    300,  0,    0,    (75)         |Add constant
    342,  0,    0,    1(99)        |Multiply by y0 to give linear approximation, =y1 say
    121,  92,   0,    1            |Set count for two cycles of loop
    356,  0,    0,    1(99)        |Store y         )
    334,  0,    0,    (99)         |x               )
    374,  0,    0,    1(99)        |Divide by y     )y(n+1)=½(x/y(n)+y(n))
    300,  0,    0,    1(99)        |Add y           )
    342,  0,    92,   (73)         |Multiply by ½   )
    203,  126,  92,   -5(0)        |Cycle           )
    356,  0,    0,    1(99)        |Store                    )Last iteration
    343,  0,    0,    1(99)        |Multiply by -y,          )y(n+1)=y(n)
    310,  0,    0,    (99)         |Add x double length      )+½(x-y(n)squared)/y(n)
    373,  0,    0,    (76)         |Multiply by -½           )
    374,  0,    0,    1(99)        |Divide by y              )
    302,  0,    0,    1(99)        |Negate and add y, d.l.)
    121,  126,  97,   3            |SQUARE ROOT EXIT to b97+3
    +0/0                           |Spare
45) 121,  91,   0,    2.4          |(Square root error exit, argument negative). Set marker
    121,  126,  0,    187*4001     |Jump to Monitor
47) 235   126,  0,    3(0)         |(1410 continued, with mod s: in A) Jump if ≠0 (normal)
    356,  122,  0,    0            |)If =0, set ca'=0
    756,  122,  0,    1            |)  and exit
    356,  0,    0,    (99)         |If mod s: ≠ 0, store
    367,  0,    93,   0            ||s|
    320,  0,    0,    (99)         |Add mod s:
    362,  0,    0,    (77)         |x½
    121,  95,   0,    *001         |Set mask
    121,  97,   0,    -3(60)       |Set link to exit to (60)
    121,  126,  0,    (26)         |Jump to form sq.rt(½(mod s: + |s|))
60) 107,  95,   93,   0            |)
    214,  126,  95,   4(0)         |(Jump if s≥0, setting b95=0
    121,  95,   0,    -1           |Set b95=-1 if s<0
    211,  126,  93,   2(0)         |Jump if s*≥0
    322,  0,    0,    (86)         |If s and s* both <0, negate accumulator
    356,  0,    0,    (99)         |Store, as z say
    324,  0,    93,   1            |)
    362,  0,    0,    (77)         |)½s*
    374,  0,    0,    (99)         |Divide by z
    356,  122,  95,   1            |Store as real pt. of Ca if s<0, imag pt if ≥0
    120,  95,   0,    0            |Negate b95
    324,  0,    0,    (99)         |z
    756,  122,  95,   0            |Store as real pt. of Ca if s≥0, imag. pt if <0. Exit
    +0/0                           |Spare
    +0/0                           |Spare
```

```
25) 360,  0,    0,    (0)        |(I724/5) Standardize
    I2I,  96,   0,    0          |Set marker
    I2I,  97,   0,    (96)       |Set link to exit to Dummy Exit
3)  234,  I26,  0,    (95)       |ARCTAN/COT. Jump to short cut if =0
    I2I,  92,   I24,  *777       |Set b92= exponent minus one
    236,  I26,  0,    3(0)       |Jump if a >0
    366,  0,    0,    (0)        |Otherwise set positive
    I26,  96,   0,    0.5        |and reverse digits 2I,22 of B96
    2I7,  I26,  92,   5(0)       |Jump if |x| <I
    356,  0,    0,    (99)       |)Otherwise form
    334,  0,    0,    (94)       |)reciprocal
    374,  0,    0,    (99)       |)and reverse digit 23 of B97
    I26,  97,   0,    0.I        |)
    2I7,  I26,  I24,  7(0)       |Jump if |x'|<I/8
    330,  0,    0,    (8I)       |Add I/u [u=tan (½(arctan I/8+π /4))]
    356,  0,    0,    (99)       |Store
    330,  0,    0,    (80)       |Add -(u+I/u), i.e. result = x-u
    372,  0,    0,    (8I)       |Multiply by I/u
    374,  0,    0,    (99)       |Divide by x+I/u. Result = (x-u)/(I-ux)
