HARVEST Assembly Program - HAP

By: Raymond W. Southworth

26 October 1959

Introduction

This report is a revision of HAP Report #2 and includes the changes and modifications agreed upon in the joint meeting of NSA and IBM personnel on October 5. It was also decided then to accept the proposal in Report #2 that the programmer might write as part of the set-up some of the functions actually represented in the stream instruction in the machine word. Thus, the set-up word for the logic unit may include the logic operation; that for the table address assembly unit may include cell size, mode, and other one-bit codes; and that for the F unit may include group size. Whenever a stream instruction refers to such a set-up, these fields are assembled into the stream instruction. It is also possible to write the stream instruction in complete form or to get some fields from set-up and to overrule others. Examples illustrating these points are given on pages 11 and 12 of this report.

Fields which are underlined are programmer fields and can be any symbol allowed in STRAP I.

* Means the items are not usually set up by the programmer and may be omitted.

1. Stream Byte-by Byte

SBBB(gates), LUOP, GS, TA(CS, TAM, P/S, RBA), STOP(stimulus)

Ex: SBBB(P-LU, Q-LU, LU-R), EQT(P, NB), FLIP, STOP(FL2P)

Notes:

1. The gates may be written either in terms of the system symbols for the actual units connected or the numbers of the open gates.

Gate	Units Connected
0	Reserved
1	P-LU
2	Q-LU
3	TE-LU
4	P-TA
5	Q-TA
6	LU-TA or SCTR-TA
7	SCTR substitutes for LU output
8	LU output is used
9	LU-R or SCTR-R
10	TE-R
11	LU-SACC or SCTR-SACC
12	TE-SACC

2. If a field or sub-field is null, the corresponding bits will be set to zero.

3. The various fields may be written in any order, with the exception that SBBB(gates) must be the first field.

4. The above format for SBBB includes all fields present in the 32-bit machine instruction. Several of the fields may, for programming convenience, be written in the set-up, as shown below in the discussion of those words.

5. The following gate combinations are not allowed:
(2, 3), (5, 6), (3, 6), (7, 8), (9, 10), (11, 12), (1, 3, 4).
Also, if TE is connected, at least one TA input is needed.

6. The codes for LUOP, GS, CS, and TAM are given under the description of the set-up words for the logic unit, the F unit, and the table address assembly unit.

2. Adjustments (4 types)

ADJ#(stimulus), action 1, action 2, action 3

ADJ#(stimulus, AND), action 1, action 2, action 3

ADJ#(stimulus, BR), action 1, action 2, relative address

ADJ#(stimulus, CHAIN), action 1, action 2, relative address

Ex: ADJ1(NW.NY), REP(P, Q), RESET(SACC)
Ex: ADJ5(KB1, CHAIN), ADV(P), NOP, JOE-PETE

Notes:

1. The adjustments may be numbered if desired but they will be assembled in the order written.

2. An action that has more than one operand, such as REP(P, Q) may be written either as above or separated as REP(P), REP(Q). It is the responsibility of the programmer to use no more than three actions in any one adjustment word. Also, the actions will be assembled in the order either written explicitly or implied by the order of the grouped operands.

3. In the first type of adjustment, there are simply three action fields. In the second type, the stimulus of the next lower-order priority adjustment must also be present for the actions to take place. In the third type, there is a skip relative to the address of the stream instruction to an instruction in the arithmetic mode. In the fourth type there is a skip relative to the address of the stream instruction to another adjustment; in the second example above JOE is the address of the adjustment being chained to, and PETE is the address of the stream instruction.

3. Set-up

SETUP

SETUP END

Notes:

1. SETUP and SETUP END are pseudo-ops to be used whenever it is desired to write a full set-up of 10 words. SETUP will reserve the block, beginning at a full word, set all words to zero, and then OR in the given fields. The following set-up words are self-identifying and may be written in any order. They are assumed to include all words up to the first word which is not a set-up word, or up to SETUP END, whichever occurs first.

