NORTH STAR ★ COMPUTERS, INC.
2547 Ninth Street • Berkeley, California 94710 • (415) 549-0858

North Star

FLOATING POINT BOARD FPB-A

Contents

Parts List 3
Warranty 5
Using the FPB 6
Theory of Operation 12
Assembly and Check-out 20
Drawings 24

Copyright, North Star Computers, Inc., 1978

REVISION 5

CAUTIONS

1. Correct this document from the errata before doing anything else.

2. Do NOT insert or remove the FPB from the computer while the power is turned on.

3. Do NOT insert or remove IC's from the board while the power is turned on.

4. Be sure the +5 volt regulators are generating +5 volt output voltages before installing any IC's.

5. Be careful to insert all IC's in correct positions and with correct orientation.

PARTS LIST

1	printed circuit board (5" x 10")
2	bolts, 6x32x1/2"
2	lock washers
2	nuts, 6x32
1	heat sink, 680-.5220
2	5 volt regulators, 7805 or 340T-5
1	crystal, 8MHz
33	16-pin low profile IC sockets
13	14-pin low profile IC sockets
1	24-pin low profile IC socket
1	Documentation package
1	BASIC manual
1	BASIC paper tape

Integrated Circuits		Schematic Label
1	74LS00	AN
1	74LS02	NR
1	74LS27	3A
1	74LS30	8N
1	74LS42	SD
1	74LS51	AOI
2	74LS74	SY,CH
5	74LS75	DIL,DIR,AL,BL,TL
1	74LS109	DS
2	74LS132	MN,ST
1	74LS136	XR
1	74LS151	BMX
1	74LS157	EN
2	74LS161	PCM,PCL
3	74LS169	RR,LP,CNT
1	74LS181	ALU
1	74LS258	AMX
1	7405	OC
1	74109	GR
2	74367 or 8097 or 8T97	RD,LD
1	74S00	FN
1	74S08	FA
4	74S189	LRAM,RRAM,DRAM,CRAM
11	6301 or 74S287 or 82S129	CCOP,BAMP,BALP,BCSP,MSCP,CONP ASRP,ACSP,STOP,BSRP,CADP

Capacitors

1	100uf electrolytic	C1
2	6.8uf tantalum	C2,C3
1	330pf dipped mica	C4
2	33pf dipped mica	C5,C7
1	100pf dipped mica	C6
15	.047 ceramic disc	*

Resistors

3	470 ohm	R1,R2,R5
2	2.2 K	R3,R6
1	1.2 K	R4

WARRANTY

North Star Computers, Inc. warrants the electrical and mechanical parts and workmanship of this product to be free of defects for a period of 90 days from date of purchase. If such defects occur, North Star Computers, Inc. will repair the defect at no cost to the purchaser. This warranty does not extend to defects resulting from improper use or assembly by purchaser, nor does it cover transportation to the factory. Also, the warranty is invalid if all instructions included in the accompanying documentation are not carefully followed. Should a unit returned for warranty repair be deemed by North Star Computers, Inc. to be defective due to purchaser's action, then a repair charge not to exceed $30 without purchaser's consent will be assessed. Any unit or part returned for warranty repair must be accompanied by a copy of the original sales receipt. This warranty is made in lieu of all other warranties, expressed or implied, and is limited to the repair or replacement of the product.

USING THE FLOATING POINT BOARD (FPB-A)

ADDRESS SELECTION

Before using the floating point board, the memory addresses to which the FPB responds must be selected with jumper wires. The jumpers determine the four most significant bits of the addresses. The remaining 12 bits are fixed on the board. The jumper area is located just above IC XR. There are four pads labeled 15, 14, 13, and 12 associated with the four most significant address bits. The two pads at the ends provide ground(G) and high logic level(H). All address bits that should be "ones" should be daisy chain jumpered to G and all "zero" bits should be jumpered to H. The following table shows the RESTART and DATAIN addresses in hexadecimal corresponding to the 16 possible jumper combinations.

15	14	13	12	RESTART	DATAIN
H	H	H	H	ØFF2	ØFF1
H	H	H	G	1FF2	1FF1
H	H	G	H	2FF2	2FF1
H	H	G	G	3FF2	3FF1
H	G	H	H	4FF2	4FF1
H	G	H	G	5FF2	5FF1
H	G	G	H	6FF2	6FF1
H	G	G	G	7FF2	7FF1
G	H	H	H	8FF2	8FF1
G	H	H	G	9FF2	9FF1
G	H	G	H	AFF2	AFF1
G	H	G	G	BFF2	BFF1
G	G	H	H	CFF2	CFF1
G	G	H	G	DFF2	DFF1
G	G	G	H	EFF2	EFF1
G	G	G	G	FFF2	FFF1

TO USE BASIC VERSION 5-FPB THE JUMPERS MUST BE WIRED TO USE ADDRESSES DFF2 AND DFF1. For proper operation no other memory in the computer can respond to addresses in the same block of sixteen as RESTART and DATAIN. For example, if RESTART and DATAIN are BFF2 and BFF1 then the FPB requires full use of addresses in the range BFFØ-BFFF.

PROGRAMMING

The 8Ø8Ø or Z8Ø program for causing the FPB to perform an arithmetic operation must perform the following steps in sequence.

1. Restart the FPB with a memory read of the RESTART address.

2. Send the command byte which specifies the precision and operation to be performed. The command byte is the first data byte (i.e. not the first byte of an instruction) read after the RESTART.

3. Send the N bytes of the right operand. These are the N data bytes read after the command byte. No other data bytes may be read during the transmission of arguments.

4. Send the N bytes of the left operand. These are next N data bytes read. The FPB immediately starts performing the specified operation after the Nth data byte is read.