    I2I,  92,   0,    0.I        |Mark B92 odd for |x'|>I/8
    356,  0,    0,    (99)       |Store as y
    342,  0,    0,    (99)       |y squared
    I2I,  9I,   0,    4.0        |Set count
    346,  0,    0,    I(99)      |Store y squared, clear A
    300,  0,    9I,   (83)       |(Power series loop) Add coefficient
    342,  0,    0,    I(99)      |Multiply by y squared
    203,  I26,  9I,   -2(0)      |Cycle
    330,  0,    0,    (82)       |Add first coefficient
    342,  0,    0,    (99)       |Multiply by y
    2II,  I26,  92,   2(0)       |Jump if x' small
    330,  0,    0,    (84)       |Otherwise add arctan u (approx)
95) 2II,  I26,  97,   2(0)       |Jump if b97 even
40) 302,  0,    0,    (85)       |If b97 odd (cos, x<I; sin, x>I; tan, x>I),
    300,  0,    96,   0.2(86)    |Add 0 or -π                  |form π/2-result
    2II,  I26,  96,   2(0)       |Jump if b96 even (I72I to 5,x>0;I4II,I726, s>0)
    302,  0,    0,    (86)       |Otherwise negate result
    I2I,  I26,  97,   0          |ARC TAN/COT EXIT to b97
II) 334,  0,    II9,  I          |(I4II,bI26 odd, and I400 continued) Set s* in A
46) 2I0,  97,   I26,  (96)       |(I726, BI26 odd) Set link to exit
    356,  0,    0,    (99)       |Store a, =x say            |for I4II, I726
    I2I,  96,   0,    0          |Clear marker
    324,  0,    II9,  0          |s
    234,  I26,  0,    8(0)       |Jump if zero
    236,  I26,  0,    3(0)       |Jump if >0
    366,  0,    0,    (0)        |)If <0, take modulus
    I2I,  96,   0,    I.5        |)and set marker
    356,  0,    0,    I(99)      |Store |s|
    324,  0,    0,    (99)       |Bring back x
    374,  0,    0,    I(99)      |Divide by |s|
    I2I,  I26,  0,    (3)        |Jump, form arctan. I4II,I726 exit to (96); I400 to (I5
    345,  0,    0,    (99)       |(Here if s=0) Set x in L, sign thro' M, exp unchngd (
    234,  I26,  97,   0          |Exit if a=0 (i.e. if x=0 also) with result =0
    237,  96,   0,    0.I        |If x≠0, set b96 odd if x < 0
    I2I,  I26,  0,    (40)       |Jump with A effectively containing zero, to form ± π /
96) 52I,  0,    0,    (0)        |Dummy exit
I)  I2I,  97,   0,    (I5)       |(I400) Set link
    I2I,  I26,  0,    (I2)       |Jump to form sq.rt (s.s + s*.s*) . Exit to 3(I5)=(I8
7I) 235,  I26,  0,    (93)       |(I720,I,2,3 continued) Sq.rt (I-x.x) in A. Jump if ≠0
    324,  0,    0,    3(99)      |If zero, recover x (=+I)
    342,  0,    0,    (85)       |Multiply byπ/2 (a'=±  π/2)
    I2I,  I26,  0,    (95)       |Jump to adjust for sin/cos. Exit to (96)
    +0/0                         |Spare
    +0/0                         |Spare
    +0/0                         |Spare
    +0/0                              |Spare
93) 356,  0,    0,    I(99)           |(I720-3 continued) a=Sq.rt(I-xx), (≠0). Store
    324,  0,    0,    3(99)           |x
    374,  0,    0,    I(99)           |Divide by sq.rt(I-xx)
    I2I,  I26,  0,    (3)             |Jump to form arctan/cot.Exit to (96)
2I) 356,  0,    0,    3(99)           |(I722,3 with 0.7 in BI26;I720,I) Store aq (=x)
    373,  0,    0,    3(99)           |Form -x squared
    I2I,  97,   I26,  -I8             |Set link (exit to 3(96)=(7I)Form SQ.RT; exit to (96)
    3I0,  0,    0,    (87)            |Form I-x.x                    |Form arctan/cot)
    I65,  96,   97,   0.2             |b96'=0(I720,I)or 0.2(I722,3)
    236,  I26,  0,    (94)            |Jump to form sq.rt(I-x.x),if ≥0. Exit to 3(96)=(7I)
    I2I,  9I,   0,    4               |IfI-x.x<0, Set mark
    I2I,  I26,  0,    I87*400I        |and jump to Monitor for error.