4. Match Units

WMAT(character, connections, R/F), WOR, action

XMAT(character, connections, R/F), XOR, action

YMAT(character, connections, R/F), YOR, action

ZMAT(character, connections, R/F), ZOR, action

Ex: XMAT((8)377, P.Q, F), XOR, OMIT ALL

Notes:

1. The match character may be given by means of any entry mode permitted in STRAP, and also in hexadecimal.

2. The connections as used to cause a stimulus for adjustments, counters, and group size are indicated by a period between units when logical AND is meant and by a V for logical OR. The connections are assumed to be only of the OR type when considering the stimulus for the omitting (swallowing) of bytes, as indicated in the stimulus XOR.

3. The R/F bit indicates whether the match is on only the rightmost bit or on the full byte. A null field is assumed to mean a match on the full byte.

5. SCTR Unit

SCTR(limit, value), stimulus, action

Ex: SCTR(1000), XVY, SC+1

Notes:

1. Limit and value must be written in the order given above.

6. SACC Unit

SACC(SAM, threshold, value), stimulus, action

Notes:

1. SAM, threshold and value must be written in the order given above.

2. The codes for SAM are:

U	Unsigned, normal
U16	Unsigned, 16 bit fields
S	Signed, normal
SR	Signed, reset negative values

7. Logic Unit

LU, action (modulus), GS, F(limit), (stimulus)

Ex: LU,C(P.Q)(150), FLIP, F(2), (KK, LM), SO

Notes:

1. The modulus as assembled will be positioned so as to eliminate leading zeros, as required in the machine word.

2. The possible actions are:

C0000	or	C(0)	Connect and pseudo-connect operations
C0001		C(P.Q)	
C0010		C(P.NQ)	
C0011		C(P)	
C0100		C(NP.Q)	
C0101		C(Q)	
C0110		C(PXVQ)	Exclusive OR
C0111		C(PVQ)	
C1000		C(NP.NQ)	
C1001		C(PEQ)	
C1010		C(NQ)	
C1011		C(PVNQ)	
C1100		C(NP)	
C1101		C(NPVQ)	
C1110		C(NPVNQ)	
C1111		C(1)	
MAX(P,Q)			Maximum of P and Q
MIN(P,Q)			Minimum of P and Q

(conclusion of logic unit operations)

EQT(P, 0)	Equals test
EQT(P, NB)	
EQT(0, P)	
EQT(NB, P)	
EQT(0, Q)	
EQT(NB, Q)	
GET(P-Q, 0)	Greater than or equals test
GET(P-Q, NB)	
LET(P-P, 0)	Less than or equals test
LET(Q-P, NB)	
MOD(P-Q)	P-Q, modulo the modulus
MOD(P-P)	Q-P, modulo the modulus
RDX(P+Q)	P+Q, using the modulus as a radix
MOD(P+Q)	P+Q, modulo the modulus

3. In each of the test functions, the comparison is between the P and Q bytes, always in that order. If the test is satisfied, the output is the first byte given within parentheses; if it is not satisfied, the output is the second byte.

Notes for F Unit

1. The stimulus in this case is written as the bit or bits to be tested. A one coming into any one of them will cause the counter to advance according to the indicated action. The test bits are KK, KL, KM, LK, LL, LM, MK, ML, and MM.

2. The action codes are:

NOP	
SO	Stay on one
I	Invert
ISO	Invert and stay on one

3. The group size field will be assembled into the stream instruction referring to this set-up.

8. Table Assembly Unit

TA(CS, TAM, P/S, RBA), TBA(MDM), TAP(TPS, TPI, TPN, TPJ*),
TAQ(TQS*, TQI*, TQN*, TQJ*), value, TM

Notes:

1. CS = cell size. Codes for cell size are:
   00 = 0
   01 = C8
   10 = C16
   11 = C24

2. The codes for TAM are:

NOP	
ADTE	No memory reference. Address goes to TE.
EXT	Referenced word is extracted.
OR	A 1 is OR-ed into referenced word.
EXTOR	Extract and OR
M+1	A 1 is added to memory word.
EXTM+1	Extract and add a 1.