5. Wait for and receive the result status byte. The status byte is read by performing a read to the DATAIN address. If the sign bit of this value is zero then the FPB is still computing the result value. If the sign bit is one then the read byte is the status byte of the result.

6. Read the N bytes of the result value. The next N reads of the DATAIN address after reading the status byte provide the N bytes of the result value. After the last result byte is received at least 1Ø microseconds must elapse before another RESTART can be done. The result bytes must be read even after an error or else the next operation will not perform correctly.

The follwing sample program demonstrates the efficient use of the FBP.

IMPORTANT NOTE: After power-on a dummy operation must be performed to initialize the FPB. Subsequent operations will then perform corectly. The following program will initialize the FPB.

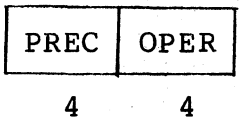
```
LDA RESTART
MVI A,2*16
LDA Ø
LDA Ø
LDA Ø
```

```
*SAMPLE USE OF THE NORTH STAR FPB
*FOR A DIVIDE OPERATION WITH 6 DIGIT PRECISION.
*IN THIS EXAMPLE ASSUME ARGUMENTS ARE IN MEMORY IN FORM:
*  MOST SIGNIFICANT BYTE (MSB) DIGIT PAIR
*  SUSEQUENT DIGIT PAIRS FOLLOW THE MSB
*  EXPONENT+SIGN BYTE FOLLOWS LSB DIGIT PAIR.
*POINTERS ADDRESS THE EXPONENT+SIGN BYTE.
*BC HAS LEFT ARG POINTER.
*DE HAS RIGHT ARG POINTER.
*HL HAS RESULT POINTER.

*THE FPB RECEIVES ITS ARGUMENTS BY "PEEKING" AT THE 8080 BUS
*WHEN THE ARGUMENT VALUES ARE LOADED TO ACCUMULATOR.

   FDIV LDA RSTRT              "WAKE UP" FPB
      MVI A,6*16+DIVOP         SPECIFY PRECISION AND OPERATION CODE
      LDAX D                   EXPONENT+SIGN BYTE OF RIGHT ARG
      DCX D                    ADVANCE POINTER TO NEXT BYTE
      LDAX D                   LEAST SIGNIFICANT DIGITS OF RIGHT ARG
      DCX D                    ADVANCE POINTER TO NEXT BYTE
      LDAX D
      DCX D
      LDAX D                   MOST SIGNIFICANT DIGITS OF RIGHT ARG
      LDAX B                   EXPONENT+SIGN BYTE OF LEFT ARG
      DCX B
      LDAX B                   LEAST SIGNIFICANT DIGITS OF LEFT ARG
      DCX B
      LDAX B
      DCX B
      LDAX B                   MOST SIGNIFICANT DIGITS OF LEFT  ARG
   *  NOW THE FLOATING POINT BOARD IS PERFORMING THE OPERATION
      LXI D,DATAIN             RECEIVE DATA ADDRESS FOR FPB
   FDIV1 LDAX D                WAIT LOOP FOR COMPLETION SIGNAL
      ORA A                    SIGN BIT "1" MEANS FPB IS DONE
      JP FDIV1                 LOOP IF SIGN BIT IS STILL "0"
      ANI EBITS                CHECK FOR ERROR, TESTED AT END
      LDAX D                   EXPONENT+SIGN OF RESULT
      MOV M,A                  STORE EXPONENT+SIGN OF RESULT
      DCX H                    ADVANCE POINTER
      LDAX D                   LEAST SIGNIFICANT DIGITS OF RESULT
      MOV M,A
      DCX H
      LDAX D
      MOV M,A
      DCX H
      LDAX D                   MOST SIGNIFICANT DIGITS OF RESULT
      MOV M,A                  STORE IT
      RZ                       RETURN IF NO ERROR WAS DETECTED
      JMP ERROR                GO REPORT ERROR
```

COMMAND BYTE FORMAT



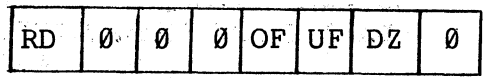
PREC specifies the precision of the operand and result. It must be one of the following values.

Digits of Precision	PREC(hexadecimal)
2	2
4	4
6	6
8	8
10	A
12	C
14	E

OPER specifies the operation to be performed. It must be one of the following values.

Operation	OPER
Add	1
Subtract	2
Multiply	3
Divide	4

STATUS BYTE FORMAT



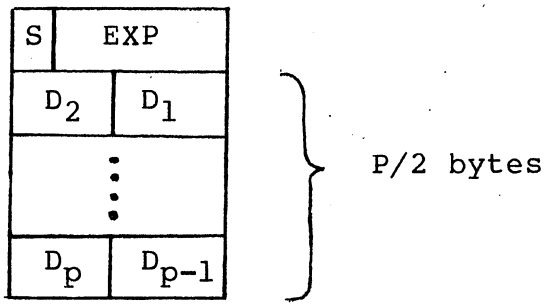
RD if 1 indicates the operation is done and that the next N bytes read from the FPB will be the result. The other bits are valid only when RD is 1.

OF if 1 indicates an overflow error, that is the magnitude of the result value would be too large to be represented.

UF if 1 indicates an underflow error, that is the magnitude of the result value would be too smal to be represented.

DZ if 1 indicates a divide by zero error

FLOATING POINT VALUE FORMAT



The first byte of a floating point value passed to the FPB contains the sign bit(S) and exponent(EXP). S=Ø indicates a positive value and S=1 indicates a negative value. The exponent is represented in excess 64 binary. The exponent represented is to the base ten. Some examples of exponent values follow.