I5) 356,  I22,  0,    I               |(I400 continued) a=arctan (s*/s). Store as imag.pt.of
    334,  0,    0,    4(99)           |Bring back sq.rt.(s.s+s*.s*)
    I2I,  I26,  0,    (27/I500)       |Jump to form log. Exit to (95/I400)
I8) 356,  0,    0,    4(99)           |(I400 continued) a=sq.rt(ss+s*.s*). Store
    I2I,  I26,  0,    (II)            |Jump to form arctan(s*/s). Exit to (I5)
80) *00353565/*6753II22               |)
8I) *00220266/*6574I5II               |)
82) *00777777/*7777773I               |)
83) *00I52525/*25332676               |)
    *000I463I/*34747I75               |) Coefficients for Arctan/Cot
    *00I66674/*23667077               |)
    *77667345/*25I0037I               |)
    *7773503I/*054I0443               |)
84) *0003507I/*3I247463               |)
85) *002I444I/*76652I04               |  π/2
86) *40000000/*00000000               |Floating-point zero
    *00346674/*02253570               |-π
87) *002I0000/*00000000               |+I
```

0023 1103 7552 4210

9.7 The I700 Extracodes

```
    (0)=I792*4
    324,  0,    II9,  0                |I700 am'=log s.  Set aq'=s
    I2I,  I26,  0,    (24/I500)        |I70I am'=log aq.  Jump I700,I
    324,  0,    II9,  0                |I702 am'=exp s.  Set aq'=s
    I2I,  I26,  0,    (35/I500)        |I703 am'=exp aq  Jump I702,3
    334,  0,    II9,  0                |I704 a'=int.pt.s.Set a'=s
    I2I,  I26,  0,    (5)              |I705 a'=int.pt.a.Jump I704,5
    334,  0,    II9,  0                |I706 a'=sign s   Set a'=s
    I2I,  I26,  0,    (7)              |I707 a'=sign a   Jump I706,7
    324,  0,    II9,  0                |I7I0 am'=sq.rt.s Set aq'=s
    I2I,  I26,  0,    0.I(22/I600)     |I7II am'=sq.rt.aq.  Jump I7I0,I
    I2I,  I26,  0,    0.I(I4/I600)     |I7I2 am'=sq.rt.(aq.aq+s.s).  Jump
    I2I,  I26,  0,    (26/I500)        |I7I3 am'=am to power s.  Jump
    I2I,  I26,  0,    (I4)             |I7I4 am'=I/s.  Jump
    356,  0,    0,    (99)             |I7I5 am'=I/am.  Store am
    325,  0,    0,    (96)             |Set I in A
    774,  0,    0,    (99)             |Divide by''am'' and exit
    324,  0,    II9,  0                |I720 am'=arcsin s.  Set aq'=s
    I2I,  I26,  0,    (2I/I600)        |I72I am'=arcsin aq.  Jump I720,I
    324,  0,    II9,  0                |I722 am'=arccos s.  Set aq'=s
    I2I,  I26,  0,    0.7(2I/I600)     |I723 am'=arccos aq.  Jump I722,3
    324,  0,    II9,  0                |I724 am'=arctan s.  Set aq'=s
    I2I,  I26,  0,    (25/I600)        |I725 am'=arctan aq.Jump I724,5
    I2I,  I26,  0,    0.I(46/I600)     |I726 am'=arctan(aq/s)
    I2I,  I26,  0,    (27)             |I727 c'=c+I,2 or 3 as am >,=,< s
    324,  0,    II9,  0                |I730 am'=sin s Set aq'=s
    I2I,  I26,  0,    0.I(3I/I500)     |I73I am'=sin aq.  Jump I730,I
    324,  0,    II9,  0                |I732 am'=cos s.  Set aq'=s
    I2I,  I26,  0,    (3I/I500)        |I733 am'=cos aq.  Jump I732,3
    324,  0,    II9,  0                |I734 am'=tan s.  Set aq'=s
    I2I,  I26,  0,    *4004400         |I735 am'=tan aq  Jump I734,5
    I24,  I26,  0,    0.5              |I736 c'=c+2 if |am| ≥ s. Add 0.5 to bI26
    356,  0,    0,    (99)             |I737 c'=c+2 if |am| < s. Store am
    366,  0,    0,    (0)              |Form |am|
    32I,  0,    II9,  0                |Subtract s
    237,  I26,  I26,  I.3              |Jump if |am|-s<0, to 2(0) if I737, to 3(0) if I73 6
    2II,  I26,  I26,  2(0)             |(|am|-s>0) Jump if I737
    I24,  I27,  0,    I                |(I736, |am|-s≥0; I737, |am|-s<0) Set c'=c+2