3. P/S = parallel or series addition of P and Q bytes.

4. RBA if present indicates TBA is replaced by address just formed.

5. TBA = table base address.

6. MDM if present indicates memory distributor mode is to be used.

7. TPS = initial offset of P stream.

8. TPI = increment to be applied tc offset of P stream.

9. TPN = number of bytes from P stream.

10. TPJ = reset address for P bytes (not usually set up).

11. TQS, TQI, TQN, and TQJ apply to the Q stream.

12. Value (not usually set up).

13. TM = count of bytes (not usually set up).

9. Table Extract Unit

TE, TEI, TEN, TEBM, TES*, TEJ*, TEM*

Notes:

1. TEI = increment to be added to address.
2. TEN = number of bytes.
3. TEBM = byte mask for TE unit.
4. TES = initial address for TE (not usually set up).
5. TEM = count of bytes (not usually set up).
6. TEJ = reset address (not usually set up).

10. Stream Stimulus Mask

SSM(mask)

Ex: SSM(FL1P, LB1)

Notes:

1. The mask may be written in numeric form with an entry mode or it may be written as a list of the system symbols for the stimuli to be tested. The system symbols for the maskable stimuli are:

FL1P	Flag 1, P
FL2P	
FL3P	
FL1Q	Flag 1, Q
FL2Q	
FL3Q	
FL1R	Flag 1, R
FL2R	
FL3R	
ELTE	End of level, TE
EG	End of group
SAGETH	SACC greater than or equal to threshold
SABN	SACC becomes negative
SCLIM	SCTR = limit
W	W match unit signal
X	
Y	
Z	

(conclusion of ...skable stimuli)

NW	Not W
NX	
NY	
NZ	
F1	F limit
KB1	KB 1
LB1	
MB1	
KG1	KG 1
LG1	
MG1	
EC	Any end of chain

11. Debug Code

DEBUG(mask)

Ex: DEBUG(SCAN, ADJ, BL, FLAG)

Notes:

1. The mask may be written either in numeric form with an entry mode or it may be written as a list of the system symbols for the bits to be tested. The system symbols for the four bits are:

SCAN	Scan bit
ADJ	Adjustment stop
BL	Any branch level
FLAG	Any flag

12. Starting Addresses for Stream Units and Index Tables

PAD(stream address, index table address)

QAD(stream address, index table address)

RAD(stream address, index table address)

13. Index Words

P
QX(mode, CC/EC, FF, SR, RC, M * / R , FS), N, I, BM/BRLA, offset, RBL, J,M * *
R RM
0

(See notes on following page)

Notes:

1. The allowable modes are:

TRUN	Triangular, upper, normal
TRUO	offset
TRLN	lower, normal
TRLO	offset
NESN	Nested, normal
NESO	offset
SEQN	Sequential, normal
SEQO	offset

2. EC = end chain; CC continue chain.
3. FF = following level is virtual first level.
4. SR = suppress reset of S address.
5. RC = R control bit: data formed in R will replace corresponding bits in memory but will not affect adjacent bits.
6. M = match only.
7. R = runout only.
8. RM = runout and match.
9. FS = first subsequent (not usually set up).
10. N = number of bytes.
11. I = increment to be applied to address.
12. BM = byte mask.
13. BRLA = branch level address.
14. RBL = residual byte length, used only in an offset mode.
15. J = reset address (not usually set up).
16. M = count of bytes (not usually set up).
17. For index words with flag bits, the codes PXFL1, PXFL2, and PXFL3 should be used.
18. For a branch level word, the code PXBL should be used. The branch address is then written in place of the byte mask.
19. For branch level and flags, the codes PXFL1BL, PXFL2BL, and PXFL3BL should be used.
20. The programmer fields must be in the order shown. Null fields are indicated by a zero; field drop-out is from right to left.

The following examples are intended to illustrate the general procedure to be followed in writing a stream instruction.

