REVISED HARVEST SPECIFICATIONS

28 August 1959

Chapter I

STREAM INSTRUCTIONS

TABLE OF CONTENTS

1.1 INSTRUCTION FORMATS

SBBB

13	3					6		3	3	32
DATA GATING	TAM	PS	RPL	SC	DP	LU OP	1	GS	STOP	ADJ 1
32										32
ADJ 2										ADJ 3
32										32
ADJ 4										ADJ 5

NO OP

20		6	32
	XX0000		ADJ 1

SMER

9		7		6	32
DATA	DIED TYPE		XX0010		ADJ 1

SSER

6	3				7		6	32
	SCD	DATA	DOUD	TYPE		XX0100		ADJ 1

SSEL

11		7		6	32
	LG TYPE		XX0110		ADJ 1

STIR

8		7		6	32
	RDTA DIAD TYPE		XX1000		ADJ 1

SMLU

12		7		6	32
	MODE		XX1010		ADJ 1

SQNL

4		14		6	32
	P0		XX1100		ADJ 1

SILS

12		7		6	32
	LS		XX1110		ADJ 1

1.2 SBBB Stream Byte By Byte

Gates (13) Zero (0) = Closed; no data passes
One (0) = Opened; data passes

0 - SCTR ≡ P
1 - P ⟶ LU
2 - Q ⟶ LU
3 - U ⟶ LU
4 - P ⟶ T
5 - Q ⟶ T
6 - LU ⟶ T
7 - C ≡ LU
8 - L ≡ LU
9 - LU ⟶ R
#10 - U ⟶ R
#11 - LU ⟶ SACC
#12 - U ⟶ SACC

Restrictions:

Gates # 2 and # 3 cannot be open simultaneously. They both refer to one input to the LU.

Gates #5 and #6 cannot be open simultaneously. They both refer to one input to T.

Gates #3 and #6 cannot be open simultaneously. This "figure 8" path is difficult to control.

Gate #3 or Gate #10 cannot be open if Gates #4, #5, and #6 are closed. There must be an input to T if U is extracting.

Gates #7 and #8 cannot be open simultaneously.

Gates #9 and #10 cannot be open simultaneously.

Gates #11 and #12 cannot be open simultaneously.

TAM(3) Table Address Mode

000 - No OP
001 - no memory reference is made. The 26 bit table address is sent directly via the adder to bits 30 55 of U, S_U must be set up. The remainder of U is unchanged.
010 - the reference word is extracted from memory and sent to U
011 - a 1 is or-ed to the referenced word in memory at the position designed by the bit portion of the address.
100 - the successive performance of operations 010 and 011
101 - (fast memory only) a 1 is added to the last position of the cell in memory containing the addressed bit
110 - (fast memory only) the successive performance of operations 010 and 101
111 - no OP

PS(1) - Parallel series

0 - P and Q bytes in parallel
1 - P and Q bytes in series

RPL(1) Replace Bit

0 - no replace of TBA
1 - TBA replaced by just formed address

SD(1) Single Double Column Add/Subtract

0 - 8 bit byte operation
1 - two 4 bit byte operation

CP(1) Carry Propagate mode

0 - no carry propagate
1 - carry propagate

L OP(5) Logic Unit Operation

00000 - 0
00001 - $P \cdot \overline{Q}$
00010 - $P \cdot \overline{Q}$
00011 - $\overline{P}$
00100 - $\overline{P} \cdot Q$
00101 - Q
00110 - $P \neq Q$
00111 - $\overline{P} \vee \overline{Q}$
01000 - $\overline{P} \cdot \overline{Q}$
01001 - $P = Q$
01010 - Q
01011 - $\overline{P} \vee \overline{Q}$
01100 - $\overline{P}$
01101 - $\overline{P} \vee Q$
01110 - $\overline{P} \vee \overline{Q}$
01111 - 1
10000 - MAX (P, Q)
10001 - MIN (P, Q)
10010 - P if P = Q, otherwise all 0
10011 - P if P = Q, otherwise no byte output
10100 - P is P $\neq$ Q, otherwise all 0
10101 - P if P $\neq$ Q, otherwise no byte output
10110 - Q if P $\neq$ Q, otherwise all 0
10111 - Q if P $\neq$ Q, otherwise no output
11000 - P - Q if $P \geq Q$, otherwise all 0
11001 - P - Q if $P \geq Q$, otherwise no byte output
11010 - Q - P is $Q \geq P$, otherwise all 0
11011 - Q - P is $Q \geq P$, otherwise no byte output
11100 - P - Q modulo MOD (MOD _ 255
11101 - Q - P " " "
11110 - P + Q binary (RADIX $\leq$ 255)
11111 - P + Q modulo MOD (MOD $\leq$ 255)

K, L and M for the above are:

	K = 1	L = 1	M = 1
00000	Connection of odd parity	Connection all 0	Connection of even parity but all 0
01111			
10000	$P > Q$	$P = Q$	$P < Q$
11110			
11111	$P + Q \geq M$	0	$P + Q < M$

GS(3) Group Size

000 No OP	100 F_1 Q
001 EOL_1P	101 EOL_1R
010 F_1P	110 W
011 EOL_1Q	111 Z

STOP(3) Stop Code

000	No OP
001	F_2P
010	F_3P
011	F_2Q
100	F_3Q
101	F_2R
110	F_3R
111	End of Sequence

1.3 Adjustments

6 STIMULUS	2 TAG	8 REACTION 1	8 REACTION 2	8 REACTION 3

STIMULI FOR ADJUSTMENTS

0.	No-op	32.	KB = 1 & F ≠ limit
*1.	FL_1 in Q	*33.	MB = 1
*2.	FL_2 in Q	*34.	LB = 1
*3.	FL_3 in Q	35.	KB = 0
*4.	SCTR Steps	*36.	KB = 1
*5.	FL_1 in P	37.	LB = 0
*6.	FL_2 in P	38.	MB = 0
*7.	FL_3 in P	39.	Not used
8.	SCTR ≠ LIM & EOG	40.	MB = 1 & F ≠ limit
9.	SACC < THR & EOG	*41.	MG = 1
*10.	SACC goes / to -	*42.	LG = 1
*11.	SACC ≥ THR	43.	KG = 0
*12.	SCTR = LIM	*44.	KG = 1
*13.	FL_1 in R	45.	LG = 0
*14.	FL_2 in R	46.	MG = 0
*15.	FL_3 in R	47.	Not used
*16.	W	48.	LB = 1 & F ≠ limit
17.	W and X	49.	MB = 1 & F = limit
*18.	$\overline{W}$	50.	LB = 1 & F = limit
19.	W and Y	51.	KB = 0 & F = limit
*20.	Y	52.	KB = 1 & F = limit
21.	X or Y	53.	LB = 0 & F = limit
*22.	EOL in U	54.	MB = 0 & F = limit
*23.	$\overline{Y}$	*55.	F = limit
*24.	Z	56.	F ≠ LIMIT & EOG
*25.	X	57.	MG = 1 & EOG
*26.	$\overline{Z}$	58.	LG = 1 & EOG
27.	Initial	59.	KG = 0 & EOG
*28.	CC = 0	60.	KG = 1 & EOG
*29.	$\overline{X}$	61.	LG = 0 & EOG
30.	No-op	62.	MG = 0 & EOG
31.	F ≠ LIMIT	*63.	EOG

*Stream Stimulus Status and Mask Register

TAG

00 - Do REACTION 1, REACTION 2, REACTION 3

01 - AND of this and next stimulus

10 - Do Reaction 1 and Reaction 2, and add the value of Reaction 3 to IC for next instruction.

11 - Do Reaction 1 and Reaction 2, and add the value of Reaction 3 to IC for next adjustment halfword.

ADJUSTMENT REACTIONS

Reaction #	Octal Coding	Reaction
1.	000	No-op
2.	020	Disable match units for runout
3.	050	Skip extraction for T address which will contain special byte.
4.	051	Cancel T address which will contain special byte.
5.	052	Substitute (TBA-1) for T address which will contain special byte.
6.	053	Reset T base address to TBA
7.	054	Add SCTR to TBA
8.	060	Insert W in L
9.	061	Insert X in L
10.	062	Insert Y in L
11.	063	Insert Z in L
12.	064	Insert MOD in L
13.	070	Swallow special-byte output of L
14.	073	Suppress L output for remainder of group which contained special byte.
15.	074	Process remainder of data in pipeline.
16.	075	Reference (TBA-1) and increment.
17.	100	Reset this level in P.
18.	101	Reset thru level FL_1 in P
19.	102	Reset thru level FL_2 in P.
20.	103	Reset through level FL_3 in P.
21.	104	Advance next level in P.
22.	105	Advance next level above FL_1 level in P.
23.	106	Advance next level above FL_2 level in P.
24.	107	Advance next level above FL_3 level in P.
25.	110	Repeat special byte from P.
26.	111	Runout this level in P to R.
27.	112	Runout thru level FL_1 in P to R.
28.	113	Runout thru level FL_2 in P to R.
29.	114	Swallow byte after special byte from P.
30.	115	Match-only this level in P.
31.	116	Match-only thru level FL_1 in P.
32.	117	Match-only thru level FL_2 in P.
33.	120	Reset this level in Q
34.	121	Reset thru level FL_1 in Q
35.	122	Reset thru level FL_2 in Q.
36.	123	Reset thru level FL_3 in Q.
37.	124	Advance next level in Q.
38.	125	Advance next level above FL_1 level in Q.
39.	126	Advance next level above FL_2 level in Q.
40.	127	Advance next level above FL_3 level in Q.
41.	130	Repeat special Byte from Q.
42.	131	Runout this level in Q to R.
43.	132	Runout thru level FL_1 in Q to R.
44.	133	Runout thru level FL_2 in Q to R.
45.	134	Swallow Byte after special byte from Q.
46.	135	Match-only this level in Q.

47.	136	Match-only thru level FL_1 in Q.
48.	137	Match-only thru level FL_2 in Q.
49.	140	Reset this level in R.
50.	141	Reset thru level FL_1 in R.
51.	142	Reset thru level FL_2 in R.
52.	143	Reset thru level FL_3 in R.
53.	144	Advance next level in R.
54.	145	Advance next level above level FL_1 in R.
55.	146	Advance next level above level FL_2 in R.
56.	147	Advance next level above level FL_3 in R.
57.	150	Swallow special byte into R.
58.	154	Skip space in R before reading in special byte.
59.	160	Reset U to beginning of reference which contained special byte.
60.	162	Insert MOD for U.
61.	164	Skip remaining U extraction of reference which contained special byte.
62.	170	Repeat special byte from U.
63.	171	Runout to R the remainder of U reference which contained special byte.
64.	172	Runout to SACC the remainder of U reference which contained special byte.
65.	173	Runout to R the remainder of U group which contained special byte.
66.	174	Swallow byte after special byte from U.
67.	175	Match-only the remainder of U reference which contained special byte.
68.	176	Match-only the remainder of U group which contained special byte.
69.	204	Reset SACC
70.	210	Readout low-order 8 bits of SACC to R.
71.	211	Readout low-order 16 bits of SACC to R.
72.	212	Readout 24 bits of SACC to R.
73.	220	Step SACC by plus 1.
74.	242	Reset SCTR
75.	244	Readout low-order 8 bits of SCTR to R.
76.	245	Readout 16 bits of SCTR to R.
77.	246	Readout low-order 8 bits of SCTR to R and reset SCTR.
78.	247	Readout 16 bits of SCTR to R and reset SCTR.
79.	260	Step SCTR by minus 1.
80.	250	Step SCTR by plus 1.
81.	3(oxx)X	Disable specified stimuli for this byte.
82.	3(1xx)X	Disable specified stimuli for duration of group.

