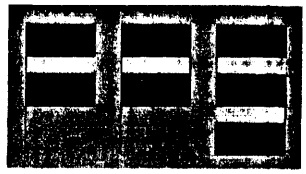

FLOATING POINT
SYSTEMS, INC.

FPS AP-120B
MATH LIBRARY ROUTINES

FPS-7352

FPS-7352

PUBLISHED IN THE UNITED STATES OF AMERICA
JUNE 1977

P.O. BOX 23489 PORTLAND, OR 97223 11000 S.W. 11TH STREET, BEAVERTON, OR 97005 (503) 641-3151 TLX: 360470 FLOATPOINT PTL

NEW APPROACHES TO THE AP-120B

I. HIGHER LEVEL LANGUAGES FOR THE AP-120B

1. What language? (Is Fortran immortal?) Some choices:

    A. Scalar (Standard) Fortran, vectorized automatically
       Questions:
       i. How much does the user have to modify his code to make it vectorizeable?
       ii. Who splits the code between the host computer and the AP, the user, or the vectorizer?
       iii. Does the vectorizer run on a large computer, or a mini?

    2. Scalar Fortran with Vector calls (like the VFC)
    3. Fortran with vector syntax added, i.e. ability to declare vectors and operate on them. PL/1 allows this. Some possibilities
       `A(1,*)=B(1,*)+C(1,*)` add the first column of two matrices
       `A=A+10.0` add 10.0 to all the elements of A

    4. Other scalar languages (ALGOL, etc)
    5. A vector language such as APL.
    6. Optimization, how good?

2. How does the higher level language affect the AP-120B - Host relationship?

II. THE AP-120B/HOST COMPUTER DUAL PROCESSOR

1. The current approach: the AP-120B as tightly controlled slave of the host computer.
2. A 'virtual channel' to lessen operating system overhead.
3. The Vector Function Chainer: moving the CALLS from the host to the AP-120B.
4. Who is master? or how to coordinate host I/O with AP-120B computation.

APPENDIX B INDEX OF ROUTINES IN PAGE ORDER

Page	Name	Operation	Typical Execution Time/Loop (us)		Program Size (AP120B PS words)	
			167	333	167	333
		DATA TRANSFER AND CONTROL OPERATIONS (APEX)				
D-2	APPUT	PUT DATA INTO THE AP120B	#.#	#.#	0	0
D-4	APGET	GET DATA FROM THE AP120B	#.#	#.#	0	0
D-6	APCLR	INITIALIZE THE AP120B	#.#	#.#	0	0
D-7	APWD	WAIT FOR AP120B DATA TRANSFER	#.#	#.#	0	0
D-8	APWR	WAIT FOR AP120B PROGRAM EXECUTION	#.#	#.#	0	0
D-9	APWAIT	WAIT FOR AP120B	#.#	#.#	0	0
D-10	APGSP	READ AN AP120B S-PAD REGISTER	#.#	#.#	0	0
D-11	APCHK	CHECK AP120B PROGRAM ERROR CONDITION	#.#	#.#	0	0
D-12	APSTAT	GET AP120B HARDWARE STATUS	#.#	#.#	0	0
		BASIC VECTOR ARITHMETIC				
D-15	VCLR	VECTOR CLEAR	0.2	0.3	13	4
D-16	VMOV	VECTOR MOVE	0.5	0.8	14	6
D-17	VSWAP	VECTOR SWAP	1.2	1.5	14	12
D-18	VFILL	VECTOR FILL	0.3	0.3	6	6
D-19	VRAMP	VECTOR RAMP	0.3	0.3	12	12
D-20	VNEG	VECTOR NEGATE	0.5	0.8	17	7
D-21	VADD	VECTOR ADD	0.8	1.3	17	10
D-22	VSUB	VECTOR SUBTRACT	0.8	1.3	17	10
D-23	VMUL	VECTOR MULTIPLY	0.8	1.3	17	12
D-24	VDIV	VECTOR DIVIDE	1.7	1.7	75	75
D-25	VSADD	VECTOR SCALAR ADD	0.5	0.8	16	8
D-26	VSMUL	VECTOR SCALAR MULTIPLY	0.5	0.8	17	10
D-27	VSQ	VECTOR SQUARE	0.5	0.8	9	9
D-28	VSSQ	VECTOR SIGNED SQUARE	0.5	0.8	18	10
D-29	VABS	VECTOR ABSOLUTE VALUE	0.5	0.8	14	7
D-30	VSQRT	VECTOR SQUARE ROOT	1.8	1.8	79	79
D-31	VLOG	VECTOR LOGARITHM (BASE 10)	4.7	4.7	52	52
D-32	VLN	VECTOR NATURAL LOGARITHM	4.0	4.0	52	52
D-33	VALOG	VECTOR ANTILOGARITHM (BASE 10)	4.8	5.0	39	39
D-34	VEXP	VECTOR EXPONENTIAL	4.2	4.2	42	42
D-35	VSIN	VECTOR SINE	4.4	4.4	46	46
D-36	VCOS	VECTOR COSINE	4.7	4.7	46	46
D-37	VATAN	VECTOR ARCTANGENT	8.7	8.7	89	89
D-38	VATN2	VECTOR ARCTANGENT OF Y/X	13.8	13.8	90	90
D-39	VRAND	VECTOR RANDOM NUMBERS	1.2	1.2	16	16
D-40	VMA	VECTOR MULTIPLY AND ADD	1.2	1.8	20	15