    734,  0,    0,    (99)             |Recover am and exit
94) *064  /     0                      |+0, exponent 26
96) *00I  /     0                      |Floating-point -I
98) *4    /     0                      |Floating-point zero
93) *0004 /     0                      |Floating-point +½
    I2I,  I26,  0,    (52)             |I752 m'=ax, exp I2; ay'=ay -I2
    I2I,  I26,  0,    (53)             |I753 ax'=m, exp I2; ay = ay +I2
    2II,  I26,  I24,  (54)             |I754 Round am by R+, Q. Jump when A free
    I2I,  I26,  0,    (55)             |I755 ax'=ax, exp (ay-n); ay'=n
    I2I,  I26,  0,    (56)             |I756 s'=am,am'=s
    I2I,  I26,  0,    (57)             |I757 am'=s/am
    356,  0,    0,    (99)             |I760 am'=am squared
    762,  0,    0,    (99)
    I2I,  I26,  0,    (62)             |I762 m'=ax, exp I2
    I2I,  I26,  0,    (63)             |I763 ax'=m, exp -I2
    I20,  II9,  0,    0                |I764 ax'=ax, exp n.  Set bII9=-n
    I2I,  I26,  0,    (65)             |I765 ax'=ax, exp -n. Jump I764,5
    334,  0,    II9,  0                |I766 am'=|s|, X. Set s in am
    236,  I26,  0,    (97)             |I767 am'=|am|,X. Jump if a ≥ 0 to exit
    732,  0,    0,    (98)             |Set a'=-am+0, ie. negate am, and exit
97) 52I,  0,    0,    0                |I77I bI2I'=Ba, bII9'=N+ba+bm. Dummy A-type extracode
    I2I,  I26,  0,    (72)             |I772 m'=(m.sx), exp I2; ay'=ay+sy-I2
    I2I,  I26,  0,    (73)             |I773 m'=(ax/sx), exp (ay-sy-I2); ay'=I2
    347,  0,    0,    (0)              |I774 am'=am/s. Clear 1
    I2I,  I26,  0,    (74/I500)        |I775 am'=a/s.  Jump I774,5
    I2I,  I2I,  0,    0                |I776 Remainder and quotient. Set bI2I=0
    I2I,  I26,  0,    (76)             |Jump
```

|1700 Extracodes, Page 2

```
5)  217,  124,  124,  0         |(1704,5) Set exponent = 0 if negative
    710,  0,    0,    (94)      |Add 0 with exp 26 (ie shift int.pt. to bottom of L). Exit
27) 356,  0,    0,    (99)           |(1727) Store am
    321,  0,    119,  0         |am-s
    234,  127,  127,  1         |Add 1 to b127 if am=s
    237,  127,  127,  2         |Add 2 to b127 if am<s
    734,  0,    0,    (99)      |Restore am and exit
14) 325,  0,    0,    (96)      |(1714) Set +1 in A
    774,  0,    119,  0         |Divide by s and exit
57) 356,  0,    0,    (99)      |(1757) Store am
    324,  0,    119,  0         |Bring out s
    774,  0,    0,    (99)      |Divide by am and exit
54) 101,  91,   0,    6*6       |(1754) Extract V6
    354,  0,    0,    (0)       |R+
    300,  0,    0,    (98)      |Add zero and standardize, i.e. shift down
    513,  91,   0,    6*6       |Restore V6 and exit         |if result superstandard
53) 124,  124,  0,    *014      |(1753)  Add 12 to exponent
63) 356,  0,    0,    (99)      |(1763 joins) Store am
    345,  0,    0,    (99)      |Set in L
    764,  0,    0,    (0)       |Shift up one octal place and exit
56) 356,  0,    0,    (99)      |(1756) Store am
    334,  0,    119,  0         |Set s in A
    356,  0,    0,    1(99)     |Store
    334,  0,    0,    (99)      |Recover am
    356,  0,    119,  0         |Store in s
    734,  0,    0,    (99)      |Reset s in A and exit
72) 352,  0,    119,  0         |(1772) Multiply a by s
52) 122,  124,  0,    *014      |(1752 joins) Subtract 12 from exponent
62) 121,  91,   124,  1         |(1762 joins) Preserve exponent in B91. Also set d20=1
    121,  124,  0,    *014      |Set exponent =12
    121,  126,  0,    (71)      |Jump