1.4 Adjustment System

The Harvest microinstruction system has been drawn together into a more centralized unit and a more logically consistent set of operations. This has made it possible to implement and at the same time time more flexible and easier to program. It recognizes that except for special combinations the execution of adjustments is necessarily serial.

The system now has three distinct sections. The first consists of setup operations which include match unit swallows and insertions and SCTR stepping. These have the characterists that the stream need not be stopped and there is no problem of priority of execution. These operations always preceed an adjustment initiated by the same stimulus or one arising on the same cycle. The second section consists of the five general microinstructions which are carried with the macroinstruction. The third section consists of the stream stimulus mask which provides facility for detecting the occurrence of any number of streaming stimuli. The function of this section is to stop the stream in a position which will permit adjustment execution and provide a link to another instruction which will command the appropriate adjustment.

There has been no basic change in the setup uperations. These have, however, been limited to operations requiring high speed because of frequent occurrence and requiring no priority because of independence of execution. The five microinstructions have the same stimulus specification permitting one of sixty stimuli to activate the operation. The coding of the operation fields has been modified to yield three subfields each capable of specifying any desired reaction. Generality is increased by insluding a two bit tag which governs the execution of the microinstruction. If the tag is zero three reactions are specified in the three subfields. If the tag is one the operation is performed only if the next (lower priority) stimulus is also present. This provides the ability to react to any combination of two stimuli and many combinations of three or four. If the tag is two the first two subfields specify internal reactions and the third is used to increment the instruction counter to yield the address of the next arithmetic mode instruction which is immediately fetched. This is an alternative implementation of the old Skam option.

The option provided by a tag of three is a new feature in the machine, but not really a new proposal. This option provides the ability to extend indefinitely the number of reactions that can result from a given stimulus. This is accomplished by again using the third subfield to specify an instruction counter increment. In this case, however, the half word which is fetched is considered as a microinstruction rather than an arithmetic instruction. It is possible to have any length chain of these adjustments in memory. If the Initial stimulus is used on a chained adjustment the effect is to merely

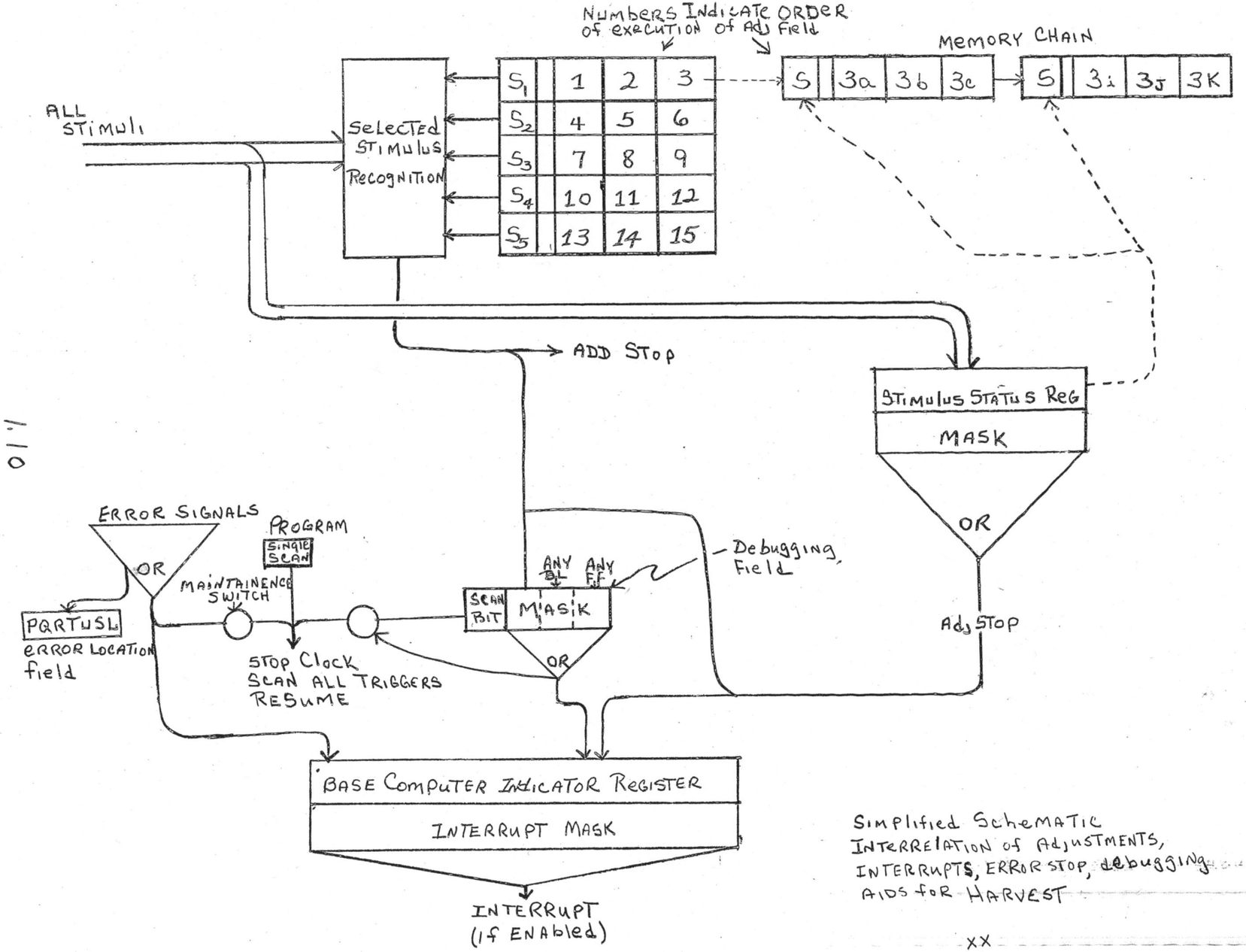
increase the number of reactions resulting from the prime stimulus. If another stimulus is specified the result is either to yield an Anding of prime and chained stimuli to produce the reactions in the chained instruction, or to set up a sequence of tests in which stimulus A produces reaction X which may produce stimulus B which will then cause reaction Y and so on.

In order to have the state of streaming stimuli available for testing when a chained adjustment is executed it is necessary to provide a stimulus status register. This also is essential for operation in conjunction with the third section of the adjustment mechanism which is the stream stimulus mask. As discussed above and illustrated in the schematic drawing this mask stops the stream on any number of stimuli in addition to those specified in the five microinstructions. When a masked on stimulus occurs an interrupt is taken if it is masked on and enabled or the instructions is terminated at I. C. + 4 if either condition is not satisfied. Any of the five adjustments that may have been simultaneously stimulated are executed before ending the instruction.

The full power of the arithmetic mode can be used when the stream is stopped by means of the stimulus mask. Since the stimulus status triggers are addressable they may be operated upon by connective or bit branching instructions. It is possible to also test the status triggers in streaming mode. To do this the macroinstruction is repeated with an End of Adjustment stop code and the required set of five microinstructions. The advantages of this method are that the desired reactions can be directly specified and up to five stimuli can be tested with one instruction.

During the execution of the five directly specified adjustments it is possible to generate additional stimuli. If these match the stimulus specifications they will command the corresponding adjustments. It is thus possible to execute adjustment two and then return and execute adjustment one rather than three if two stimulates one. Under rare combinations of adjustments it is possible to generate the same stimulus twice on the same byte-adjustment cycle. If this should occur the Lost Stimulus indicator will be turned on.

The priority of adjustment execution may be summarized in the following manner. Any setup operations are performed prior to the adjustment stop. The leftmost of the five adjustment fields having an on stimulus is executed first. The execution of that operation includes all subfields and chained instructions. Completion of the operation turns off the command trigger for that field. At that time the leftmost of the five adjustment fields having an on stimulus is executed. This process continues until none of the five fields has an on stimulus. Interruption or transfer to I. C. + 4 due to stimuli that are on and masked occurs at this time. It should be emphasized that when a stimulus appears in one of the five microinstructions it is not necessary to also have on the corresponding bit in the stimulus mask unless both interruption and adjustment is desired.



Simplified Schematic
Interrelation of Adjustments,
Interrupts, Error Stop, debugging
Aids for Harvest

XX

1.5 Adjustment Checking

In order for an adjustment stimulus to be compatible with an adjustment reaction, the pair must yield identical results when repeated, i.e. the adjusting operation must be independent of any random effects due to bus priorities or busy memories. For any gating configuration each of the adjustment stimuli will be incompatible with certain of the adjustment reactions. If a programmer should blunder and attempt to use a pair which is incompatible with his data-gating configuration, the resulting output would be unpredictible.

The situation is complicated by the fact that one streaming instruction may switch between different data-gating configurations (e.g. table-lookup and adjustment runout), and a particular stimulus-reaction combination may be valid for one gating pattern and not for another. Suppose a programmer has advance knowledge that a certain match should occur only during a runout. He might wish to use the match and its reaction in a table-lookup instruction even though the pair is incompatible with the table-lookup process. This would be a completely proper operation. If, by some chance, however, the match should subsequently occur unexpectedly during a table-lookup portion of the instruction, the machine would operate unpredictable. Unlike the simple blunder which, theoretically at least, could always be spotted by checking the program in advance, there is no way the latter situation could be detected.

It has been proposed that the machine include a validity checker to warn the programmer if the machine is being asked to use a stimulus-reaction combination which is illegal with the currently specified data-gating pattern. The attached drawing shows the proposed logical layout of this checker. It checks almost all stimuli and reactions except those which involve SCTR.

The maximum number of asynchronous streams which can be running simultaneously in HARVEST is three. The first two types are an "upstream" stream which contains T and a "downstream"stream which contains U. They are asynchronous because of the indeterminite length of the memory link between T and U. The third type is an "independent" stream which is not connected with either T or U. An example of one instruction's containing all three streams would be the gating pattern where P feeds L which feeds SACC, stream, the Q-T stream would be an upstream stream, and the U-R Stream would be a downstream stream.

The checking scheme is based on the fact that a unit which activates a stimulus must be synchronous with the unit which the corresponding reaction adjusts.

Since all the general stimuli [1] now occur in one place (i.e. time-zone #2) it is sufficient to demand that both units be in the same stream. Moreover, since in any one instruction there can never be more than one stream of each type, the validity requirement may be modified to demand that both units be in the same type of stream. This latter condition is what the checker checks.

With reference to the diagram, the block labeled "unit-status determiner" associates each unit in the machine with the type of stream in which it is being used. For each unit, i.e. P, Q, U, L, R, T, and SACC, there is a set of output lines. For most units the set contains 4 mutually-exclusive lines representing the following 4 situations:

1. The unit is part of an upstream stream
2. The unit is part of a downstream stream.
3. The unit is part of an independent stream.
4. The unit is ungated.