Page	Name	Operation	Typical Execution Time/Loop (us)		Program Size (AP120B PS words)	
			167	333	167	333
D-41	VAM	VECTOR ADD AND MULTIPLY	1.2	1.8	20	15
D-42	VSMSA	VECTOR SCALAR MULTIPLY AND SCALAR ADD	0.5	0.8	22	15
D-43	VMMA	VECTOR MULTIPLY, MULTIPLY, AND ADD	1.5	2.3	24	20
D-45	VAAM	VECTOR ADD, ADD, AND MULTIPLY	1.5	2.3	18	20
D-47	VAND	VECTOR LOGICAL AND	0.8	1.3	17	8
D-48	VEQV	VECTOR LOGICAL EQUIVALENCE	0.8	1.3	17	8
D-49	VOR	VECTOR LOGICAL OR	0.8	1.3	17	8
D-50	VFRAC	VECTOR TRUNCATE TO FRACTION	0.7	0.8	13	13
D-51	VINT	VECTOR TRUNCATE TO INTEGER	0.5	0.8	14	14
D-52	VINDEX	VECTOR INDEX	0.8	1.3	25	23
		VECTOR-TO-SCALAR OPERATIONS				
D-54	SVE	SUM OF VECTOR ELEMENTS	0.3	0.3	7	7
D-55	SVEMG	SUM OF VECTOR ELEMENT MAGNITUDES	0.3	0.3	10	10
D-56	SVESQ	SUM OF VECTOR ELEMENT SQUARES	0.3	0.3	10	10
D-57	SVS	SUM OF VECTOR SIGNED SQUARES	0.3	0.3	11	11
D-58	DOTPR	DOT PRODUCT	0.5	0.8	19	10
D-59	MAXV	MAXIMUM ELEMENT IN VECTOR	0.3	0.3	19	19
D-60	MINV	MINIMUM ELEMENT IN VECTOR	0.3	0.3	19	19
D-61	MAXMGV	MAXIMUM MAGNITUDE ELEMENT IN VECTOR	0.3	0.3	19	19
D-62	MINMGV	MINIMUM MAGNITUDE ELEMENT IN VECTOR	0.3	0.3	19	19
D-63	MEANV	MEAN VALUE OF VECTOR ELEMENTS	0.3	0.3	45	45
D-64	MEAMGV	MEAN OF VECTOR ELEMENT MAGNITUDES	0.3	0.3	48	48
D-65	MEASQV	MEAN OF VECTOR ELEMENT SQUARES	0.3	0.3	48	48
D-66	RMSQV	ROOT-MEAN-SQUARE OF VECTOR ELEMENTS	0.3	0.3	77	77
		VECTOR COMPARISON OPERATIONS				
D-68	VMAX	VECTOR MAXIMUM	0.8	1.3	19	14
D-69	VMIN	VECTOR MINIMUM	0.8	1.3	19	14
D-70	VMAXMG	VECTOR MAXIMUM MAGNITUDE	1.0	1.3	14	14
D-71	VMINMG	VECTOR MINIMUM MAGNITUDE	1.0	1.3	14	14
D-72	VCLIP	VECTOR CLIP	0.5	0.8	16	16
D-73	VLIM	VECTOR LIMIT	0.5	0.8	14	14
		COMPLEX VECTOR ARITHMETIC				
D-75	CVMOV	COMPLEX VECTOR MOVE	0.8	1.3	9	9
D-76	CVFILL	COMPLEX VECTOR FILL	0.5	0.7	8	8
D-77	CVCOMB	COMPLEX VECTOR COMBINE	1.1	1.7	10	10
D-78	CVREAL	FORM COMPLEX VECTOR OF REALS	0.8	1.2	9	9
D-79	VREAL	EXTRACT REALS OF COMPLEX VECTOR	0.5	0.8	15	7