Exponent	EXP(hexadecimal)
-63	Ø1
-1	3F
Ø	4Ø
1	41
1Ø	4A
63	7F

The value zero is represented by a first byte with all zero bits (i.e. S=Ø and EXP=Ø). If the precision is P digits then the fraction part is represented by P/2 bytes following the first byte. Two binary encoded decimal digits are packed per byte. The decimal point is assumed to be to the left of the most significant digit. D_1 is the least significant digit and D_p is the most significant digit. All argument values must be normalized (i.e. D_p non-zero). The FPB will always return a normalized value.

Examples in six digit precision:

Value	Representation(hexadecimal)
Ø	ØØ ØØ ØØ ØØ
1	41 ØØ ØØ 1Ø
$.123456 \times 1Ø^{8}$	48 56 34 12
$-.987654 \times 1Ø^{-9}$	B7 54 76 98

FPB-A BUS INTERFACE DESCRIPTION

The FPB-A is compatible with the S-100 bus, i.e. the bus used by ALTAIR and IMSAI computers. The following signals on the bus are used by the FPB-A. All signals are positive, high true, TTL logic levels.

DI_7-DI_0	The 8 input data lines. The FPB reads operand bytes from these lines and tri-states result bytes onto these lines.
A_{15}-A_0	The 16 address lines. The FPB decodes these lines to recognize the RESTART and DATAIN addresses.
SMEMR	Status line that indicates to the FPB that a bus cycle is memory read.
SM1	Status line that indicates to the FPB whether a memory read is a first byte of an instruction fetch or a data byte fetch.
PDBIN	Timing signal used by FPB to strobe result bytes onto DI bus and to indicate operand bytes have been strobed onto the DI bus by a memory module.
PSYNC	Used by the FPB to recognize the beginning of a memory cycle.
Ø2	Used by the FPB to synchronize the memory reference control logic with the CPU.
PRDY XRDY	The logical AND of these two signals indicates that read data is stable on the DI bus. The FPB uses these signals when reading operand bytes.
PHLDA	Indicates that the memory reference is a DMA cycle. The FPB ignores DMA cycles while reading operand bytes.

THEORY OF OPERATION

HARDWARE

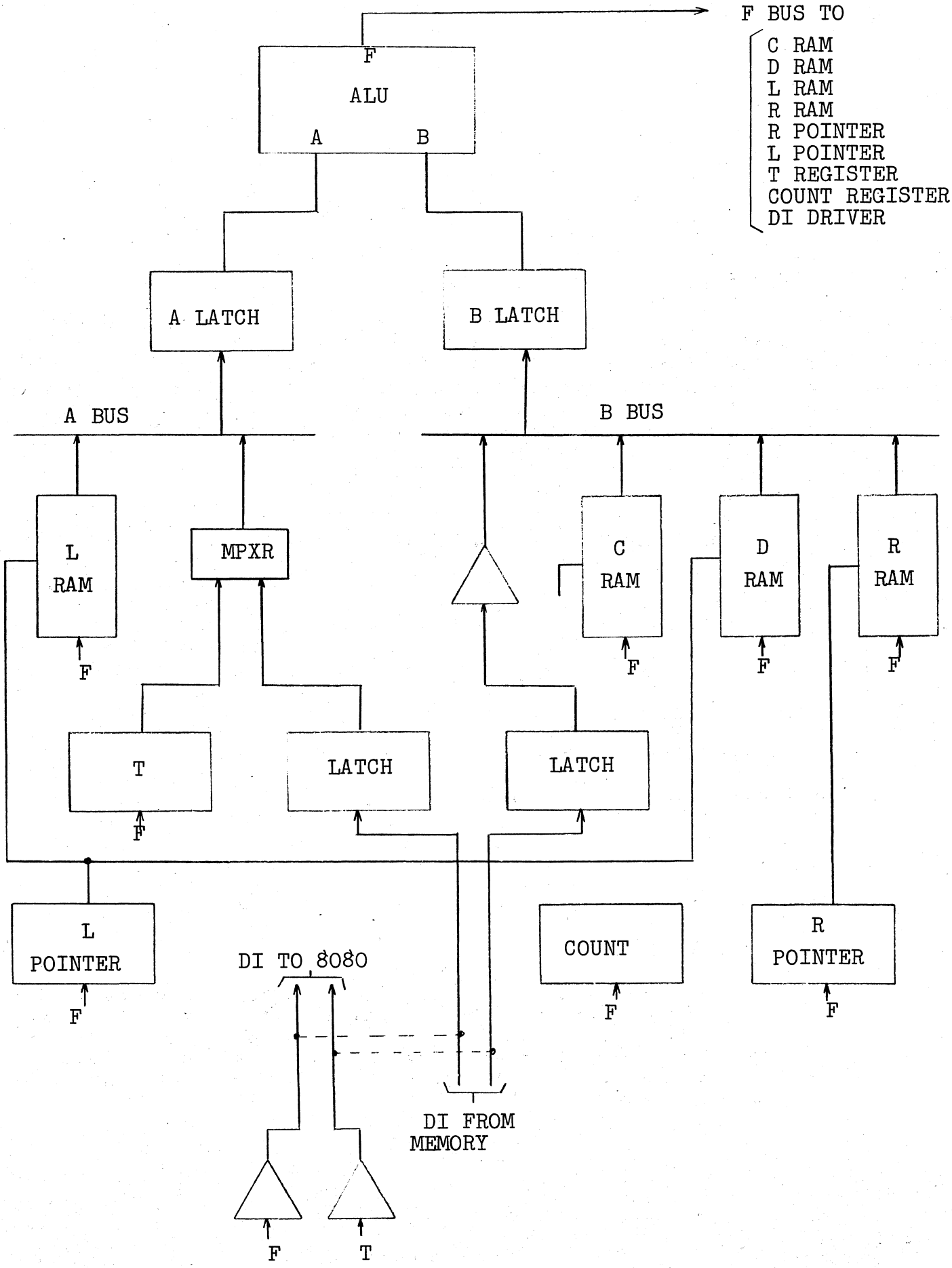
The FPB is a microprogram controlled processor designed specifically to perfrom high speed decimal floating point arithmetic operations. The unit is implemented entirely from medium and small scale TTL integrated circuits and PROM memory. All data paths in the microprocessor are 4 bits wide thus enabling processing one decimal digit at a time. The FPB is divided into four functional sections:

1. Data Paths

The following diagram shows the microprocessor data paths. There are three main buses in the data paths: A, B, and F. During each instruction cycle of the microprocessor, values are gated onto the A and B buses which are latched at the end of the first half of the microcycle by the A and B latches. The latched A and B values serve as inputs to the ALU. The ALU output goes on the F-bus which is clocked into one of the RAM's or counters at the end of the microcycle. The T-LATCH is always loaded from the F-bus at the end of every microcycle. There are four 4 x 16 RAM's. L-RAM and R-RAM generally hold the fraction part of the left and right floating point operands and the D-RAM holds the result. C-RAM is used to hold the exponents, various constants and temporary results required during the calculation. There are three counters. L-PTR and R-PTR are used to address the L, R, and D RAM's. CNT is used to count the number of iterations through a program loop. The F-bus and T-LATCH together are used to gate an 8 bit result byte onto the DI bus. The two data input latches are used to catch operand bytes off the DI bus.

2. Clock Circuit

The clock circuit is a crystal controlled oscilator(IC's FN, FA). The generated 8 MHz signal is used to generate three clock signals. LCLK/ is a free running 8 MHz signal. DST-CLK and WRT-T are timing signals that operate at 4 MHz. These signals do not occur while the microprocessor is waiting for a bus memory cycle.


F BUS TO
C RAM
D RAM
L RAM
R RAM
R POINTER
L POINTER
T REGISTER
COUNT REGISTER
DI DRIVER
F
ALU
A
B
A LATCH
B LATCH
A BUS
B BUS
L RAM
MPXR
C RAM
D RAM
R RAM
T
LATCH
LATCH
L POINTER
COUNT
R POINTER
DI TO 8080
DI FROM MEMORY
T

3. Control Logic

The control logic is responsible for:

a. sequencing the micro program counter. At the end of every microcycle the p-counter is either loaded causing a branch, incremented to address the next instruction, or left unchanged during loops. The branch multiplexor(BMX) determines which of seven conditions will cause a branch. The P-CON signal out of the condition PROM(CCOP) determines if the p-counter will increment.

b. interfacing and decoding bus control lines. The ARGS-AV signal indicates when an operand byte is on the DI bus and is used to gate the DI latches. The GO signal indicates that a new floating point computation is to begin. After being sychronized to the microprocessor clock it is used to force the microprogram counter to zero. The BUSY RESPONSE and DONE RESPONSE signals indicate that a status or result byte should be gated onto the DI bus.

c. synchronizing the microprocessor with memory references from the CPU. The bus synchronizing logic(IC's GR, CH, AOI, SY, DS) cause the microprocessor to hang if an expected memory read reference on the bus has not yet occurred. The SELA8 and RES8 microprogram conditions indicate that an argument byte is required by the microprogrm or that the CPU should take a result byte, respectively. The SY flip-flops catch the memory conditions with the CPU clock. HSY is the or of these conditions synchronized to the microprocessor clock. The signal is then used to generate GATE. GATE is used to enable DST-CLK and WRT-T and to enable A-LATCH and B-LATCH. GATE is kept high by the synchronizing logic to hang the microprocessor during the first half of a microcycle.

4. Microprogram Storage

The microprogram is stored in ten bipolar PROM's organized 4 bits by 256 words. A hexadecimal listing of the PROM contents follows. The meaning of each PROM is now given.

a. Branch Address PROM (BAMP). Most significant 4 bits of the branch address.

b. Branch Address PROM (BALP). Least significant 4 bits of the branch address.

c. Branch Condition PROM (BCSP).
bit 3 indicates arithmetic should be done in decimal mode.
bits 2-Ø indicate one of eight branch conditions.

d. Misc. PROM (MSCP).
   bit 3 enables clocking of the carry flip-flop at the end of a microcycle.
   bit 2 indicates that a result byte in the T-LATCH and on the F-bus is ready for the CPU.
   bit 1 indicates whether L-PTR and R-PTR should count up or down.
   bit Ø enables clocking of the three counters at the end of the microcycle.

e. Const PROM (CONP). Contains the 4 bit constant to gate onto the A-bus.

f. A Source PROM (ASRP). Specifies what to gate onto the A-bus.

g. ALU Control PROM (ACSP). Specifies the ALU function bits.

i. Store PROM (STOP).
   bit 3 is the ALU mode bit
   bits 2-Ø specify which of seven destinations will get loaded from the F-bus at the end of the microcycle.

j. B Source PROM (BSRP). Specifies what to gate onto the B-bus.

k. C Address PROM (CADP). Specifies which word in the C-RAM to address.