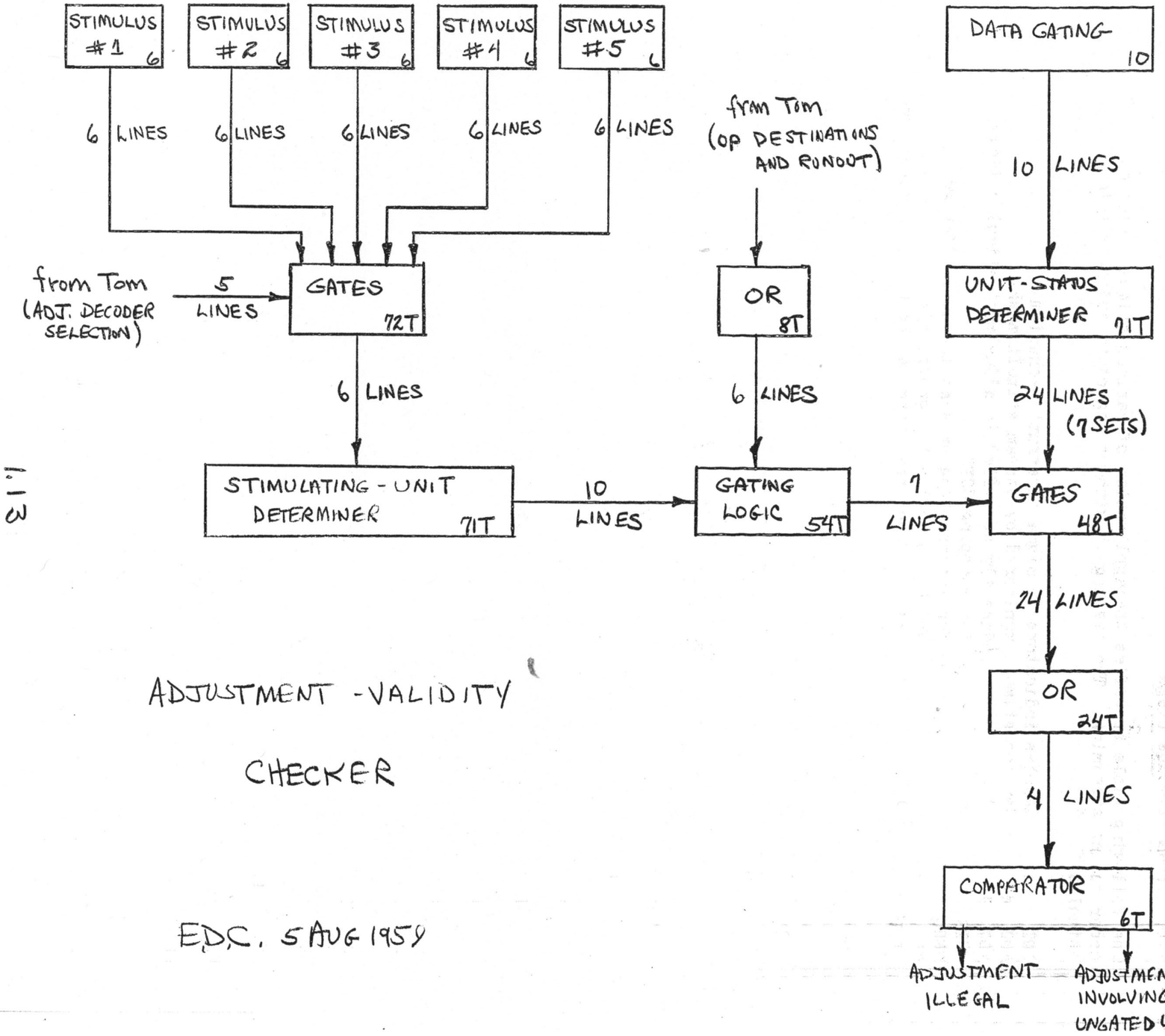
These 7 sets of lines are fed to a group of gates which are controlled by the 7 outputs from the block labeled "gating logic."

The stimulus field for the half-word currently being decoded is fed through some circuitry which determines the stimulating-unit for the half word. At the same time an ORing of the decoded "reaction-op-bit" destinations indicates which unit is to be adjusted. These are ORed in the block labeled "gating logic" and yield thepreviously-mentioned 7 outputs. Each line corresponds to one of the 7 units. It is on if its unit is participating in any way in the current reaction (i.e. either directly or indirectly as either the stimulus or the reaction) and it is off if its unit is not involved.

The result is that for any unit involved in the current reaction the unit's entire corresponding set of lines is gated from the "unit-status determiner" to the OR circuits. Each of the 4 OR circuits combines all the lines pertaining to a particular type of stream. The output of the "downstream" OR circuit, for example, will be on if any of the units involved in the current reaction was part of a downstream stream.

The final step is to compare the "upstream", "downstream", and "independent" lines and make sure that no more than one of them is on. If more than one is ever on, the illegal adjustment indicator is set. Because a programmer might have valid reasons for using an ungated unit, however, the "ungated" line does not go the "illegal adjustment" indicator but to another indicator which simply says that an ungated unit was involved in an adjusting operation.

1. Stimuli involving SACC or Higher levels in R are no longer considered general since they are restricted to use with "Go to I.C." adjustments.



ADJUSTMENT - VALIDITY

CHECKER

E.D.C. 5 AUG 1959

1.6 Stream Stimulus Usage

The following table specifies the application of various stimuli which may occur during streaming. The listing is without regard to coding within the stimulus fields.

INT	denotes	indicators in Sigma interrupt machanism
ADJ. M	denotes	stimuli monitored by stream stimuli mask
ADJ. S	"	" directly specified in adjustment stimulus field
SCTR	"	" for stepping SCTR
DEBUG	"	" for interrupting or scanning during debugging
GS	"	" which can define the group size
STOP	"	" which defines the termination of the instruction

	STG. Reg	INT	Adj. M.	Adj S.	SCTR	Debug	GS	STOP
0 Noop				X	X	X	X	X
1 Initial				X				
2 EOL_1P							X	
3 F_1P			X	X			X	
4 F_2P			X	X				X
5 F_3P			X	X				X
6 BLP						X		
7 FFP								
8 CCP		X						
9 EOL1Q							X	
10 F1Q			X	X			X	
11 F2Q			X	X				X
12 F3Q			X	X				X
13 BLQ						X		
14 FFQ								

		Stg Reg	INT	Adj M	Adj S	SCTR	Debug	GS	STOP
15	CCQ		x						
16	EOL1R								
17	F1R			X	X				
18	F2R			X	X				X
19	F3R			X	X				X
20	BLR						X		
21	FFR								
22	CCR		X						
23	EOLU			X	X				
24	CCU								
25	EOG			X	X				
26	T ADR. Formed					X			
27	OP in LU					X			
28	BYTE from P					X			
29	BYTE from Q					X			
30	BYTE to R					X			
31	+ BYTE to SACC					X			
32	- BYTE to SACC					X			
33	SACC TH			X	X				
34	SACC goes minus			X	X				
35	SACC TH & EOG				X				
36	SACC TH & Byte in					X			
37	SCTR step			X	X				

		STG Reg	INT	Adj M	Adj S	SCTR	Debug	GS	STOP
38	SCTR = lim			X	X				
39	SCTR ≠ lim & EOG				X				
40	W			X	X	X		X	
41	X			X	X	X			
42	Y			X	X	X			
43	Z			X	X	X		X	
44	X or Y				X				
45	W & X				X				
46	W & Y				X				
47	$\overline{W}$			X	X	X			
48	$\overline{X}$			X	X	X			
49	$\overline{Y}$			X	X	X			
50	$\overline{Z}$			X	X	X			
51	$\overline{W}$ & $\overline{Y}$					X			
52	$\overline{W}$ & $\overline{X}$ & $\overline{Y}$ & $\overline{Z}$					X			
53	F = lim			X	X				
54	F ≠ lim				X				
55	F ≠ lim & EOG				X				
56	KB = 1			X	X				
57	LB = 1			X	X				
58	MB = 1			X	X				
59	KB = 0				X				

		Stg Reg	INT	Adj M	Adj S	SCTR	Debug	GS	STOP
60	LB = 0				X				
61	MB = 0				X				
62	KB = 0 & F = lim				X				
63	KB = 1 & F = lim				X				
64	KB = 1 & F ≠ lim				X				
65	KB = 0 & F ≠ lim					X			
66	LB = 0 & F = lim				X				
67	LB = 1 & F = lim				X				
68	LB = 1 & F ≠ lim				X				
69	LB = 0 & F ≠ lim					X			
70	MB = 0 & F = lim				X				
71	MB = 1 & F = lim				X				
72	MB = 1 & F ≠ lim				X				
73	MB = 0 & F ≠ lim					X			
74	KG = 1			X	X				
75	KG = 0				X				
76	LG = 1			X	X				
77	LG = 0				X				
78	MG = 1			X	X				
79	MG = 0				X				
80	KG = 1 & EOG				X				

		Sta Reg	INT	Adj M	Adj S	SCTR	Debug	GS	STOP
81	KG = 0 & EOG				X				
82	LG = 1 & EOG				X				
83	LG = 0 & EOG				X				
84	MG = 1 & EOG				X				
85	MG = 0 & EOG				X				
86	Memory count overflow		X						
87	Sacc overflow/underflow		X						
88	lost stimulus		X						
89	SCTR overflow/underflow		X						
90	extract wraparound		X						
91	op invalid		X						
92	arith. error		X						
93	adj 1								
94	adj 2								
95	adj 3								
96	adj 4								
97	adj 5								
98	adj stim recognized		X						
99	any adj						X		
100	any cc = 0				X	X			
101	any index flag						X		
102	DEBUG		X						
103	P error	X							

		STG Reg	INT	Adj M	Adj S	SCTR	Debug	GS	STOP
104	Q error	X							
105	R error	X							
106	S error	X							
107	T error	X							
108	U error	X							
109	L error	X							
110	any byte into SACC					X			
111	end of sequence								X
112	STOP								
113	EOL2P								
114	EOL2Q								
115	EOL2R								
116	Lookup in wrong memory		X						
117	ungated unit adjusted								

1.7 MERGE

DA (1) DATA - ADDRESS

0- MOVE DATA
1- STORE ADDRESS

IE (1) INTERNAL-EXTERNAL

0- INTERNAL
1- EXTERNAL

UD (1) UP DOWN ORDERING

0- UP
1- DOWN

TYPE (1) TYPE

0- SIMPLE
1- OFFSET

1.8 SSER STREAM SEARCH

SCD (3) SEARCH CONDITION (MOVE RECORD IF)

000 - INVALID SCD
001 - $P < Q$
010 - $P = Q$
011 - $P \leq Q$
100 - $P > Q$
101 - $P \neq Q$
110 - $P \geq Q$
111 - INVALID SCD

OR(1) ORDERED RANDOM

0 - ORDERED
1 - RANDOM

1.9 SSEL STREAM SELECT

LG(1) LEAST GREATEST

0 - LEAST
1 - GREATEST

1.10 STIR Stream Takeout, Insert, or Replace

ID(1) Instruction data

0 - instruction control
1 - data control

RT(1) Replace/takeout (when ID = 0)

0 - replace
1 - takeout

1.11 SMLU Stream Multiple Lookup

Mode(1)

0 - Address from P, parameters from Q, arguments from R, results to R.
1 - Arguments from P, parameters from Q, result to R

1.12 SQNL Stream Sequential Lookup

P(1) Gate #4 P T

0 - closed
1 - opened

Q(1) Gate #5 Q T

0 - closed
1 - opened

1.13 SILS Stream Indirect Load or Store

LS(1) Load Store

0 - load
1 - store

1.14 ADJUSTMENT REACTION DEFINITIONS

1. No-op

No-op is just what is implied and the bits representing a No-op are set to zero.