Page	Name	Operation	Typical Execution Time/Loop (us)		Program Size (AP120B PS words)	
			167	333	167	333
D-80	VIMAG	EXTRACT IMAGINARIES OF COMPLEX VECTOR	0.5	0.8	16	8
D-81	CVNEG	COMPLEX VECTOR NEGATE	0.8	1.3	11	11
D-82	CVCONJ	COMPLEX VECTOR CONJUGATE	1.2	1.3	12	12
D-83	CVADD	COMPLEX VECTOR ADD	1.0	2.0	13	12
D-84	CVSUB	COMPLEX VECTOR SUBTRACT	1.0	2.0	13	12
D-85	CVMUL	COMPLEX VECTOR MULTIPLY	1.0	2.0	25	26
D-87	CVSMUL	COMPLEX VECTOR SCALAR MULTIPLY	0.8	1.3	11	11
D-88	CVRCIP	COMPLEX VECTOR RECIPROCAL	5.2	5.2	50	50
D-89	CRVADD	COMPLEX AND REAL VECTOR ADD	1.3	1.8	14	14
D-90	CRVSUB	COMPLEX AND REAL VECTOR SUBTRACT	1.3	1.8	14	14
D-91	CRVMUL	COMPLEX AND REAL VECTOR MULTIPLY	1.3	1.8	14	14
D-92	CRVDIV	COMPLEX AND REAL VECTOR DIVIDE	3.3	3.3	92	92
D-93	CVMA	COMPLEX VECTOR MULTIPLY AND ADD	2.7	2.7	30	30
D-95	CVMAGS	COMPLEX VECTOR MAGNITUDE SQUARED	0.7	1.2	13	18
D-96	SCJMA	SELF-CONJUGATE MULTIPLY AND ADD	1.3	1.5	15	15
D-97	POLAR	RECTANGULAR TO POLAR CONVERSION	19.5	19.5	119	119
D-98	RECT	POLAR TO RECTANGULAR CONVERSION	10.7	10.7	46	46
D-99	CDOTPR	COMPLEX DOT PRODUCT	0.7	1.3	15	16
		DATA FORMATING OPERATIONS				
D-101	VFLT	VECTOR INTEGER FLOAT	0.7	0.8	11	11
D-102	VFIX	VECTOR INTEGER FIX	0.7	0.8	15	7
D-104	VSCALE	VECTOR SCALE (POWER 2) AND FIX	0.7	0.8	12	12
D-106	VSCSCL	VECTOR SCAN, SCALE (POWER 2) AND FIX	1.5	1.7	19	19
D-108	VSHFX	VECTOR SHIFT AND FIX	0.7	0.8	9	9
D-109	VUP8	VECTOR 8-BIT BYTE UNPACK	0.5	0.5	61	61
D-110	VUPS8	VECTOR 8-BIT SIGNED BYTE UNPACK	0.9	0.9	69	69
D-111	VPK8	VECTOR 8-BIT BYTE PACK	0.9	0.9	62	62
D-112	VUP16	VECTOR 16-BIT BYTE UNPACK	0.8	0.8	49	49
D-113	VUPS16	VECTOR 16-BIT SIGNED BYTE UNPACK	1.3	1.3	61	61
D-114	VPK16	VECTOR 16-BIT BYTE PACK	0.8	0.8	40	40
D-115	VFLT32	VECTOR 32-BIT INTEGER FLOAT	1.7	1.7	61	61
D-116	VFIX32	VECTOR 32-BIT INTEGER FIX	1.2	1.2	35	35
		MATRIX OPERATIONS				
D-118	MTRANS	MATRIX TRANSPOSE	0.5	0.9	18	17
D-119	MMUL	MATRIX MULTIPLY	0.62*	0.83	59	59
D-121	MMUL32	MATRIX MULTIPLY (DIMENSION <=32)	0.50*	0.73	27	27
D-123	MATINV	MATRIX INVERSE	1.6 *	2.1	160	160
D-125	MVML3	MATRIX VECTOR MULTIPLY (3X3)	2.0 *	2.2	30	30
D-127	MVML4	MATRIX VECTOR MULTIPLY (4X4)	3.3 *	3.8	39	39
D-129	CTRN3	3-DIMENSION COORDINATE TRANSFORMATION	2.3 *	2.5	37	37
D-131	FMMM	FAST MEMORY MATRIX MULTIPLY	0.43*		61	