73) 121,  92,   0,    *014      |(1773) Set b92=12 in exponent position
    121,  121,  0,    46        |Set B121 to point at B92
41) 340,  0,    0,    (0)       |(1473 joins)  Standardize
    356,  0,    0,    1(99)     |Store am
    324,  0,    119,  0         |s, standardized
    356,  0,    0,    2(99)     |Store
    324,  0,    0,    1(99)     |Bring back am
    374,  0,    0,    2(99)     |Divide by s
47) 113,  122,  0,    (99)      |(1452 continued joins) Store ba (=b92 =12 if 1773)
    101,  119,  0,    (99)      |Set into B119
55) 122,  119,  124,  *4        |(1755 joins) Set b119'=b119-b124+256 in exponent position
    124,  124,  119,  *4        |Set original b119 in B124
    125,  119,  0,    0         |)Shift b119 to integer position and subtract 256,
    125,  119,  0,    -256      |)i.e. original b119-b124 in integer posn with sign propagated
65) 121,  91,   124,  1         |(1764,5 joins) FIXING ROUTINE.Preserve exp in B91, set d20=1
    217,  126,  119,  5.1(0)    |Jump if b119 <0, i.e. shift up required, set marker in B126
    214,  126,  119,  (97)      |Jump to exit if b119=0
    120,  119,  0,    1         |(Shift down) Negate and add 1
    365,  0,    0,    (0)       |Shift down one (ensures correct handling of superstandard nos.)
    214,  126,  119,  (97)      |Jump to exit if b119 now =0, i.e. one shift only was required
    121,  92,   119,  27        |(Shift up rejoins)) Set b119=-27 if b119 <-27
    217,  119,  92,   -27       |                        )i.e. if out of range
    125,  119,  0,    0         |)Shift b119 to exponent position
    125,  119,  0,    0         |)
    211,  126,  126,  6(0)      |Jump if shift down
    123,  124,  119,  *777      |SHIFT UP. set b119 +vely in B124, correcting for 7777 at bottom
71) 340,  0,    0,    (0)       |(1752,62,72 cont join) Standardize i.e. shift up adjusting b124
    217,  126,  124,  4(0)      |Jump if exponent now -ve. i.e. shifted too far
    203,  126,  124,  8(0)      |Jump if exponent >0, i.e. more shift up reqd. Subtract 1
    521,  124,  91,   0         |If exp = 0(i.e. correctly shifted) recover original exp & exit
    121,  124,  119,  0         |SHIFT DOWN. Set b119 (negative) in B124
    310,  0,    0,    (93)      |Add ½ with exponent zero, i.e. shift down correctly and add ½
    357,  0,    0,    (99)      |Preserve l.s. ½
```

```
     331,  0,    0,    (93)     |(Shift down cont) Remove ½ from top(no shifting) and clear L
     344.  0,    0,    (99)     |Recover l.s. ½
     521,  124,  91,   0        |Recover original exponent and exit
     364,  0,    0,    (0)      |(Here if shift up beyond standard required) Shift up
     203,  126,  124,  -1(0)    |Cycle counting
     147,  91,   0,    6*6      |)Set A0 by 'or-ing',
     121,  124,  91.   0        |)        recover original exponent
     513;  91.   0,    6*6      |)        and exit
 7)  237,  126,  0,    3(0)     |(1706,7)  Jump if -ve
     234,  126,  0,    2.4(0)   |Jump if zero
     725,  0,    0,    (96)     |If positive set +1 in A and exit
     734,  0,    126.  -100.4   |Set -1 or 0 in A if -ve or zero. Exit
77)  356,  0,    0,    2(99)    |(1477 with 0.4 in B126) REMAINDER.Store quotient
     300,  0,    0,    (94)     |Take integer part,= Q SAY
76)  356,  0,    119,  0        |(1776 joins with b121=0) store Q
     342,  0,    0,    (99)     |x denominator (s)                        )
     356,  0,    0,    4(99)    |Store m.s.½                              )
     355,  0,    0,    (0)      |Shift 1 to m                             )Form a-Qs