2. Disable match units for runout

The machine immediately disconnects the match units from

the byte-bus and leaves them disconnected until the next termination of a runout.

If no runout is already in process or is signalled by the time the Primary Stimuli Register is empty, the adjustment is considered in error; the machine performs the adjustment as specified and disconnects the match units from the regular stream until such time as some runout, subsequently started, has terminated.

3. Skip extraction of reference now forming in T

Details of the timing of the execution of this adjustment have not yet been specified. In order to do any count or existence operations desired the completed address must be sent to memory, but the bit portion of this address will be marked before being sent to the bit-address bank which links T and U. When U receives the marked bit-address it will automatically execute "Swallow output of U for duration of reference".

This adjustment may be signalled in any time-zone as long as it is signalled from a stream that is synchronous with T.

4. Cancel T

Bytes will flow into T until all those pertaining to the current reference have entered, but T will not send any address to memory; all functions to be performed on the address will thus be cancelled.

This adjustment will be restricted to use with stimuli arising in time-zone #2 in the stream feeding T.

5. Substitute (TBA-1) for T Address

The method by which this adjustment is executed depends on the gating pattern specified by the streaming instruction. If the machine is doing table-lookups as part of the primary streaming process, the adjustment execution, which merely substitutes one reference address for another, uses only the T unit. If the machine is not doing table-lookup as part of the primary streaming process but is performing this adjustment table-lookup to obtain an insertion for the L stream, the adjustment execution also uses the byte-bus. Details of the timing of the execution have not yet been specified, but the probable sequence of operations

for each case is known. If the TRU is part of the primary stream, the machine performs a "Reset T". Then, as soon as the T-adder is available, the machine gates the TBA register as an addend and a -1 as an augend into the T-adder. One cycle later it gates the T-adder output of (TBA-1) into one of the T registers. The T register then sends this address to memory which subsequently treats it as if it had been a regular reference address. If the TRU is not part of the primary stream the machine immediately gates an addend of TBA and an augend of -1 into the T-adder and one cycle later gates the T-adder output of (TBA-1) into one of the T registers. The T register then sends this address to memory. When U receives the referenced word from memory, the machine then performs one or more "insert in L" type of adjustment operation using the specified bits of U as the source of the insert. It should be noted that in both cases the execution of this adjustment is independent of, but does not interfere with, the operation of the RPL bit.

It is almost certain that this operation will be restricted to use with stimuli arising in time-zone #2. The stimulus will have to be synchronous with T if the machine is doing table-lookups as part of the primary streaming process, and synchronous with L if the machine is not doing table-lookups as part of the primary streaming process.

6. Reset T base address to TBA

Details of the execution of this adjustment have not yet been specified. The probable method will simply be to gate the contents of the TBA register into the T-adder with an augend of 0 and then, on the following clock pulse, gate the T-adder output of TBA into the T register which holds the current table base-address. It should be noted that this adjustment utilizes the current contents of the TBA register; if the TBA register has previously been adjusted via "Add SCTR to TBA" the machine does not reset to the original table base-address, but to the sum of that base-address and SCTR.

When this adjustment is used to reset a cumulative table base-address that has been formed during a table-lookup stream in the RPL mode, it is almost certain that this adjustment will be restricted to use with stimuli originating in time-zone #2 in the stream feeding T. If this adjustment is being used to reset the base-address formed by successive use of "Reference (TBA-1) and increment," the restrictions will probably be less severe. Their exact nature, however, cannot yet be specified.

7. Add SCTR to TBA

This adjustment has not yet been completely defined as the positioning of the 16 bits from SCTR in the 24-bit TBA register remains undecided.* Although the timing has not been worked out the probable sequence of operations will begin with a two-cycle transfer of the 16 SCTR bits to a latch in the match-unit output (used for "Insert" adjustments). Overlapping this will be a two-cycle transfer from the match-unit latch to one of the T registers; the whole SCTR-to-T transfer should take 3 cycles for the 16 bits. As they enter T, one byte at a time, these SCTR bits will be added to TBA by the T adder as if they were regular data. Finally, the entire T-register contents of (TBA + SCTR) will be transferred in parallel over the internal bus to the TBA register.

Since the timing and control of this adjustment have not been worked out, it is not possible to say at present what restrictions, if any, will apply to this adjustment.

8-12. Insert W, X, Y, Z and MOD in L. Output.

The machine pulses all data gates and control gates between time-zone #3 and time-zone #4. Simultaneously with the above gating the machine gates the contents of the match-unit into all the intermediate-storage register of time-zone #3 which are being fed by L, and gates a zero into the tracer-bit positions of all the intermediate-storage registers of time-zone #3 which are not being fed by L. The byte-bus is then ready for narmal advancing.

This adjustment should be signalled from time-zone #2. It will work when signalled from any time-zone, but the contents of match-unit W will always be inserted between the bytes which are in time-zone #1 and time-zone #2 when the stimulus arises.

13. Swallow special - byte output of L.

The machine gates a zero into the tracer-bit position of all the intermediate-storage registers of time-zone #3 which are being fed by L. The byte-bus is then ready for normal advancing.

This adjustment should be signalled from time-zone #2. It will work when signalled from any time-zone, but the byte swallowed is the byte which is in time-zone #2 at the time the stimulus arises.

14. Suppress L Output for remainder of Group which contained special byte.

No definition

15. Process remainder of data in Pipe line.

No definition

16. Reference (TBA-1) and increment

This adjustment should only be used when the machine is not doing table-lookups as part of the primary streaming process. Details of the timing of the execution of this adjustment have not yet been specified but the probable sequence of operations is known. The first time this adjustment is performed, the procedure is more complicated than on subsequent executions. The machine gates to the T-adder the contents of the TBA register as an addend and -1 as an augend. Then, on the following clock pulse, the machine gates the T-adder output of (TBA-1) into one of the T registers. This T register sends the reference address of (TBA-1) to memory, immediately after which the machine gates the same T register output as an addend to the T-adder. This is added to an augend of $\pm$ 1 and the results of (T $\pm$ 1) are gated back into T. This incrementing by $\pm$ 1 is completed while the original reference from T -- in this case (TBA-1) -- is still in memory. When U receives the referenced word from memory, the machine then performs one or more "Insert in L" type of adjustment operation using the specified bits of U as the source of the insert. After this adjustment has been executed once, the T portion of subsequent executions becomes much simpler. Without any preliminary operations the machine immediately sends the contents of the appropriate T register to memory, and then does the incrementing by $\pm$ 1 in the manner described above. The execution of this adjustment is independent of the RPL bit.

If the machine is doing table-lookups as part of the primary streaming process when this adjustment is signalled, the adjustment is considered in error; the machine performs "Reference (TBA-1)".

It is almost certain that this adjustment will be restricted to use with stimuli arising in time-zone #2 in a stream synchronous with L.

17,33,49. Reset this level in P, Q, and R

The machine immediately forces the EOL routine for this level (which must be a reading level), setting M to 0 and, if the SR bit = 0, putting the contents of J into S. Readout of this level is then restarted. The EOL signal for this level is not activated and except for the SR bit any other control bits present in this level are ignored.

This adjustment must be signalled from time-zone #2. If it is signalled from time-zone #4 and one of the last two bytes in this level signals the adjustment, the results may be drastically abnormal.

18,19,20, Reset thru level FL_1, FL_2, FL_3, in P, Q, and R.
34,35,36,
50,51,52, Force EOL routine for this level, Next level---- FL_1, FL_2, FL_3; set M to zero, put J in S if SR = $\emptyset$ and FS = 1, and restart on level 1 or FL_1, FL_2, FL_3 in accordance with level FL_1, FL_2, FL_3; NS bit. The presence of an FF or BL bit in a level below FL_1, FL_2, FL_3 terminates the operation. The FF level is reset and indexing returns to the Start of that level or the first level depending on the NS bit. If a BL level is encountered first the operation terminates with the resetting of the BL_1 level and the indexing continues in the first level of that nest. No end of level signals are provided. A FL_1, FL_2, or FL_3 bit in this level is ignored. A CC bit = $\emptyset$ on a level below FL_1, FL_2, or FL_3 terminates the operation.

21,37,53. Advance next level in P, Q, and R.

The machine immediately resets this level (which must be a reading level). It activates an EOL signal for this level and advances, in accordance with the control bits on this level, to the next level. If the next level is also a reading level (sequential or virtual first) the machine branches normally and continues. If the next level is neither of the above, but a nested level, the machine

increments the next level, returns to the first level, and restarts readout of the first level.

This adjustment must be signalled from time-zone #2. If it is signalled from time-zone #4 and one of the last two bytes in this level signals the adjustment, the results may be drastically abnormal.

22,23,24, 38,39,40, 54,55,56. Advance next level above tagged level in P, Q, and R. The machine immediately resets this level (which must be a reading level). Then, with the location and nature of each new level being determined in accordance with the normal interpretation of all pertinent control data, the machine continues resetting levels until it has reset a level with an HL bit and advanced to the next level above the level with the FL bit. If the next level above the level with the FL bit is a reading level (sequential or virtual first) the machine starts readout of that level. If the next level above the level with the FL bit is a branch level the machine branches normally and continues. If the next level above the level with an HL bit is a nested level, the machine increments that level, returns to the first level, and restarts readout of this level. In all cases EOL signals are activated during the EOL's of all levels up to and including the level with the HL bit.

This adjustment should be signalled from time-zone #2. If it is signalled from time-zone #4 and one of the last two bytes in this level signals the adjustment, the results may be drastically abnormal. It is possible in certain special cases for any byte to signal this adjustment from time-zone #4, but the restrictions are very complicated and beyond the scope of this report.

25. Repeat Special byte from (P_j).

With the special byte frozen in time-zone #3, the gates are pulsed in the loop formed by IS_{LP} and LBS_P and the two registers interchange their contents. Then suppressing P indexing and the gates into IS_{MP} and IS_{TP} and feeding IS_{LP} from LBS_P instead of P, the byte-bus advances one cycle. Normal indexing and gating are then resumed. If P is jointly feeding a downstream unit with Q or U, Q indexing and the Q byte stream or U indexing and the U byte stream must advance one cycle to furnish a new byte as the byte-bus advances. The repeated byte is not included in the count of bytes leaving P.

This adjustment should usually be signalled from time-zone #2. It will work for any stimulus, but the byte which is repeated is the byte which is in time-zone #2 when the stimulus arises.

26,42. Runout this level in (P, Q) to R.

Data flow from other sources is halted and the data from (P,Q) is streamed, regardless of the gating pattern specified on the instruction, through L and into R. The data is unmodified by L and is monitored only by match units connected to (MS_P, MS_Q) or MS_L. The process terminates as soon as the EOL signal for this level in (P,Q) (which must be a reading level) is activated, and all EOL control data for this level is subsequently used in the normal fashion. During the streaming from (P,Q), adjustments are treated in the same manner as if they had arisen in any regular stream.

This adjustment must be signalled from time-zone #2. If it is signalled from time-zone #4 and one of the last two bytes in this level signals the asjustment, the results may be drastically abnormal.

27,28, 43,44. Runout thru FL_1 and FL_2 in (P,Q) to R.

The data from the specified source passes through the logical unit into R until the end of level marked FL_1 or FL_2 occurs. Action on an earlier FF or BL causes the process to stop at the end of the BL level.

29,45. Swallow Byte after Special Byte from (P,Q).

No definition

30,46. Match-Only this level in (P,Q).