Page	Name	Operation	Typical Execution Time/Loop (us)		Program Size (AP120B PS words)	
			167	333	167	333
D-133	FMMM32	FAST MEMORY MATRIX MULTIPLY (<=32)	0.41*		33	
		FFT OPERATIONS				
D-136	CFFT	COMPLEX TO COMPLEX FFT (IN PLACE)	0.28	0.40	191	189
D-138	CFFTB	COMPLEX TO COMPLEX FFT (NOT IN PLACE)	0.20*	0.28	248	246
D-140	RFFT	REAL TO COMPLEX FFT (IN PLACE)	0.18*	0.27	260	258
D-142	RFFTB	REAL TO COMPLEX FFT (NOT IN PLACE)	0.14*	0.20	316	314
D-144	CFFTSC	COMPLEX FFT SCALE	0.8 *	1.3	42	42
D-145	RFFTSC	REAL FFT SCALE AND FORMAT	0.7	0.8	59	59
		AUXILIARY OPERATIONS				
D-148	CONV	CONVOLUTION (CORRELATION)	0.28*	0.28	106	106
D-150	DEQ22	DIFFERENCE EQUATION, 2 POLES, 2 ZEROS	0.8	0.8	25	25
D-151	VPOLY	VECTOR POLYNOMIAL EVALUATION	1.0 *	1.2	41	41
D-153	VSUM	VECTOR SUM OF ELEMENTS INTEGRATION	0.7	0.8	13	13
D-154	VTRAPZ	VECTOR TRAPEZOIDAL RULE INTEGRATION	0.7	0.8	16	16
D-155	VSIMPS	VECTOR SIMPSONS 1/3 RULE INTEGRATION	0.7	0.8	25	25
D-156	WIENER	WIENER LEVINSON ALGORITHM	0.50*	0.65	100	100
		SIGNAL PROCESSING OPERATIONS (optional)				
D-159	HIST	HISTOGRAM	1.3	1.4	69	69
D-160	HANN	HANNING WINDOW MULTIPLY	0.7	0.8	43	43
D-162	ASPEC	ACCUMULATING AUTO-SPECTRUM	1.3	1.5	15	15
D-163	CSPEC	ACCUMULATING CROSS-SPECTRUM	2.7	2.7	30	30
D-164	VAVLIN	VECTOR LINEAR AVERAGING	0.8	1.3	51	46
D-165	VAVEXP	VECTOR EXPONENTIAL AVERAGING	0.8	1.3	52	47
D-166	VDBPWR	VECTOR CONVERSION TO DB (POWER)	1.2	1.3	74	74
D-167	TRANS	TRANSFER FUNCTION	3.6	3.6	75	75
D-168	COHER	COHERENCE FUNCTION	4.3	4.8	88	93
D-169	ACORT	AUTO-CORRELATION (TIME-DOMAIN)	0.29*	0.29	113	113
D-171	ACORF	AUTO-CORRELATION (FREQUENCY-DOMAIN)	1.80*	2.70	376	357
D-173	CCORT	CROSS-CORRELATION (TIME-DOMAIN)	0.29*	0.29	113	113
D-175	CCORF	CROSS-CORRELATION (FREQUENCY-DOMAIN)	2.58*	3.93	388	365
D-177	TCONV	POSTTAPERED CONVOLUTION (CORRELATION)	0.30*	0.30	113	113
		TABLE MEMORY OPERATIONS (optional)				
D-180	MTMOV	VECTOR MOVE (MD TO TM)	0.2	0.3	7	7