Example 1:

Here the SBBB instruction is written with all the fields contained in the actual machine word.

```
TI, 10, JOE, $HR                    'Transmit set-up to registers

BES, PETE

JOE SETUP
    WMAT(character, connections), WOR, action

    SCTR(limit, value), stimulus, action

    SACC(mode, threshold, value), stimulus, action

    LU(modulus)

    F(limit), (stimulus), action

    TA, TBA(MDM), TAP(TPS, TPI, TPN)

    TE, TEI, TEN, TEBM

    SSM(mask)

    DEBUG(mask)

    PAD(stream address, index table address)

    SETUP END

PETE SBBB(gates), LUOP, GS, TA(CS, TAM, P/S, RBA), STOP(stimulus)

    ADJ1(stimulus), actions

    ADJ2(stimulus), actions

    ADJ3(stimulus), actions

    ADJ4(stimulus), actions

    ADJ5(stimulus), actions

    B, address

    ADJ6(stimulus), actions
            etc.
```

Example 2:

Here the SBBB instruction is to be filled in from the set-up by the assembly program. Words in the set-up which are the same as those above have been omitted, but would, of course, have to be included in any real problem.

```
TI, lO, JOE, $HR
BES, PETE
JOE SETUP
     LU, action (modulus) GS, F(limit), (stimulus), action
     TA(CS, TAM, P/S, RBA), TAP(TPS, TPI, TPN)
     .
     .
     .
     SETUP END
PETE SBBB(gates), STOP(stimulus), SETUP (JOE)
     ADJ1(stimulus), actions
               etc.
```

Example 3:

Here the SBBB instruction is to be filled in partly from the "set-up".

```
     IT, lO, JOE, $HR
     BES, PETE
JOE SETUP
     LU(modulus)
     TA(CS, TAM, P/S, RBA), TBA(MDM), TAP(TPS, TPI, TPN)
PETE SBBB(gates), LUOP, GS, STOP(stimulus), SETUP (JOE)
               E    etc.
```

Notes:

1. Overruling of the set-up as in Example 3 above may be done only for

a complete field. Thus, LUOP, or TA(CS, TAM, P/S, RBA), may be overruled, but not just the TAM field in TA, for example.

14. Formats for the Hybrid Instructions

SNOP, GS, STOP(stimulus)

SMER(UP/DN, IN/EX, SIM/OFF), STOP(stimulus)

SSER(DATA/AD, ORD/RAN, UP/DN, SIM/OFF, SCD), STOP(stimulus)

Note:

The codes for SCD, the search condition, are

PLEQ	P less than or equal to Q
PGEQ	P greater than or equal to Q
PLQ	P less than Q
PGQ	P greater than Q
PEQ	P equal to Q
PNEQ	P not equal to Q

SSEL(LST/GST, SIM/OFF), STOP(stimulus)

STIR(RPL/TAKE, IC/DC, UP/DN, SIM/OFF), STOP(stimulus)

SQNL(P-TA, Q-TA, TE-SACC, TE-R), GS, TA(CS, TAM, P/S, RBA), STOP(stimulus)

Note:

The table entry format for use with SQNL is

TEY(Q/R, EC/CC), N, offset, (entry mode) data, address
30 bits 18 bits

SILS(LU-SACC, LU-R), (LD/ST), GS, TA(CS, TAM, P/S, RBA), STOP(stimulus)

Notes:

1. SQNL and SILS may be written without the GS and TA fields if desired. In this case, as with SBBB, reference may be made to a "set-up" in which those fields have been written, and they will then be assembled into the above instructions.

HAP Mnemonics and System Symbols

Units

P	stream unit
Q	
R	
TA	table address assembler
TE	table extract unit
LU	logic unit
F	F unit
SCTR	statistical counter
SACC	statistical accumulator
WMAT	W match unit
XMAT	X match unit
YMAT	Y match unit
ZMAT	Z match unit

Fields in SBBB and the Hybrids

GATES	
GS	group size
LUOP	logic unit operation
TAPOS	TA parallel or series bit
TARBA	replace base address bit
CS	cell size
STOPSTIM	stop stimulus
DATAAD	data or address bit
INTEXT	internal or external bit
UPDN	up or down bit
SIMOFF	simple or offset bit
SCD	search condition
ORDRAN	ordered or random bit
LSTGST	least or greatest bit
RPLTAKE	replace or take bit
ICDC	instruction control or data control bit
PQ	gates field in SQNL
LDST	load or store bit