Data flow from other sources is halted and the data from (P,Q) is streamed only to the match units connected to (MS_P,MS_Q). The process terminates as soon as the EOL signal for this level in (P,Q)(which must be a reading level) is activated, and all EOL control data for this level is subsequently used in the normal fashion. During

the streaming from (P,Q), adjustments are treated in the same manner as if they had arisen in any regular stream.

This adjustment must be signalled from time-zone #2. If it is signalled from time-zone #4 and one of the last two bytes in this level signals the adjustment, the results may be drastically abnormal.

31,32, 47,48. Match-Only to end of tagged level in (P,Q).

Data flow from other sources is halted and, with the location and nature of each new level being determined in accordance with the normal interpretation of all control data, the data from (P,Q) is streamed only to the match units connected to MS_P, MS_Q. The process terminates with the activation of an EOL signal from a level in (P,Q), containing an FL bit, or both, or with the recognition of a level in (P,Q), containing a BL bit. In the former case the process is considered terminated as soon as the EOL signal is activated and all EOL controls for that level are subsequently used in the normal fashion. In the case of the branch level the process is considered to have terminated before the start of the branch level and the machine proceeds to execute the branch level in the usual manner. During the stream from P all EOL signals are activated and adjustments are treated in the same manner as if they had arisen during any regular stream.

This adjustment should be signalled from time-zone #2. If it is signalled from time-zone #4 and one of the last two bytes in this level signals the adjustment, the results may be drastically abnormal. It is possible in certain special cases for any byte to signal this adjustment from time-zone #4, but the restrictions are very complicated and beyond the scope of this report.

41. Repeat Special Byte from Q.

With the special byte frozen in time-zone #3, the gates are pulsed in the loop formed by IS_{LQ} and the two registers interchange their contents. Then, suppressing Q indexing and the gates into IS_{MQ} and IS_{TQ}, and feeding IS_{LQ} from LBS_Q instead of Q, the byte-bus advances one cycle. Normal indexing and gating are then resumed. If Q is jointly feeding a downstream unit with P, P indexing and the P byte stream must advance one cycle to furnish a new byte as the byte-bus advances. The repeated byte is not included in the count of bytes leaving Q.

This adjustment should usually be signalled from time-zone #2. It will work for any stimulus, but the byte which is repeated is the byte which is in time-zone #2 when the stimulus arises.

57. Swallow next byte into R.

With the special byte frozen in time-zone #3 the machine prepares to gate a zero into the tracer-bit position of IS_{RL}. Then, on the first subsequent clock pulse that advances the byte-bus, the machine gates a zero into the tracer-bit position of IS_{RL} as regular streaming is resumed. The swallowed byte is not included in the count of bytes entering R.

This adjustment may be signalled from any time-zone, but the byte swallowed is the byte that is in time-zone #1 when the stimulus arises.

58. Skip space in R before reading in Special byte

With the special byte frozen in time-zone #3, R indexing advances one cycle. The machine then resume normal streaming.

This adjustment may be signalled from any time-zone, but the space in R is inserted just before the byte that is in time-zone #2 when the stimulus arises.

59. Reset U to beginning of Reference which Contained Special Byte.

No definition

60. Insert MOD in U output

The machine pulses all data gates and control gates between time-zone #3 and time-zone #4. Simultaneously with the above gating the machine gates the contents of the MOD register into all the intermediate-storage registers of time-zone#3 which are being fed by U, and gates a zero into the tracer-bit positions of all the intermediate-storage registers of time-zone #3 which are not being fed by U. The byte-bit is then ready for normal advancing.

This U-insert adjustment differs from the previous L-insert adjustments in that the byte inserted in U is not as flexible as a regular byte from U. The inserted byte can only go to R, SACC, and T, and hence this adjustment cannot be used if U is feeding L. This adjustment should be signalled from time-zone #2. It will work when signalled from any time-zone, but the contents of the MOD register will always be inserted between the bytes which are in time-zone #1 and time-zone #2 when the stimulus arises.

61. Skip extraction of all references for duration of group

Details of the execution of this adjustment have not yet been defined. Since this adjustment will cut off all flow downstream from U, the EOG signal cannot originate downstream of U; it must come upstream of U (i.e. in the stream feeding T). Furthermore, since a time-zone #4 stimulus is too late to affect an outgoing address from T to memory, the EOG signal must come from time-zone #2 in the stream feeding T. Overlap problems between groups would then force this adjustment to be signalled from the same time-zone, and this restriction would make the adjustment almost useless from a logical viewpoint. Moreover, even if the restriction were accepted, problems involving the asynchrony between T and U would probably prevent meaningful control communication between the two streams. It has been proposed that all EOG's be marked, regardless of whether any adjustments are occurring, and that this tag be transmitted through the bit-address bank linking T to U. This approach may provide a solution, but it has not yet been sufficiently studied.

It is almost certain that there will be time-zone restrictions on the signalling of this adjustment but is has not yet been determined what they will be.

62. Repeat Special byte from U.

With the special byte frozen in time-zone #3, the gates are pulsed in the loop formed by "IS_{LQ}" and the two registers interchange their contents. Then, suppressing U indexing and the gates into "IS_{MU}" and any intermediate-storage registers in time-zone #3 which are being fed by IS_{MU} and feeding "IS_{LQ}" from "LBS_{Q}" instead of U, the byte-bus advances one cycle. Normal indexing and gating are then resumed. If U is jointly feeding a downstream unit with P, P indexing and the P byte stream must advance one cycle to furnish a new byte as the byte-bus advances. The repeated

byte is not included in the count of bytes leaving U.

This adjustment should usually be signalled from time-zone #2 downstream of U. It will work for any stimulus, but the byte which is repeated is the byte which is in time-zone #2 downstream of U when the stimulus arises.

63,64. Runout R and SACC, the remainder of W reference which contained special byte.

Data flow from other sources is halted and the data from U is streamed, regardless of the gating pattern specified in the instruction, directly into R alone.

The data is unmodified and is monitored only by match units connected to MS_P. During the streaming from U, adjustments are treated in the same manner as if they had arisen in a regular stream. The timing of the termination of this adjustment has not yet been specified; it will involve the byte bank controls and these have not yet been designed.

It is almost certain that this adjustment will have to be signalled from time-zone #2 downstream of U.

65. Runout to R the remainder of U group which contained special byte.

Details for the execution of this adjustment have not yet been specified. The actual runout proceeds in the usual fashion, but the timing of the control, as especially the termination, is still undetermined.

There will probably be some time-zone restrictions on this adjustment, but they have not yet been determined.

66. Swallow byte after special byte from U.

With the special byte frozen in time-zone #3, a zero is gated into the tracer-bit position of "IS_{LQ}" and "IS_{MU}". If U and P are jointly feeding a downstream unit, the downstream unit's demand for simultaneous inputs will force P indexing to suspend for one cycle while U fills the resultant hole in the P byte-stream.

67. Match-only the remainder of U reference which contained the Special Byte.

No definition.

68. Match-only the remainder of the U Group which contained special byte.

No definition

69. Reset SACC

On the first clock pulse after the adjustment information has been set into the SACC adjustment triggers, and with the special byte having been drained into SACC, the machine gates all zeros into the SACC-adder. Then, on the following clock pulse, the machine gates the all-zero output of the SACC-adder into SACC.

There are no restrictions on this adjustment by itself, but it is usually used in conjunction with the stepping of SACC or the reading out of SACC and these functions may indirectly impose restrictions on this adjustment.

70. Readout low order 8 bits of SACC to R

Details concerning the execution of this adjustment have not yet been completely specified.

71. Readout low order 16 bits of SACC to R

Details concerning the execution of this adjustment have not yet been completely specified.

72. Readout 24 bits of SACC to R

Details concerning the execution of this adjustment have not yet been completely specified.

73. Step SACC by $\pm$ 1

The machine gates a "hot one" into the SACC adder (located in time-zone #3). Then, on the following clock pulse the machine gates the SACC-adder output of (SACC $\pm$ 1) into SACC.

This adjustment should not be used if SACC is accumulating signed bytes since the machine needs the SACC-adder's "hot one" input for use in the subtraction process. If the machine is asked to do both operations the adjustment is considered in error; the machine performs only the subtraction.

There are no restrictions on this adjustment by itself, but it is usually used in conjunction with the reading out of SACC or the resetting of SACC and these functions may indirectly impose restrictions on this adjustment.

74. Reset SCTR

On the first clock pulse after the adjustment information has been set into the SCTR adjustment triggers, the machine gates all zeros into the SCTR-adder. Then, on the following clock pulse, the machine gates the all-zero output of the SCTR-adder into SCTR.

75. Readout low order 8 bits of SCTR to R

Details concerning the execution of this adjustment have not yet been completely specified.

76. Readout 16 bits of SCTR to R

Details concerning the execution of this adjustment have not yet been completely specified.

77. Readout low order 8 bits of SCTR to R and reset

The machine executes "SCTR readout low order 8 bits to R: and then executes "Reset SCTR".

Any restrictions on the two component adjustments will apply to this combination adjustment.

78. Readout 16 bits of SCTR to R and reset

The machine executes "SCTR readout 16 bits to R" and then executes "Reset SCTR".

Any restrictions on the two component adjustments will apply to this combination adjustment.

79. Step SCTR by -1

If the Step-SCTR setup field is set up to step SCTR in the minus direction, the machine immediately gates the SCTR-adder output of (SCTR -1) into SCTR. If the Step-SCTR setup field is set up to step SCTR in the plus direction, the machine immediately gates an augend of -1 into the SCTR adder. Then, on the following clock pulse, the machine gates the SCTR-adder output of (SCTR -1) into SCTR.

There are no restrictions on this adjustment by itself, but it is usually used in conjunction with the reading out of SCTR or the resetting of SCTR and these functions may indirectly impose restrictions on this adjustment.

80. Step SCTR by + 1

If the Step-SCTR setup field is set up to step SCTR in the plus direction, the machine immediately gates the SCTR-adder output of (SCTR + 1) into SCTR. If the Step-SCTR setup field is set up to step SCTR in the minus direction, the machine immediately gates an augend of + 1 into the SCTR adder. Then, on the following clock pulse, the machine gates the SCTR-adder output of (SCTR + 1) into SCTR.

There are no restrictions on this adjustment by itself, but it is usually used in conjunction with the reading out of SCTR or the resetting of SCTR and these functions may indirectly impose restrictions on this adjustment.

81. Disable specified stimuli for this byte

At the end of the stimulus-decoding cycle when the machine is preparing to reset the Primary-Stimuli-Register trigger it has just decoded, the machine also prepares to reset the Primary-Stimuli-Register triggers for any of the stimuli specified by this adjustment. Then, on the following clock pulse, the machine resets all these triggers.

There are no restrictions on the use of this adjustment.

82. Disable specified stimuli for duration of group

This adjustment has not yet been defined. The only expected use of this adjustment is to prevent the machine from hanging up on reset adjustments.

This adjustment must be signalled from the same time-zone as EOG. If the adjustment is signalled up stream from EOG, then if it is signalled by the first byte of a special group the forthcoming EOG signal for the previous group will immediately terminate the adjustment. The special group thus remains unadjusted. If the adjustment is signalled downstream from EOG, then if it is signalled by the last byte of a special group, the adjustment will not be terminated until receipt of the EOG signal for the group after the special group. The wrong group is thus adjusted. The only way around this dilemma is to demand that the stimulus originating this adjustment and the EOG signal arise in the same time-zone.