Page	Name	Operation	Typical Execution Time/Loop (us)		Program Size (AP120B PS words)	
			167	333	167	333
D-181	TMMOV	VECTOR MOVE (TM TO MD)	0.2	0.3	5	12
D-182	MTIMOV	VECTOR MOVE WITH INCREMENT (MD TO TM)	0.5	0.5	7	7
D-183	TMIMOV	VECTOR MOVE WITH INCREMENT (TM TO MD)	0.3	0.3	12	12
D-184	TTIMOV	VECTOR MOVE WITH INCREMENT (TM TO TM)	0.5	0.5	7	7
D-185	MMTADD	VECTOR ADD (MD+MD TO TM)	0.5	0.7	17	13
D-186	MMTSUB	VECTOR SUBTRACT (MD-MD TO TM)	0.5	0.7	17	13
D-187	MMTMUL	VECTOR MULTIPLY (MD*MD TO TM)	0.5	0.7	17	9
D-188	MTMADD	VECTOR ADD (MD+TM TO MD)	0.5	0.7	17	9
D-189	MTMSUB	VECTOR SUBTRACT (MD-TM TO MD)	0.5	0.7	17	9
D-190	TMMSUB	VECTOR SUBTRACT (TM-MD TO MD)	0.5	0.7	17	9
D-191	MTMMUL	VECTOR MULTIPLY (MD*TM TO MD)	0.5	0.7	17	9
D-192	MTTADD	VECTOR ADD (MD+TM TO TM)	0.5	0.5	16	16
D-193	MTTSUB	VECTOR SUBTRACT (MD-TM TO TM)	0.5	0.5	18	18
D-194	TMTSUB	VECTOR SUBTRACT (TM-MD TO TM)	0.5	0.5	16	16
D-195	MTTMUL	VECTOR MULTIPLY (MD*TM TO TM)	0.5	0.5	17	17
D-196	TTMADD	VECTOR ADD (TM+TM TO MD)	0.5	0.5	18	18
D-197	TTMSUB	VECTOR SUBTRACT (TM-TM TO MD)	0.5	0.5	17	17
D-198	TTMMUL	VECTOR MULTIPLY (TM*TM TO MD)	0.5	0.5	17	17
D-199	TTTADD	VECTOR ADD (TM+TM TO TM)	0.7	0.7	9	9
D-200	TTTSUB	VECTOR SUBTRACT (TM-TM TO TM)	0.7	0.7	9	9
D-201	TTTMUL	VECTOR MULTIPLY (TM*TM TO TM)	0.7	0.7	12	12
		APAL-CALLABLE UTILITY OPERATIONS				
D-203	DIV	SCALAR DIVIDE	3.8 @ 3.8		28	28
D-204	SQRT	SCALAR SQUARE ROOT	3.8 @ 3.8		28	28
D-205	LOG	SCALAR LOGARITHM (BASE 10)	4.7 @ 4.7		37	37
D-206	LN	SCALAR NATURAL LOGARITHM	4.0 @ 4.0		37	37
D-207	EXP	SCALAR EXPONENTIAL	4.2 @ 4.2		28	28
D-208	SIN	SCALAR SINE	4.4 @ 4.4		31	31
D-209	COS	SCALAR COSINE	4.8 @ 4.8		31	31
D-210	ATAN	SCALAR ARCTANGENT	8.7 @ 8.7		46	46
D-211	ATN2	SCALAR ARCTANGENT OF Y/X	13.8 @13.8		46	46
D-212	SPFLT	FLOAT S-PAD INTEGER	0.8 @ 0.8		5	5
D-213	SPMUL	S-PAD MULTIPLY	2.3 @ 2.3		14	14
D-214	VFCL1	VECTOR FUNCTION CALLER (1 ARGUMENT)	0.8	1.0	10	10
D-215	VFCL2	VECTOR FUNCTION CALLER (2 ARGUMENT)	1.0	1.0	11	11
D-216	BITREV	COMPLEX VECTOR BIT REVERSE ORDERING	0.9	1.4	45	43
D-217	REALTR	REAL FFT UNRAVEL AND FINAL PASS	0.4	0.7	51	51
D-218	FFT2	RADIX 2 FFT FIRST PASS	1.3	2.7	16	16
D-219	FFT4	RADIX 4 FFT PASS	3.7	5.3	81	81
D-220	FFT2B	RADIX 2 FFT FIRST PASS + BIT REVERSE	1.3	2.7	25	25
D-221	FFT4B	RADIX 4 FFT FIRST PASS + BIT REVERSE	2.7	5.3	44	44
D-222	STSTAT	SET FFT MODE STATUS BITS	5.0 @ 5.0		19	19
D-223	CLSTAT	CLEAR FFT MODE STATUS BITS	0.5 @ 0.5		19	19
D-224	ILOG2	LOGARITHM (BASE 2)	4.0 @ 4.0		19	19
D-225	ADV2	ADVANCE POINTERS AFTER RADIX 2 FFT	0.7 @ 0.7		8	8