     302,  0,    0,    3(99)    |Negate, add l.s ½ , of numerator (a)     ) (=R say)
     356,  0,    0,    3(99)    |Store                                    )
     334,  0,    0,    4(99)    |Bring back Q.s (m.s.½)                   )
     302,  0,    0,    1(99)    |m.s.½ of a-Q.s.                          )
     310,  0,    0,    3(99)    |Add l.s.½ of ditto                       )
     214,  126,  121,  9(0)     |Jump to exit if 1477 Ba=0 or 1776
     356,  0,    0,    4(99)    |Store Rm
     234,  126,  0,    7(0)     |Jump to exit if R=0
     113,  0,    0,    3(99)    |Clear store line
     314.  0,    121,  (99)     |Read denominator, numerator, quotient or zero
     237,  121,  121,  0.4      |Change d21 if  <0 ) Set d21 of B12151
     314,  0,    0.    4(99)    |Recover Rm         )   if remainder not
     237,  121,  121,  0.4      |Change d21 if <0   )   of required sign
     164,  126,  121,  0.4      |Skip if remainder wrong sign
     521,  0,    0,    0        |Exit if remainder correct sign
     311,  0,    0,    (99)     |If remainder wrong sign, form R-s, = a-(Q+1)s
     356,  0,    0,    4(99)    |Store (R-s)m
     314,  0,    0,    (92)     |Set +1 in Am
     357,  0,    0,    3(99)    |Store (R-s)1
     320,  0,    119,  0        |Add 1 to Q (i.e. adjust )
     344,  0,    0,    3(99)    |Recover (R-s)1 in L
     356,  0,    119,  0        |Store adjusted Q
     714,  0,    0,    4(99)    |Recover (R-s)m and exit
92)  *0021 /     0              |Floating-point +1
80)  330,  0,    0,    (94)     |(1476,B126 odd) FIXED PT. DIVISION Take int.pt of am (exp=26)
     124,  124,  0,    *776     |Subtract 2 from exponent correcting for 1(0) and 19(0)
     364,  0,    0,    (0)      |Shift up  a  so that binary point is 3 places from foot of L
81)  121,  93,   0,    -2       |(1474,5 join) Set mask
     123,  91,   124,  *7461    |Set b91 = 25 - exponent, in exponent position, plus *0007
     236,  126,  126,  5        |Jump if a>0, preserving marker in B126
     356,  0,    0,    (99)     |OTHERWISE NEGATE A.  I.e.store m
     355,  0,    0,    (0)      |             shift up 1
     302,  0,    0,    (98)     |             negate
     331,  0,    0,    (99)     |             and add back m negatively
     124,  93,   0,    0.1*4    |     also set marks in B93 (positive, odd)
     375,  0,    119,  0        |Divide by s, quotient in L remainder in M
     121,  92,   0,    -4       |Set mask
     101,  94,   119,  0        |m.s.½ of s                       ) Set b94=0 if
     127,  94,   0,    *00077   |mantissa part except sign digit) mantissa of s =0 or -1.0,
     147,  94,   119,  0.4      |'or' with rest of mantissa       ) set b94 ≠0 otherwise
     214,  126,  94,   (21)     |Jump if mantissa =0 or -1.0 to set D0
     211,  126,  126,  7(0)     |Jump if 1474,5
     356,  0,    0,    (99)     |(1476 continues) Store remainder (R)
     357,  0,    0,    1(99)    |Store quotient Q
     364,  0,    0,    (0)      |Shift up Q one octal place
     107,  91,   0,    1(99)    |3 m.s. bits of Q (mantissa), and reduce exp part to 0
```

|1700 Extracodes, Page 4

```
    215,  126,  91,   (21)    |(1476 continued) Jump for DO if Q too large
    314,  0,    0,    (99)    |Recover R, leaving Q (shifted up) in L
    347,  122,  0,    0       |(1474,5) Store Q, clearing L
    236,  126,  0,    3(0)    |Jump if R≥0
    121,  124,  0,    0       |)If R<0, add 1 (fixed point)
    331,  0,    0,    (96)    |)  to adjust for error due to 375