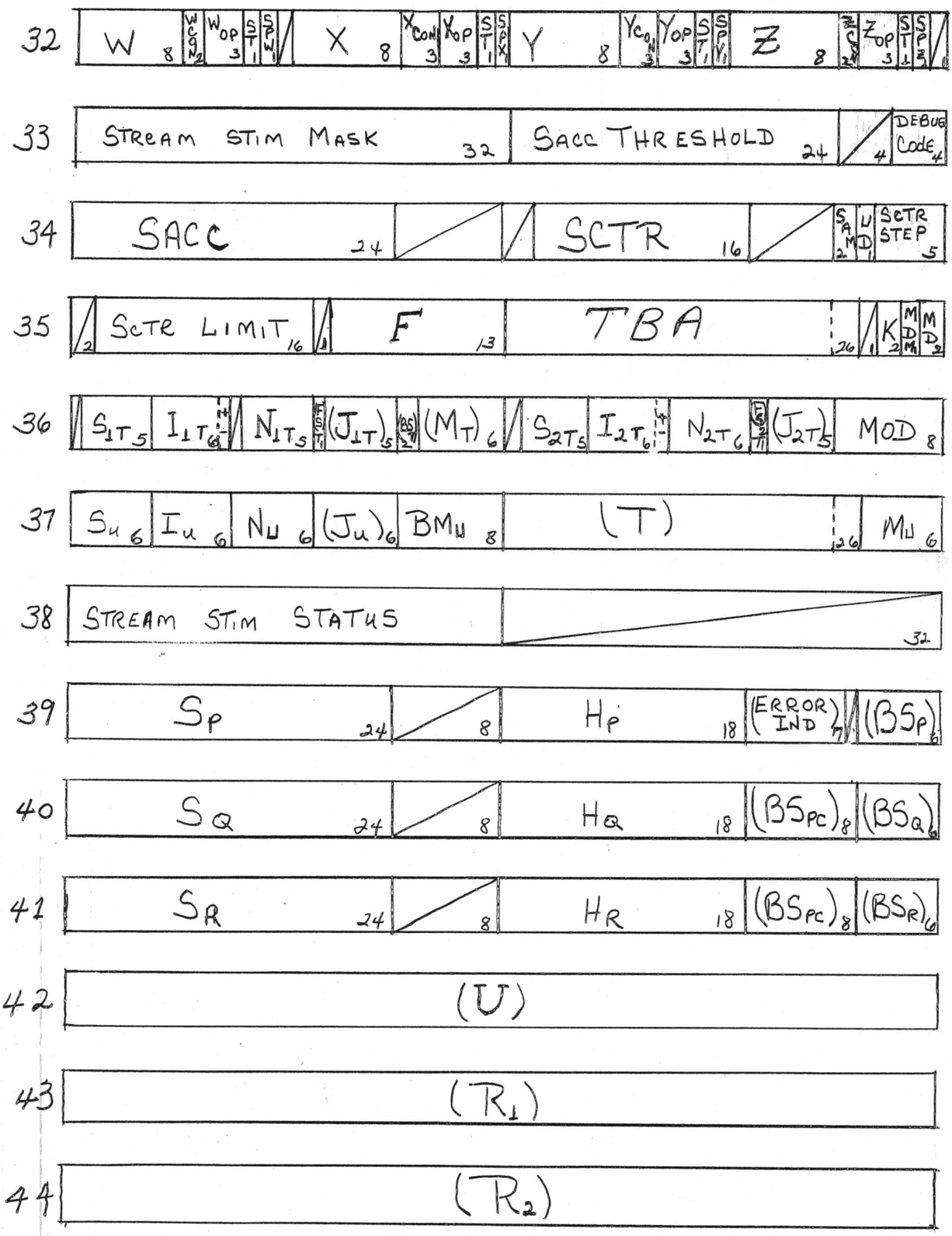
Notes on Execution of Level Marked BL:

The BL (Branch Level) level must always be a nesting (NS = 0) level. After each iteration of that level the BRHO - BRLO register provides the address of the first level of indexing for that unit. This address is used to fetch the appropriate first level parameters before the indexing continues. The end of level routine for the level follows this n-th iteration. All end of level operations occur normally at this time, including the execution of an FF operation if ordered. At the conclusion of the end of level operation the level BL ≠ 1 is fetched and executed as usual.

Chapter II

STREAM SET-UP

HARVEST SETUP WORDS

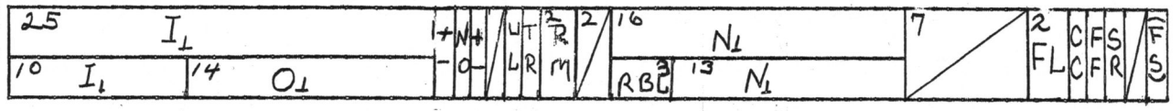


HARVEST SETUP WORDS

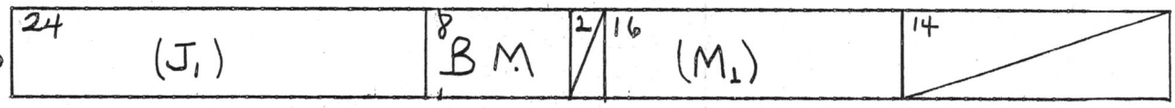
SETUP REGISTER ADDRESS

(P) (Q)

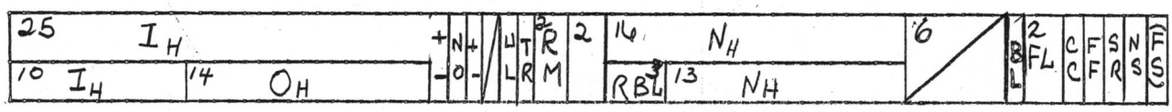
45 49



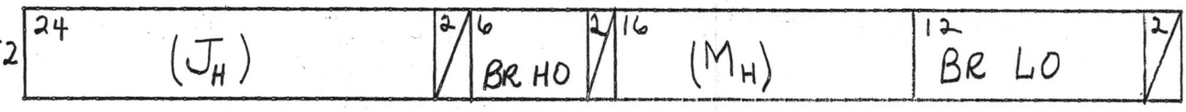
46 50



47 51

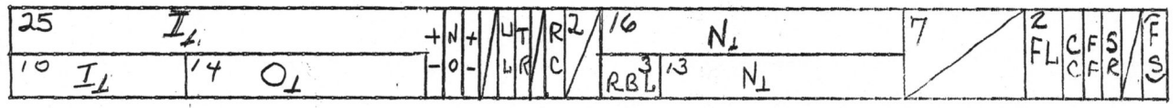


48 52

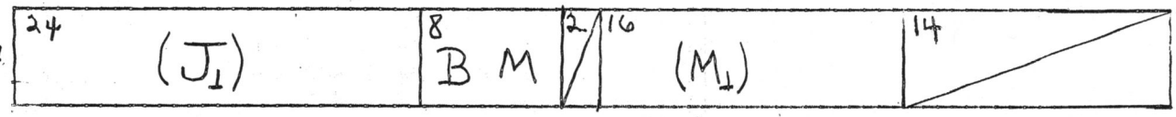


(R)

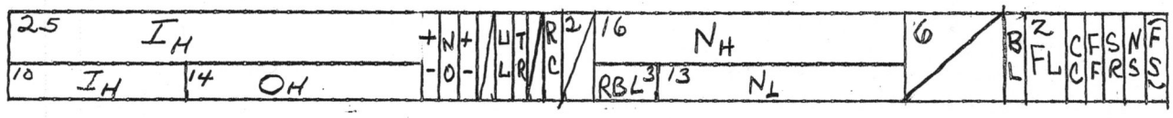
53



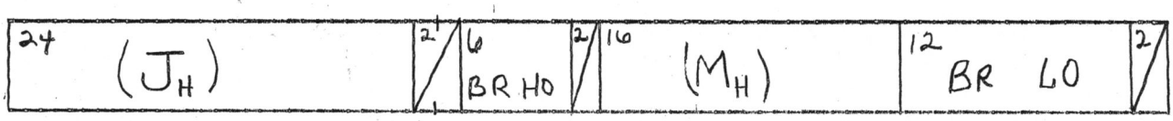
54



55



56



2.2 Match Units

W)
X)
Y) (8) Match Character
Z)

WCON(2) W connection

00 - no connection
01 - connected to P
10 - connected to Q
11 - connected to both P and Q

XCON(3) X connection

000 - no connection
001 - connected to L
010 - connected to Q
011 - connected to both Q and L
100 - connected to P
101 - connected to both P and L
110 - connected to both P and Q
111 - connected to P, Q and L

YCON(3) Y connection

000 - no connection
001 - connected to U
010 - connected to Q
011 - connected to both Q and U
100 - connected to P
101 - connected to both P and U
110 - connected to both P and Q
111 - connected to P, Q and U

ZCON(2) Z connection

00 - no connection
01 - connected to L
10 - connected to U
11 - connected to both L and U

OP Operation is to be performed independent from stimulus logical OR or AND

000 - No OP
001 - swallow both LU inputs and insert Mod
010 - swallow both LU inputs and insert match character
011 - swallow connected byte that meets operation criterion
100 - swallow all connected inputs
101 - swallow both LU inputs and insert match character and force MB to 1
110 - swallow both LU inputs and insert match character and force KB to 1
111 - No OP

ST(1) Definition of when stimulus is available for adjustments, counters and group size

0 - or
1 - and (only 2 out of 3 units on X and Y)
(L cannot be added with any other MS)

SPW(1)
SPX(1)
SPY(1) Span for W, X, Y and Z
SPZ(1)

0 - match on all eight bits
1 - match right most (low order) bit only

5/7/59

2.3 SCTR

SCTR LIMIT (16) value of SCTR limit

SCTR STEP (5)

0. NO OP
1. T Adr Formed
2. Op in LU
3. Byte from P
4. Byte from Q
5. Byte to R
6. + Byte to SACC
7. - Byte to SACC
8. SACC $\geq$ THR and Byte IN
9. W
10. X
11. Y
12. Z
13. $\overline{W}$
14. $\overline{X}$
15. $\overline{Y}$
16. $\overline{Z}$
17. $\overline{W}$ and $\overline{Y}$
18. $\overline{W}$ and $\overline{X}$ and $\overline{Y}$ and $\overline{Z}$
19. KB=0 and F$\neq$LIM
20. LB=0 " " "
21. MB=0 " " "
22. any byte into SACC

SCTR(16) Statistical Counter Value.

UD(1) up-down
0 - Sctr counts up (positive)
1 - Sctr Counts down (negative)

2.4 Statistical Accumulator

SACC THR(24) SACC Threshold Value

SAM(2) SACC Mode

00	Unsigned normal 1 byte entry
01	Unsigned 2 byte entry, high order byte first. Reset waits if only a high order byte has been accumulated.
10	Signed normal 1 byte entry. The sign of SACC is bit 23. Steps concurrent with negative byte accumulations are cancelled.
11	Signed reset nagative values. Same restrictions as 10.

SACC(24) SACC Value

2.5 Data Stream Units

Sp, P, R(24) Starting address for stream unit

H(18) Working Address

BS Boot Strap

Boot Strap Control triggers (not Set-up)

2.6 Logic Unit

F(13) F Unit

For bits KK thru MM

0 - No OP
1 - set to one if

Bit No.