Page	Name	Operation	Typical Execution Time/Loop (us)	Program Size (AP120B PS words)
			167 | 333	167 | 333
D-226	ADV4	ADVANCE POINTERS AFTER RADIX 4 FFT	0.7 @ 0.7	8 8
D-227	SET24B	SETUP FOR FFT2B AND FFT4B	1.2 @ 1.2	8 8

Notes: #.# Timing host system dependent
* Refer to description of routine for explanation of timing
@ Total execution time

```
**********                                                **********
*        *                                                *        *
*  VADD  *          ---- VECTOR ADD ----                  *  VADD  *
*        *                                                *        *
**********                                                **********
```

PURPOSE: To add the elements of two vectors.

FORTRAN CALL: CALL VADD(A,I,B,J,C,K,N)

PARAMETERS:
A = Source vector base address
I = A address increment
B = Source vector base address
J = B address increment
C = Destination vector base address
K = C address increment
N = Element count

FORMULA: $C(mK) = A(mI) + B(mJ)$ for m=0 to N-1

DESCRIPTION: Adds N elements of vector beginning at address A to N elements of vector beginning at address B, and stores results in vector beginning at address C.

EXAMPLE: CALL VADD(0,1,200,1,400,1,100)
Stores into AP120B main data memory locations 400,401,....,498,499 the sum of the numbers in locations 0 and 200, 1 and 201, ... , 98 and 298, 99 and 299.

EXECUTION	BEST	TYPICAL	WORST	SETUP(us)	
TIME/LOOP:	0.5 (B)	0.8	1.0	2.7	(167 ns memory)
(us)	1.0 (A)	1.3	1.5	1.2	(333 ns memory)
PROGRAM SIZE:	17				(167 ns memory)
(AP words)	10				(333 ns memory)

APAL CALL:	JSR VADD	
SCRATCH:	SP(0,2,4,14,15),DPX(0,1),DPY(0),FA,MD,TM	(167 ns memory)
	SP(0,2,4,6),DPX(0),FA,MD	(333 ns memory)
EXTERNALS:	SPFLT	(167 ns memory)
	None	(333 ns memory)

```
**********                                                  **********
*        *                                                  *        *
* CVMAGS * ---- COMPLEX VECTOR MAGNITUDE SQUARED ---- * CVMAGS *
*        *                                                  *        *
**********                                                  **********
```

PURPOSE: To compute the squared magnitude of the elements of a complex vector.

FORTRAN CALL: CALL CVMAGS(A,I,C,K,N)

PARAMETERS:
A = Source vector base address
I = A address increment
C = Destination vector base address
K = C address increment
N = Complex element count

FORMULA: C(mK) = A(mI)**2 + A(mI+1)**2 for m=0 to N-1

DESCRIPTION: For each of the N complex elements of the complex vector beginning at address A, the square of the real part is added to the square of the imaginary part, and the result is stored into the real vector beginning at address C.

EXAMPLE: CALL CVMAGS(0,2,200,1,50)
Stores into AP120B main data memory locations 200,201,....,248,249 the sum of the squares of the numbers in locations 0 and 1, 2 and 3, ... , 96 and 97, 98 and 99.

EXECUTION TIME/LOOP: (us)	BEST	TYPICAL	WORST	SETUP(us)	
	0.7	0.7	0.8	1.7	(167 ns memory)
	1.2	1.2	1.3	2.0	(333 ns memory)

PROGRAM SIZE: (AP words)		
	13	(167 ns memory)
	18	(333 ns memory)

APAL CALL: JSR CVMAGS
SCRATCH: SP(0,2,4),DPX(0-1),DPY(0),FA.FM,MD
EXTERNALS: None