    104,  92,   0,    6*6     |Clear Bc.  Set b92>0 and reset Bc if |xa|≥|xs|
    104,  93,   0,    6*6     |)Jump to DO if |xa|≥|xs|; otherwise add Qs digit to b93
    216,  126,  92,   (21)    |)i.e.set sign of b93  to ≠ of Q and a
    211,  126,  93,   2(0)    |)Negate R if a<0         |i.e. to not sign of final Q
    332,  0,    0,    (98)    |)  giving true R
    217,  126,  93,   5(0)    |Jump if final Q>0
    356,  0,    0,    (99)    |Store true R,
    335,  122,  0,    0       |)set -Q as final Q in C(ba)
    356,  122,  0,    0       |)
    334,  0,    0,    (99)    |and reset true R in A
    523,  124,  91,   *7631   |Reset exponent for R and exit
21) 374,  0,    0,    (98)    |Cause DO and monitor exit


    (0)=*4004400

    |TAN
    |If x = ½π(n+θ), where -½ ≤ θ <½
    |then tan x = tan (½πθ)= p(θ)/(1-θ.θ) if n even
    |             = -cot(½πθ)= -(1-θ.θ)/p(θ) if n odd

    362,  0,    0,    (88)    |Multiply x by 2/π
    121,  91,   0,    0.1     |Set marker
    217,  126,  124,  (82)    |Jump if small (<1/8)
    320,  0,    0,    (85)    |Add -½
    330,  0,    0,    (86)    |''Fix'', i.e. int.pt. in M, frac.pt. in L
    356,  0,    0,    (99)    |Store int.pt.,= n-1
    355,  0,    0,    (0)     |Set frac.pt.in M
    107,  91,   0,    0.4(99) |Set b91= 0.1 if n-1 odd, 0 otherwise
    300,  0,    0,    (85)    |Add -½ to frac.pt. Result = θ
82) 356,  0,    0,    (99)    |Store θ
    342,  0,    0,    (99)    |Form θ.θ
    121,  92,   0,    3       |Set counter
    356,  0,    0,    1(99)   |Store θ.θ
    330,  0,    0,    (87)    |Add -1. Result = -(1-θ.θ)
    346,  0,    0,    2(99)   |Store -(1-θ.θ). Clear A
    300,  0,    92,   1(89)   |Add coefficient ) Form polynomial
    372,  0,    0,    1(99)   |Multiply by θ.θ )  in  θ.θ
    203,  126,  92,   -2(0)   |Cycle              )
    300,  0,    0,    (89)    |Add 0th coefficient.
    363,  0,    0,    (99)    |Multiply by -θ.θ Result = -p(θ)
    210,  126,  91,   (83)    |Jump if n even
    356,  0,    0,    1(99)   |If n odd, store -p(θ) in 1(99)
    325,  0,    0,    2(99)   |and set+(1-θ.θ) in A
83) 774,  0,    91,   1.7(99) |Divide by c(1(99)) extra) if n odd, result = (1-θ.θ)/-p(θ)
85) *0014/0                   |-½ | by c(2(99)) if n odd, result = -p(θ)/-(1-θ.θ). Exit
86) *032/0                    |0 with exp 13
87) *001/0                    |-1
88) *00050574/*60333447       |2/π
89) *00214441/*76652102       |)
    *00156116/*03120022       |)
    *77767277/*63661370       |) Coefficients for p(θ)
    *77312142/*24070717       |)
    *77116451/*75471372       |)
```

30.8.63.

```
                         H   J00003000   J00001400
                         H   J00000600   J00000300
                         H   J00000140   J00000060
                         H   J00000030   J00000014
                         H   J00000006   J00000003
2751J4                   H   J00000000   J00000000

2759J4                   H   J00000070   J00000060
                         H   J00000050   J00000040
                         H   J00000030   J00000020
                         H   J00000010   J00000000
2763J4                   H   J00000100   J00000000
```

Extracode constants

Address	Value		
Character Masks			
548J4	H	J77	J0077
	H	J000077	7.7
	H	0	7.7
	H	K777.7	J77'
Constants			
0 J4	$+\frac{1}{2}$		
515 J4	+0		
575 J4	exp 0	arg 0	
1419 J4	$1/(2\pi)$		
1420 J4	exp 13	arg 0	
1421 J4	$\frac{1}{2}$		
1422 J4	$-\frac{1}{4}$		
1480 J4	$\log_8 e$		
1481 J4	$\log_e 8$		
1483 J4	+2		
1484 J4	+1		
1755 J4	$\pi/2$		
1757 J4	$-\pi$		
1830 J4	exp 26	arg 0	
1831 J4	-1		