Fields in Adjustments

ADJSTIM	stimulus
ADJMODE	mode
ADJACT1	action 1
ADJACT2	action 2
ADJACT3	action 3 or relative branch address

Registers (all to be preceded by $, as in $HR)

$HR	harvest registers
WCHAR	W match character
WCON	W connections
WSPAN	W span bit
WOP	W operation
WMODE	W mode, OR/AND
XCHAR	
XCON	
XSPAN	
XOP	
XMODE	
YCHAR	
YCON	
YSPAN	
YOP	
YMODE	
ZCHAR	
ZCON	
ZSPAN	
ZOP	
ZMODE	
SSM	stream stimulus mask
SATH	SACC threshold
SASTEPSTIM	SACC stimulus
DEBUG	DEBUG code
SACC	
SCTR	

SAMODE	SACC mode
SCMODE	SCTR mode
SCSTEPSTIM	SCTR step stimulus
SCLIM	SCTR limit
F	F unit
TBA	table base address
CS	table cell size
MDM	memory distributor mode bit
TAPS	initial offset, P stream
TAPI	increment
TAPN	number of bytes
TAPJ	reset address
TAPM	count of bytes
TAPFS	first-subsequent bit
TBS	bootstrap for TA
TAQS	initial offset, Q stream
TAQI	increment
TAQN	number of bytes
TAQJ	reset address
TAQFS	first-subsequent bit
MOD	modulus
TES	initial address, TE unit
TEI	increment
TEN	
TEM	
TEBM	byte mask for TE

SSS	stream stimulus mask
PS	start address, P stream
PIX	index table, P stream
QS	start address, Q stream
QIX	index table, Q stream
RS	start address, R stream
RIX	index table, R stream
ERRIND	error indicator register
PBS	boot strap for P
QBS	boot strap for Q
RBS	boot strap for R
PCBS	boot strap, program controlled
TE	table extract unit
R1	first word of R unit
R2	second word of R unit

Index Words

IXI	increment
IXIO	increment with offset
IXO	offset
IXNOFF	normal or offset bit
IXUL	upper or lower bit
IXTR	triangular bit
IXRM	runout-match field
IXN	number of bytes
IXRBL	residual byte length
IXNO	number of bytes with offset

IXFL1	flag 1
IXFL2	flag 2
IXFL3	flag 3
IXCCEC	continue chain or end chain bit
IXFF	first-to-follow bit
IXSR	suppress reset bit
IXFS	first subsequent bit
IXJ	reset address
IXBM	byte mask
IXM	count of bytes
IXBL	branch level bit
IXNS	nested or sequential bit
IXBRHO	branch address, high order part
IXBRLO	branch address, low order part

Indicators in Sigma Interrupt Mechanism used during Streaming

Bit Address	Code	
11.15	OP	operation invalid
11.22	MCO	memory count overflow
11.23	EW	extract wraparound
11.28	SAOU	SACC overflow or underflow
11.29		
11.32		
11.33		
11.41	ECP	end chain, P
11.42	ECQ	end chain, Q
11.43	ECR	end chain, R
	LST	lost stimulus
	SCOU	SCTR overflow or underflow
	AERR	arithmetic error

Stimulus Codes

NOP	
INIT	initial
EL1P	end level 1, P
EL1Q	
EL1R	
EL2P	end level 2, P
EL2Q	
EL2R	
ELTE	end level in TE
EG	end of group
ESQ	end of sequence
FL1P	flag 1, P
FL1Q	
FL1R	
FL2P	flag 2, P
FL2Q	
FL2R	
FL3P	flag 3, P
FL3Q	
FL3R	
FFP	first-to-follow, P
FFQ	
FFR	
BLP	branch level, P
BLQ	
BLR	