CONST PROM
A010870F 0F08770F 0C0FEDB0 FFF5BEF4 F7FFFF00 000007F0 0FDFEFE0 AFF0EFFF
4FA00FFF 7B77B70F 0F0F0E00 EDFEF005 F7FF0F5F FE00F7FF 0FFF0FFF FFF0EF0F
F00FF000 0E001EFD 000DF00F F00000FF 7F0F7F0F F0000FFF D7FFFD0F 0FEFF000
FFFF00F0 EEFF0DF4 FA0000F0 00F7F0FF 00FFFF7F 00FFF0FF FFFFFFFF FFFFFFFF

CADDR PROM
E746514F FF320F1F 47FF777F FFFF8CB9 F15623E6 3B52AAA4 BBFFF14F FFFC32FC
FFF4CF23 22FFFFA4 B6352BCB BFFF4EFF F04C4FF4 FC32FFFF 4FFFFF23 5647CFCF
4FECFF74 CFF77CC4 9BECF7C6 579E3223 210FF023 2C4FFF56 FF23C47C FCF4FEFF
FC4FFEF7 C7FFC7CF FF657810 32FF0232 C4FFFFF1 0565F4FF F0000000 0000000F

STORE PROM
FFFFFF1D EEFFF8FB 18AA8889 EACFFFFF 0F888808 0F80F888 09DEB814 408777CB
04483888 88888888 08F8F0FF 1BAD9044 0F9B0449 87778888 1880CC88 888F3A7D
122B8478 438FF7B1 02230F7A A2207788 88F888F8 83188088 8888F0FB CB49D022
0B9D2287 7FB87FB0 44AA228F 77888F88 31880888 8F88D188 00000000 00000000

BSOURCE PROM
66F6FF66 6F6FFF66 666F6666 66666666 66666666 66666666 6666666A C6C66666
CCC66C66 66666666 66666666 666666AC 66666CC6 C6666666 66666F66 66666666
6A66CC66 6C666666 66666666 66666666 66666666 66666666 66666666 66C666A6
6666A666 666C666C CC666666 66666666 66CC6666 666666AA 6EEEEEEE EEEEEEE6

ASOURCE PROM
CBCBCCCC BCBCCCAC CCBCCCCC CCCCCCCC CCCCCCCC AACAACCC ACCCCCC0 CCCCCCCC
CCCCACCC CCCCCCCC ACACACAC CCCCCC0C CCCCCCCC CCCCCCCC CCCCBCCC CCCACCCC
C00CCCCC AC0CCCCC 000CCCCC C00C00CC CCACCCAC CCC00CCC CCCCCCAC CCCCCC00
CCCC00CC CCCCCCCC CCCC00CA 00CCCACC CCCCCCCC AACCCCCC C8888888 8888888C

ALU PROM
FFBFBB9F FAFBBB6F 96FA666F FFFFFFFF F6AAAA96 9F69FBA6 9AFFFA99 99A999FA
99969AAA BBFFFF6A 9AFAF9F6 9FFFA969 96AA969A A999FFFF 9FF9FAAA AAAF9F9F
999AA996 99FFF9A9 99999F9A A99999AA BAFFFAEA A99FF9AA FFAAF9FA FA9AF969
9AAF99F9 9FFA9FA9 996699AF 99FFAEAA 99AA9FFA 6EAAF9AA 90000000 0000000F

BRADDR MSBITS PROM
00000000 00000001 171122A1 11111112 222A2E22 222224A3 A3333533 33333334
44444344 D44646E5 E5555555 55535565 66666666 36666667 77717776 76777788
88888778 88888889 99999999 999999A4 4AAAAAAA AAAAA1BB B6B6BBBB BBBBBBCB
BCCCCCCC BCCCCCCD DDDDDDDD D4DDDDDE EEEE1EEE EEEEEEEE 10000000 00000007

BRADDR LSBITS PROM
12345678 98BCDEF0 143221E8 9A8CDEF0 02436578 9ABCDE30 32335C79 7AFCDEF0
63145B8C ACBEDE50 52345678 9AA5DE0E 72345758 FAB8DEF0 1217548C ACBCDE30
13146FC8 9ACEDEF0 33147638 9ABCDE0C 92345678 9ABCB722 1E4C5678 9ABCDE0E
92347479 7ABEDEF2 20345678 96BCDEF0 12327678 9ABCDEFE 70000000 00000008

MISC PROM
BBBBBBBB 88BBBBBB BB88BBBB BB8BBBBB BBBBBBB3 B33BBBBB BBB8BBB8 0A83BBBB
8808B8BB BBBFBFBB BBBBBBBB BB8BBB80 BBBBB80A B3BABFBF B8CB88BB BBBBBBBB
B80BB32B 3BBBBBB3 880B8B38 A8AB8BBB BBBBFBBB F3B8CBBB BFBBBBBB BBBBBB80
BBBB80B3 BBBBBBB8 808A8ABB 8BBFBBBF 3B8CBBFB BBBFBB8C BBBBBBBB BBBBBBBB

BRCOND PROM
77777777 73777777 70730007 77377777 77101077 77777117 57737179 37077777
68777110 01777717 67777777 77377793 27777937 17737777 77377310 10777777
79370717 77077777 68773777 77777710 17777777 77773711 77107777 77777793
27779370 77707776 87777777 77777777 77737777 77777773 70000000 00000007

MICROPROGRAM

The FPB microprogram contains 256 instructions. The function of the program is now described. The common start up sequence starts at address zero and is executed when a RESTART memory read is performed by the CPU.

Common Start Up Sequence

1. Receive the command byte and save the precision and operation code.

2. Receive the right operand and save in the R-RAM

3. Receive the left operand and save in the L-RAM.

4. Dispatch to one of the four arithmetic routines based on the operation code.

ADD Routine

1. Test for zero args. If the left argument is zero then return the right argument as the result. If the right argument is zero then return the left argument as the result.

2. Scaling. If the exponent difference is greater than the precision, then the argument with larger exponent is returned as the result. Otherwise, the argument with the smaller exponent is scaled right by the exponent difference. For example:

$$.1234 \times 10^{3} + .5678 \times 10^{6}$$
after scaling becomes
$$.00012 \times 10^{6} + .5678 \times 10^{6}$$

   Notice that an extra digit of precision is maintained internally.