0 - KK - K_0 = 1 and K = 1
1 - KL - K_0 = 1 and L = 1
2 - KM - K_0 = 1 and M = 1
3 - LK - L_0 = 1 and K = 1
4 - LL - L_0 = 1 and L = 1
5 - LM - L_0 = 1 and M = 1
6 - MK - M_0 = 1 and K = 1
7 - ML - M_0 = 1 and L = 1
8 - MM - M_0 = 1 and M = 1
9 - For bit IF invert F_0

0 - No OP
1 - invert F_0

10 - For bit SO stay on 1

0 - No OP
1 - stay on 1

11, 12 - For FL (2) field F limit

00 - limit of 4
01 - limit of 1
10 - limit of 2
11 - limit of 3

MOD(8) Modulus

00000000 - 2, 4, 8, 16, 32, 64, 128, 256
10000000 - 1, 2, 4, 8, 16, 32, 64, 128
11000000 - 3, 6, 12, 24, 48, 96, 192
10100000 - 5, 10, 20, 40, 80, 160
11111111 - 255

The pertinent bits of the modulus are specified by the corresponding leftmost "ones" of byte mask P and Q.

2.7 Table Reference Unit

TBA(26) Table Base Address

K(2) Cell Size

00 - no cell size
01 - 8 bits
10 - 16 bits
*11 - 24 bits

*24 bit cells start at full and half word addresses. The machine supplies counting address of bit positions 23 and 55.

$S_{1T}(5)$ - start address of first byte from P

$S_{2T}(5)$ - Start address of first byte from Q

$I_{1T}(6)$ - increment for S_{1T}

$I_{2T}(6)$ - increment for S_{2T}

$N_{1T}(5)$ - number of bytes from P

$N_{2T}(6)$ - number of bytes from Q

$J_{1T}(5)$ - reset address for P bytes (not normally setup)

$J_{2T}(5)$ - reset address for Q bytes (not normally setup)

$M_T(6)$ - count of bytes from P,Q (not normally setup)

$S_U(6)$ - Bit address from T (not normally setup)

$I_U(7)$ - increment for U

MDM(1) memory distributor mode

0 - No OP
1 - distribute

MD(2) memory distributor value

00 - zero
01 - 1
10 - 2
11 - 3

N_U(6) - number of bytes extracted from U

BM_U(8) - U byte mask

T(26) - T register (not normally setup)

M_U(6) - count of bytes being extracted (not normally setup)

U(64) - U register (not normally setup)

(FS_1) T (1)
(FS_2) T (1) First Subsequent (not Setup)

(BS) T (2) "Boot Strap" Control triggers (not setup)

2.8 Stream Unit "R" Registers

R_1(64) - R_1 register (not normally setup)

R_2(64) - R_2 register (not normally setup)

2.9 Stream Stimulus

1. Stream Stimulus Mask (32) Mask Register
   The mask register is in Register 33 and is setup by the programmer to correspond with the Stream Stimulus Register.

2. Stream Stimulus Status (32) Stimulus Register
   The Stream stimulus register is in register 38. This register (38) is fixed with Conditions as noted in the Adjustments.

3. Debug (4)

000(0) No OP on any Adjustment
000(1) Any adjustment

00(0)0 No OP on any BL_P, BL_Q or BL_R
00(1)0 Any BL_P, BL_Q or BL_R or No OP

0(0)00 No OP on any FL_1, FL_2 or FL_3
0(1)00 Any FL_1, FL_2, or FL_3

(0)000 No OP on SCAN
(1)000 Any SCAN

4. Error Indicator

0000001 P Error
0000010 Q Error
0000100 R Error
0001000 S Error
0010000 T Error
0100000 U Error
1000000 L Error

2.10 Stream Unit Index Words

$J_1(24)$ - first level reset address (not normally setup)

$J_H(24)$ - higher level reset address (not normally setup)

BM(8) - byte mask

$M_1(16)$ - count of bytes read on first level (not normally setup)

$M_H(16)$ - count of iterations of higher level (not normally setup)

UL(1) - upper lower triangle

0 - upper
1 - lower

TR(1) - triangular mode indexing

0 - No OP
1 - triangular

RM(2) - runout match

00 - no OP
01 - match only
10 - runout only
11 - runout and match

FL(2) - Flag

00 - no flag
01 - Flag 1
10 - Flag 2
11 - Flag 3

CC(1) - continue chain

0 - no OP (STOP)
1 - continue indexing

FF(1) - first to follow

0 - no OP
1 - following level is virtual first

BL(1) - branch level

0 - no OP
1 - branch to level at Br Hi - Br Lo address

SR(1) - suppress S address reset

0 - reset S address
1 - suppress S address reset

NS(1) - nested sequential

0 - nested
1 - sequential

FS(1) - first subsequent (not setup)

RC(1) - R control

0 - the data in R is sent to memory totally obliterating any previous data in that memory word. Any bits not specified will be made zeros.
1 - the data formed in R will replace the corresponding bits in memory but not affect adjacent bits.

NO(1) - nested offset

0 - nested
1 - offset

$I_1(25)$ - first level increment in bits for NO = 0

$I_1(10)$ - first level increment in bits for NO = 1. Sign from $I_1(25)$ applies

$O_1(14)$ - first level offset in bits for NO = 1. Sign following NO bit applies

$N_1(16)$ - number of bytes read for NO = 1.

$N_1(13)$ - number of bytes read for NO = 1.

RBL(3) - residual byte length in bits for NO = 1.

$I_H(25)$ - higher level increment in bits for NO = 0.

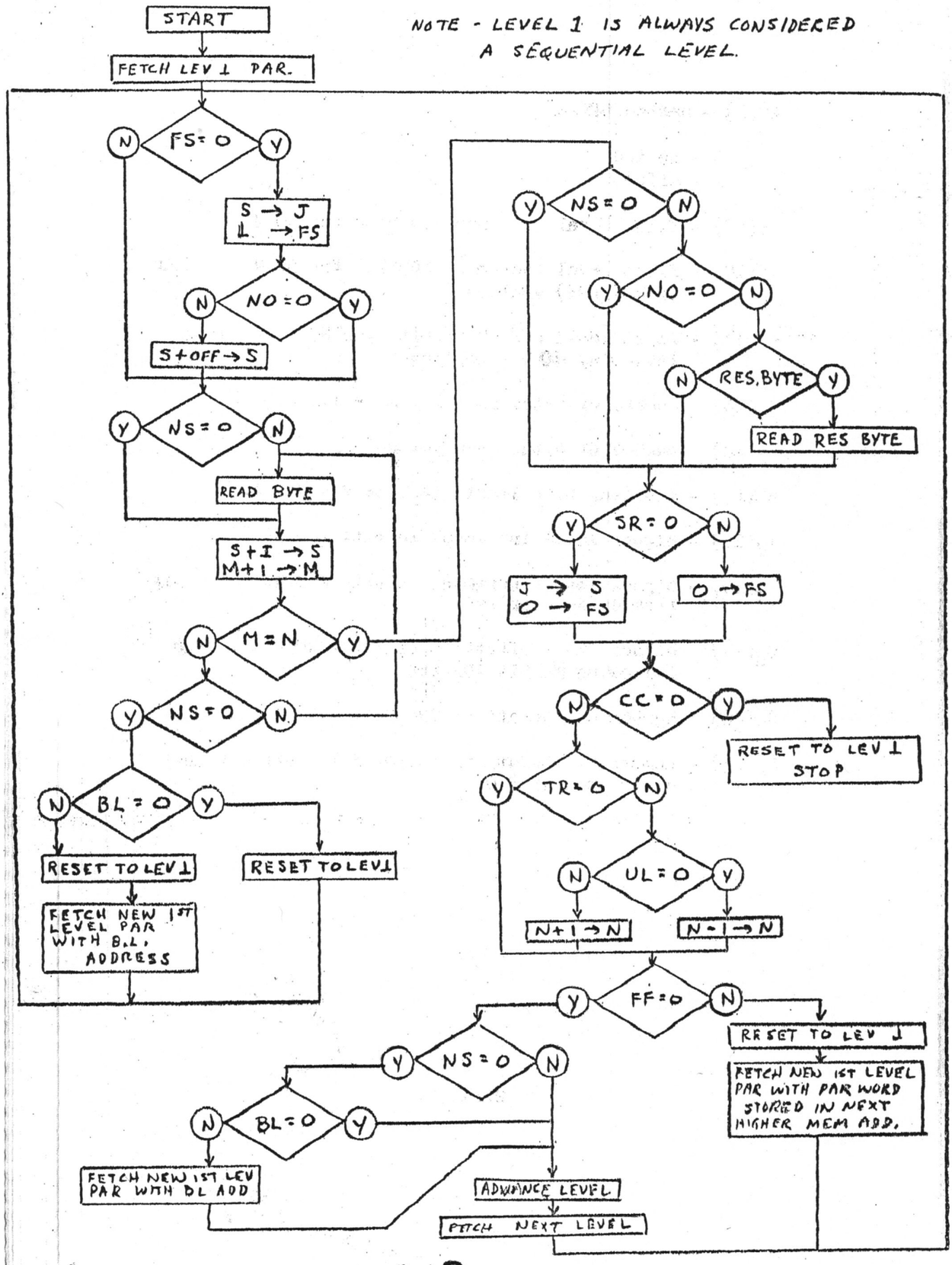
$I_H(10)$ - higher level increment in bits for NO = 1. Sign from $J_H(25)$ applies

$O_H(14)$ - higher level offset in bits for NO = 1. Sign following NO bit applies.

$N_H(16)$ - number of iterations for NO = 0.

$N_H(13)$ - number of iterations for NO = 1. RBL does not apply when NS = 0.

2.11 STREAM UNIT INDEXING CONTROL SEQUENCE



3.5 SILS Stream Indirect Load and Store

This instruction transmits fields from Q to R. Either the address of Q or the address of R is an indirect address obtained by the table lookup with data from P as an argument.

N_1T bytes are read from P to T as the first source. There is no second source. P indexing may be nested or sequential. TBA, RPL, TAM, S_{IT}, I_{1t}, and N_{IT} are set up. The K field of setup is not used. The KK field in the SILS instruction is used in determining the counting positions.

TAM: 000-NoOP
001-The assembled address is sent to S_Q or S_R for LS = 0 or 1, respectively.
010-The reference is sent to S_Q or S_R for LS= 0 or 1, respectively.
011-The address portion of the half word referenced by the first 19 bits of the assembled table address is sent to S_Q or S_R for LS - 0 or 1, respectively. Then the bit in memory addressed by the whole 24-bit assembled table address has a 1 or-ed to it.
100-(1/2-microsecond memory only) The address portion of the half-word referenced by the first 19 bits of the assembled table address is sent to S_Q or S_R for LS = 0 or 1, respectively. Then this same address portion of the memory word has a 1 added to it in a position determined by the KK field.

After S_Q or SR is supplied with an indirect address the data specified by all levels of Q including H_L is run out serially to R.

For LS=0(load) all stream indexing for P, Q, R, and T must be specified except S_Q, 1_{2T}, N_{2T}.

For LS=1(store) all stream indexing for P, Q, R, and T must be specified except S_R, second and higher levels of R, I_{2T}, N_{2T}, and S_{2T}.

The operation stops at the end of the highest level in P.

CHAPTER 4

SPECIAL FEATURES

4.1, Interrupts

The following criteria are necessary is appropriate interaction between the streaming mode operation of Harvest, the Harvest interrupt scheme, and the use of the Supervisor as a master control program is to be expected.