Stimulus Codes continued

ECP	end chain, P
ECQ	
ECR	
ECTE	
BYP	byte from P
BYQ	
BYR	byte to R
OPLU	operation in LU
+BYSA	+byte into SACC
-BYSA	
SAGETH	SACC greater than or equal to threshold
SABN	SACC becomes negative
SALTH.EG	SACC less than threshold AND end of group
SAGETH.BY	SACC $\geq$ threshold AND byte into SACC
SCSTEP	SCTR steps
SCLIM	SCTR = limit
SCNLIM.EG	SCTR $\neq$ limit AND end of group
W	W match unit signal
X	
Y	
Z	
XVY	S or Y signal
W.X	W AND X signal
W.Y	
NW	not W
NX	
NY	
NZ	

Stimulus Codes (continued)

NW.NY	not W AND not Y
NW.NX.NY.NZ	
F1	F = limit
F0	F ≠ limit
F0.EG	F ≠ limit AND end of group
KB0	KB = 0
KB1	KB = 1
LB0	
LB1	
MB0	
MB1	
KG0	
KG1	
LG0	
LG1	
MG0	
MG1	
KB0.F0	KB = 0 AND F ≠ limit
KB1.F0	
KB0.F1	
KB1.F1	
LB0.F0	
LB1.F0	
LB0.F1	
LB1.F1	

Stimulus Codes (continued)

MB0.F0	MB = 0 AND F ≠ limit
MB1.F0	
MB0.F1	
MB1.F1	
KG0.EG	KG = 0 AND end of group
KG1.EG	
LG0.EG	
LG1.EG	
MG0.EG	
MG1.EG	
MC0	memory count overflow
SAOU	SACC overflow or underflow
LST	lost stimulus
SCOU	SCTR overflow or underflow
EW	extract wraparound
OP	operation invalid
AERR	arithmetic error
ADJ1	adjustment 1
ADJ2	
ADJ3	
ADJ4	
ADJ5	
ADJ	any adjustment
ADJSTIM	adjustment stimulus
EC	end of chain
FLAG	flag bit

Stimulus Codes (concluded)

DEBUG	debug signal
PERR	P error
QERR	
SERR	
TAERR	
TEERR	
LUERR	
MERR	memory error
BYSA	any byte into SACC
STOP	
UUA	ungated unit adjusted
TRIGON	trigger on

Adjustment Reactions

Note: Possible operands are shown in parentheses

NOP	
DIS MU	disable match units
DIS THIS BY (2, 3, 4, 5)	disable stimuli this byte
DIS TO EG (1, 2, 3, 4, 5)	disable stimuli to end of group
RESET(P, Q, R, SACC, SCTR, TA, FAG)	
RESET THRU FL1(P, Q, R)	
RESET THRU FL2(P, Q, R)	
RESET THRU FL3(P, Q, R)	
SKIP(R, TA, TE)	
INSERT WCHAR IN LU	
INSERT XCHAR IN L	
INSERT YCHAR IN L	
INSERT ZCHAR IN L	
INSERT MOD IN L	
INSERT MOD IN TE	
SC+1	step SCTR by +1
SC-1	
SC+TBA	add TBA to SCTR
SA+1	step SACC by +1
ADV(P, Q, R)	advance to next level
ADV NEXT ABOVE FL1(P, Q, R)	
ADV NEXT ABOVE FL2(P, Q, R)	
ADV NEXT ABOVE FL3(P, Q, R)	

Adjustment reactions (concluded)

REP(P, Q)	repeat byte
RUN TO R(P, Q)	runout
RUN TO R THRU FL1(P, Q)	
RUN TO R THRU FL2(P, Q)	
OMIT THIS BY(LU, R)	swallowing action
OMIT NEXT BY(P, Q, TE)	
MATCH(P, Q)	
MATCH THRU FL1(P, Q)	
MATCH THRU FL2(P, Q)	
MATCH THRU FL3(P, Q)	
RO8(SACC, SCTR)	read out
RO16(SACC, SCTR)	
RO24(SACC)	
ROR(SACC)	read out and reset
ROR(SCTR)	
CANCEL(TA)	
PUT TBA-1 FOR TAD	substitute TAB-1 for T address