3. Perform arithmetic. If the sign of the arguments are the same then the argument fraction parts are added. If the precision is P then P+2 digits are added, one lower order rounding digit and one higher order digit in case of overflow. For example if P=4 then the result of the previous example is 0.56792. if the sign of the arguments are different then the fraction part of the right argument is subtracted from the fraction part of the left argument. An extra low order digit is included in the subtraction. If the right argument fraction part is larger than the left argument fraction part then the subtraction result must be complemented. For example:

$$.1234 - .5678 = .55560$$
and after complementing becomes
$$.44440$$

4. Normalize result. If an add was performed then the result is normalized right by one if the add caused an overflow. If a subtract was performed then the result must be normalized left for each high order zero digit in the result. For example, the following result must be normalized by two digits:

$$.5567 \times 10^{6} - .5511 \times 10^{6} = 0.00560 \times 10^{6}$$

and after normalizing becomes

$$.5600 \times 10^{4}$$

If the result of the subtraction is all zero digits then a zero result value is returned.

5. Rounding. If the extra low order digit is greater than or equal to five then one is added to the fraction part of the result. If rounding causes an overflow then the result must be normalized right by one digit. For example,

$$.99999 \times 10^{6}$$

must be rounded to

$$1.0000 \times 10^{6}$$

and now must be normalized to

$$.1000 \times 10^{7}$$

6. Test and return result. If the operations generated a result with too large or small an exponent then a zero result value is returned and the overflow or underflow error flag is returned in the status byte. Otherwise, the result is returned without an error flag in the status byte.

SUBTRACT Routine

1. The sign of the right argument is complemented and then the ADD routine is performed.

MULTIPLY Routine

1. Zero test. If either argument is zero then a zero result is returned.

2. Perform multiply. The right argument is the multiplicand and the left argument is the multiplier. First, the result is set to zero. Then, the muliply is performed by doing a repeated add of the multiplicand for each digit of the multiplier. The add of the multiplicand is done to P+2 digit accuracy thus maintaining a rounding and overflow digit. The multiplicand is added D times where D is the value of the next digit of the multiplier. After each repeated add of the multiplicand, the result is shifted right. The multiplier is processed from least significant to most significant digit.

3. Normalize. If the result has a zero overflow digit, then the result must be normalized left by one digit.

4. Rounding. If the rounding digit is greater than or equal to five then one is added to the fraction part of the result.

5. Compute exponent. The exponents of the two arguments are added to get the result exponent.

6. Test and return result. The result is tested and returned as in the ADD routine.

DIVIDE Routine

1. Test for zero. If the right arg is zero then a zero value is returned and the divide by zero error flag is set in the returned status byte. Otherwise, if the left arg is zero then a zero value is returned.

2. Perform divide. Division is done by performing repeated subtracts of the divisor from the dividend in a loop. The reapeat loop is done once for each digit of the dividend. The dividend is processed from most significant to least significant digit. After each digit is processed the dividend is shifted left. The repeated subtracts of the divisor continue until the dividend becomes negative, then the divisor is added back to make the dividend positive again. The successive digits of the result are the number of times the subtaction must be performed.

3. Normalize. If the first digit of the result is zero then then result must be normalized left by one digit.

4. Round. If the rounding digit of the result is greater than or equal to five then one is added to the result value fraction part.

5. Compute exponent. The result exponent is computed by subtracting the right arg exponent from the left arg exponent.

6. Test result and return. The result is tested and returned as in the ADD routine.

ASSEMBLY AND CHECK-OUT INSTRUCTIONS

SOLDERING TIPS

For best results use a 15 watt soldering iron or an iron with a temperature regulated tip. The tip should be no wider than the solder pads on the printed circuit board. Use only a fine gauge, rosin core solder. When soldering, keep the soldering iron tip on the pad just long enough for the solder to completely flow. If the solder does not draw up the wire then more solder is required. Use less solder if it is overflowing the pad. If the solidified joint is not shiny it may be a cold solder joint and should be remelted. The soldering iron tip should be cleaned frequently by wiping on a damp sponge.

ASSEMBLY

For best results, assemble the FPB using the following steps. Between each step test that "ground" and "+5" are not shorted since it is very difficult to debug a short after all the components are installed.

NOTE: Orient the board with the edge connector toward you and the heat sink area to the left. The component side is now up. The silk screen legend is on the component side. The solder side has the larger IC pads.

1. Install and solder the 33 sixteen-pin IC sockets. Orient them as shown on the layout sheet and on the silk screen legend.

   NOTE: IC sockets can be installed by first stuffing them into the printed circuit board, then placing another flat board over the IC sockets and finally turning over this sandwich for soldering.

2. Similarly, install the 13 fourteen-pin IC sockets.

3. Now install and solder the single twenty-four-pin IC socket.

4. Install and solder the six resistors in the locations indicated by the silk screen legend.

R1	470 ohm	yellow-violet-brown	.4" spacing
R2	470 ohm	yellow-violet-brown	.4" spacing
R3	2.2 K	red-red-red	.6" spacing
R4	1.2 K	brown-red-red	.6" spacing
R5	470 ohm	yellow-violet-brown	.5" spacing
R6	2.2 K	red-red-red	.55"spacing

NOTE: Save some of the snipped leads for use as jumpers.

5. Install and solder the 22 capacitors in the locations indicated by the silk screen legend.

C1	100uf	electrolytic	"+" toward edge of board
C2	6.8uf	tantalum	"+" of C2 and C3
C3	6.8uf	tantalum	toward each other
C4	330pf	dipped mica	
C5	33pf	dipped mica	
C6	100pf	dipped mica	
C7	33pf	dipped mica	

The remaining fifteen .047uf ceramic disc capacitors are bypass capacitors and should be installed in the unlabeled oval capacitor locations on the silk screen legend (marked with asterisks on the layout drawing).