1. Interrupts must not cause the loss of data;
2. Interrupts must not be postponed indefinitely;
3. Boundary protection must be absolute.

As nearly as is practical, these criteria will be met.

A list of streaming mode indicators follows:

Streaming Mode

OP - Invalid
Memory Count Overflow
Extract Wraparound
SACC Overflow
SACC THR
SACC 0

SACC Underflow

HL - P
HL - Q
HL - R
EOC - P
EOC - Q
EOC - R
SCTR= LIM
SCTR Overflow/Underflow
Debugging Interrupts

4.1.1 Program Interrupt System

The program interruption system may be enabled or disabled when branching into streaming mode. When enabled it forms a logical extension of and complement to the streaming adjustment system. Among its functions are monitoring input-output operations, recognizing improbable consequences of streaming operations, initiating data manipulations too cumbersome for steam processing, tracing for program debugging, and recognition of program or machine malfunction.

Because of the indefinite length of streaming instructions it is necessary to allow interruption within an instruction. It is ordinarily desirable to interrupt as soon as possible after an indicator is set. Since most interruptions will be processed in arithmetic mode and the streaming process subsequently resumed, the interruption must leave the stream in a condition from which continuation is possible. To implement these considerations, the byte bus is stopped as soon as the enabled indicator is recognized, which may be several cycles after the condition arises, but the rest of the machine is allowed to finish housekeeping operations before the interruption is executed. This always leaves stream indexing at a point where it is ready to resume and address the next byte.

It is possible for both adjustment and interruption to be ordered at the same time. In determination of priority the interruption stimuli are divided into two categories. Those which are definitely error conditions are given priority over the adjustments. If an adjustment is already in progress the interruption will occur at the end of its execution, even if other valid adjustment stimuli remain, or between words if it is a chained microinstruction. In the latter case remaining words in the chain are effectively treated as no-ops. For interruption stimuli which are not definitely errors, all called for adjustments are completed prior to execution of the interrupt.

The error class of interruption always occurs from within the instruction on which the error was made. The instruction counter if stored will give the address of the first word of that instruction. The normal class of interruptions have the same characteristics except where an adjustment of higher priority signals a skip to a new instruction. In this case the interruption will occur at the end of the first arithmetic instruction, unless it is a branch to streaming mode which will be interrupted internally. Interruption always has priority over end operation skips to I. C. + 3 or 4.

Certain routines which may be entered after interrupting a streaming instruction require use of the streaming mechanism. If the original instruction is later to be resumed the entire streaming setup must be stored. Before this can be done it is necessary to allow the interrupted instruction to proceed to a point where all essential control information is in addressable storage. This is accomplished by entering that

instruction with End of Sequence in the Stop field. At the first appropriate point the instruction will be terminated and the next instruction taken from I. C. - 3. This procedure is not necessary for a SBBB instruction since any stop is an End of Sequence. It is then only necessary to give the adjustment Process Remainder of Data in Pipeline.

A table of conditions and indicators follows.

INTERRUPT CONDITIONS
- CCP
- CCQ
- CCR
- Memory Count Overflow
- SACC Overflow/Underflow
- Lost Stimulus
- SCTR Overflow/Underflow
- Extract Wraparound
- OP Invalid
- Arith. Error
- Adj. Stimulus Recognized
- Debug
- Lookup In Wrong Memory

DEBUG CONDITIONS
- NOOP
- BLP
- BLQ
- BLR

4.2 Indirect Addressing

The indirect addressing property of the Harvest system enables the programmer to direct machine operations with the Sigma instruction set while using the Stream units to generate operand addresses. With this arrangement, index maintenance counting, value field updating, index reloading, etc., can be largely omitted from closed loops in the program.

Harvest system indirect addressing is defined to operate as follows:

> If the word address portion of the operand address in any Sigma instruction is found equal to 61, 62, 63, the current value of S-P, S-Q, or S-R, respectively, is substituted into the instruction. The format of the instruction governs the size of the Stream address field used - 18, 19, or 24 bits. The instruction is then executed as normal. The Stream unit referenced is "stepped" prior to supplying another address.

Sigma immediate and goemetric address fields are treated as such; i.e. apparent binary values of 61, 62, or 63 in these fields do not cause indirect addressing. Effective addresses are never indirect.

If a given indirect instruction causes an End of Chain stimulus, the interrupt will not be performed until after the instruction is executed. Subject instructions of the EXECUTE class of Sigma instructions will have the ability to indirectly address operands.

Since the insertion of Harvest generated addresses into Sigma instructions will occur during instruction indexing, no change in the execution time of isolated indirect instructions will result. However, if a sequence in which every instruction addresses operands per the same Stream unit occurs, reduction in performance may result. The extent of this effect is very sensitive to several characteristics of the instructions themselves.

This reduction is primarily a result of the nature of instruction and operand look ahead Sigma. The Sigma Index Arithmetic Unit prepares the instruction and fetches its memory operand before the execution of the instruction is actually begun. (Look ahead design provides for as many as four instructions being prepared, and having operands accessed into the computer at any one time.) If an interrupt occurs after instruction preparation but before execution is begun, the instruction must not have had any effect on any addressable registers. It is therefore apparent that if several successive instructions request an address from the same Stream unit, look ahead must be prevented exercising its full capabilities in order to avoid "false steps", i.e. incorrect advancing of index parameters.

Little effect on machine performance will result from a sequence which cycles through the three Stream units. Therefore, if the programmer is working with only one Stream unit in indirect addressing, reduction in speed may be circumvented by setting up the three units with identical parameters, adjusting the count fields to compensate for the cycling.

4.2 Address Interpretation

Arithmetic Mode

Effective Addresses	Interpretation
0 - 31	As in 7000 S Manual
32 - 60 *	Harvest setup registers
61 - 63	Invalid. Fetch or store to 0 set invalid address indicator
64 and higher	Core memory

Operand Addresses

0 - 60	As in 7000 S Manual
61 - 63	Indirect Addresses as defined in Harvest specs 25 May 1959
64 and higher	As in 7000 S Manual

Streaming Mode - Addresses as sent to memory from any source.

0 - 63	Invalid. Fetch or store to 0. Set invalid address indicator
64 and higher	Core memory

4.3 Arithmetic Mode

The arithmetic mode provides all the facilities of the basic Sigma (Stretch) computer as set forth in the 7000S preliminary manual. However, it must be emphasized that the treatment of addresses 32 through 63 is inherently different from that of a base system not equipped with HARVEST features. Programs intended for general use in either type of system must take this specific difference into account.

Certain instruction code points which are not used in the base system are employed in the HARVEST System to provide the additional arithmetic mode instructions TRANSMIT FULL WORD, TRANSMIT HALF WORD, CLEAR MEMORY, BRANCH ENABLED TO STREAMING, and BRANCH DISABLED TO STREAMING. Again general programs must avoid use of these code points as NO-OPS or invalid instructions.

4.4 Streaming Mode

The larger class of instructions added with the HARVEST features requires more code points than are available in arithmetic mode and in any case are more conveniently handled by radically different formats. The streaming mode brings into play a completely different set of bit-wise and field-wise instruction interpretations. The shift to streaming mode is accomplished by either of the arithmetic instructions BRANCH ENABLED TO STREAMING (BES) OR BRANCH DISABLED TO STREAMING (BDS) (What is enabled or disabled is the basic interrupt mechanism). Streaming mode instructions are always three consecutive full words on full word boundaries. The half word address bit (bit 18 or 50) of both BES and BDS is used to control whether streaming is to be simply resumed from a previous interruption or whether a completely new stream is to be initiated.

The shift back to arithmetic mode occurs whenever any new instruction is fetched. For example, the taking of an interrupt or the execution of any "Go to I. C." adjustment resets the mode to arithmetic. In particular the next instruction after a streaming instruction will always be interpreted as an arithmetic instruction. Also, as implied in the preceding paragraph, the next instruction after a BES or BDS will always be interpreted as a streaming instruction.

4.5 Program Debugging and Maintenance

Both debugging and maintenance procedures make use of the Base Computer Interrupt system together with a special debugging field and an automatic readout of all register and control trigers (SCAN) which is independent of the normal computer functions. In addition, all adjustment stimuli are fed through a selecting mask and ORed into a bit of the interrupt register. Adjustments, interrupts, error responses, and debugging aids are interrelated as shown in the simplified schematic.

4.5.1 Stops

Normal program operation, debugging, and maintenance all may involve breaks or stops in the smooth flow of streaming. Because of time storage in the computer wiring and clock circuits even the most urgent stop is not quite instantaneous but necessarily allows one more cycle of operation after that in which the stop is signalled. Less urgent stops are restricted in their occurrence by logically required relations between the various asynchronous units. The various kinds of stops are listed belong in order of decreasing possible frequency per machine cycle; not necessarily decreasing probability of occurrence:

SCAN STOP

Directly initiated by any machine error signal (if manual maintenance console switch is not inhibited), by program setting of the addressable single scan trigger, or by any masked-on debugging stimulus if the scan bit is on. Machine is stopped after following cycle by inhibition of clock pulses to all sections. Ensuing read out (SCAN) of all registers and control triggers is accomplished independently of normal function and does not alter machine state. Because resumption of clocking after the scan allows operation to continue as if the scan had not intervened this stop can be allowed on any cycle.

Adjustment Stop

Initiated by the occurrence of either one of the five specific adjustment stimuli specified in the five adjustment half words accompanying the stream instruction or by the occurrence of any stimulus enabled by the stimulus mask.

Adjustment Stop (cont'd)

Stream operation is stopped after the following cycle by inhibition of clock pulses to the pipeline area. Stream index units P, Q, and R are not directly inhibited but are allowed to stop as required by the normal stream flow control. Indexing level changes normally arising from the adjustment signal cycle or the one following are allowed to occur. Adjustments affecting such an index unit are not taken until the unit finishes level changing to the next reading level. Adjustments not affecting such an index unit are taken without delay.

Interrupt Stops

Initiated by the program interruption system of the base computer. In the absence of adjustments interrupts will be taken on any byte processing cycle. If both an adjustment and an interrupt stop are signalled within the same cycle, the class 1 (error indicator) interrupts are given first priority; adjustments are given second priority; and class 2 interrupts are given third priority.

Instruction Stop

Initiated by a stimulus satisfying the STOP field of the instruction. This stop is specifically intended to bring the stream to rest at a reproducible point.

4.5.2 Maintenance Scan

The maintenance scan is initiated after a scan stop. It is independent of normal computer operations and produces a read-out of all register and control triggers without altering their states. In normal maintenance use the information is sent to the maintenance punch to provide the engineer with a permanent record of the computer contents as near as possible to the time of error occurrence. Alternatively the scan output can be sent to memory via the Exchange. When this is desiired the Exchange must previously have been given an appropriate instruction. A scan to memory will not be also punched.

4.5.3 Debugging Aids

A four bit debugging field is provided in setup to allow the programmer to command interrupts, scans, or both on the

4.5.3 Debugging Aids (cont'd)

occurrence of selected discontinuities in streaming. Three bits are used as a mask to specify action separately on the occurrence of any adjustment, any branch level (BL) in indexing, or any first-to-follow (FF)* bit in indexing. The fourth bit determines whether or not a scan is taken as part of the action. Alternatively a single scan can be initiated at any point in the interrupt routine by setting on the addressable single scan trigger by a suitable instruction.

* This is revised to be any index flag ($F_{12}F_2$)