6. Install and solder the crystal. In addition to the two leads the top of the crystal case should be soldered flat against the PC board by soldering a piece of clipped resistor lead between the crystal case and the provided solder pad.

7. Place the heat sink on the PC board such that the pads for the regulators show.

8. For both regulators, bend down the three regulator leads 90 degrees such that the leads go into the correct holes while the bolt holes line up. But don't solder yet.

9. Install the heat sink hardware so that the following sequence results from bottom to top: 6x32 bolt, PC board, heat sink, regulator, lock washer, and nut. Tighten the bolts so that the regulator leads are well separated from the heat sink and the heat sink is 1/8" away from the edge of the board for clearance with card guides. Now solder the regulator leads.

NOTE: Heat sink grease may be used though it is not generally needed. Don't tighten the bolts too tight to avoid cracking the PC board.

The board is now completely assembled except for inserting the IC's. Procede with the check-out procedure.

CHECK-OUT PROCEDURE

The following procedure should be followed for systematically checking-out the FPB. An oscilliscope is required for some steps in this procedure.

NOTE: It may be desireable to solder a piece of wire into the ground pad to the left of chip FA for clipping the scope ground.

DO NOT INSERT IC'S WITH THE POWER ON.

1. Check for +5 volts at the "+" side of each tantalum capacitor.

2. Install IC's FN and FA. Check for an 8MHz square wave at FN-6 (i.e. pin 6 of IC FN).

3. Install IC's PCM and PCL. Check that they count with a 2 microsecond period at PCL-11 and a 32 microsecond period at PCM-11.

4. Install the ten PROM's in the top row(note they are oriented upside down) and the IC AN. The contents of each PROM should now be sequencing at pins 9, 10, 11, and 12. This provides a way to verify the contents of PROM, should this be required.

5. Install IC's AL and ALU. There should be activity on ALU pins 2, 9, 10, 11, 13, 19, 21, and 23.

6. Install IC GR. Where previously there was an 8MHz signal at FN-6, now there should be 4MHz negative pulses with a 25% duty factor.

7. Install the address selection jumpers in the area between IC's XR and CNT. See the ADDRESS SELECTION description in the USING THE FLOATING POINT BOARD section for the details.

8. Install the following IC's: bottom row SY, NR, ST, OC, XR, 8N, next row DS, CH, 3A, MN, next row AOI. Now check the address comparison logic by referencing the jumper selected addresses using the front panel or using a program. When the RESTART address is referenced MN-8 should go low. When the DATAIN address is referenced MN-6 should go low.

9. Install all the remaining IC's. Now run the following test program.

```
Address  Hex          Symbolic

1000     3A F2 XF     LDA RESTART       X depends on address jumpers
1003     C3 00 10     JMP 1000
```

The following signals should now be observed.

a. "GO" low true pulses at GR-11

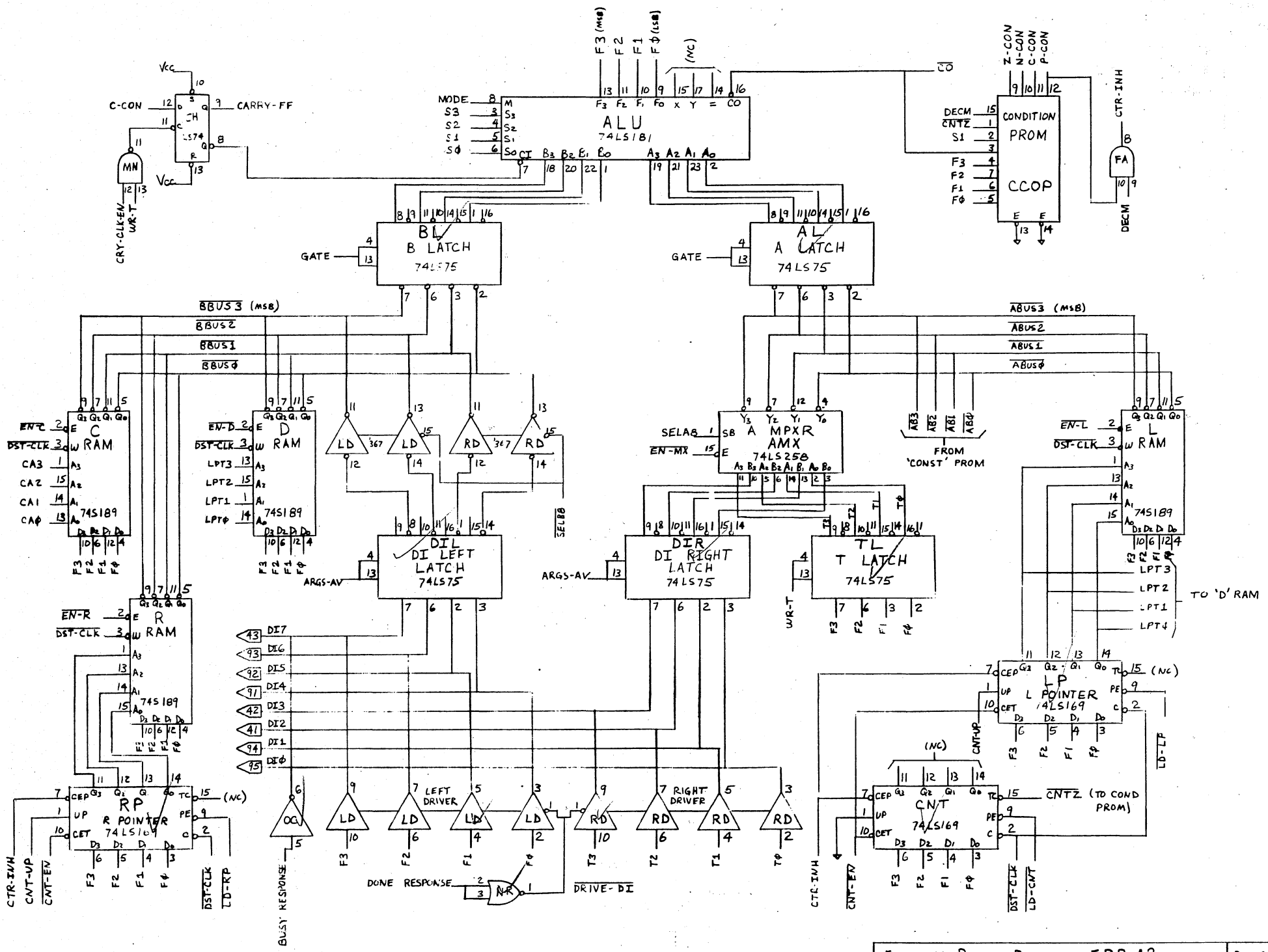
b. "RESTART" low true pulses at GR-9

c. bursts of clock pulses at FN-6 (microinstructions being executed)

d. pulses at SY-9 indicating the microcode waiting for an argument

e. activity at ALU pins 1,2,9,10,11,13,18,19,20,21,22,23

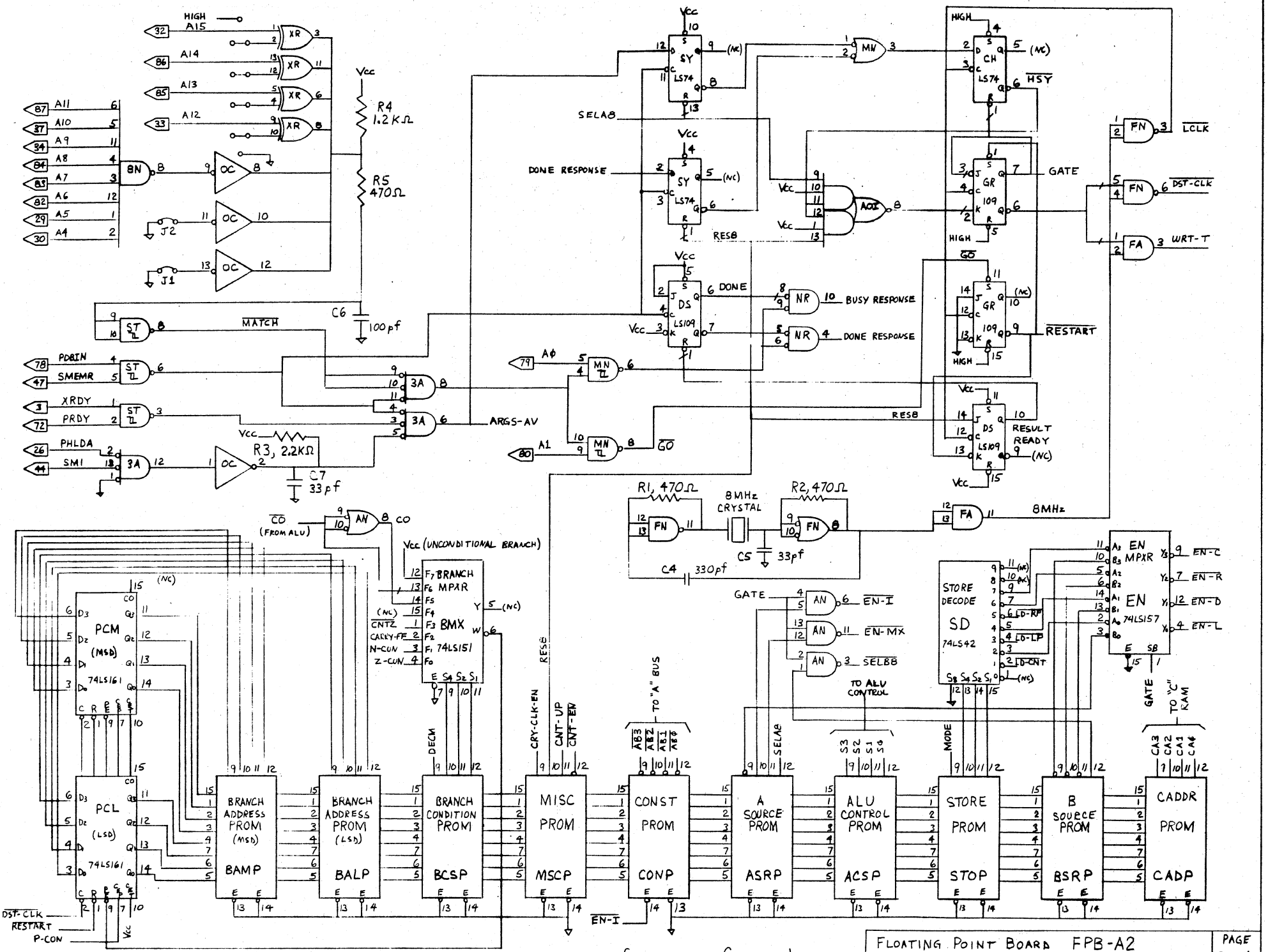
10. Run the add test program shown on page 4 of the drawings and verify that the results stored in memory bytes 5F5A, 5F5B, 5F5C, and 5F5D are correct.



SCHEMATIC — DATA PATHS

FLOATING POINT BOARD FPB-A2

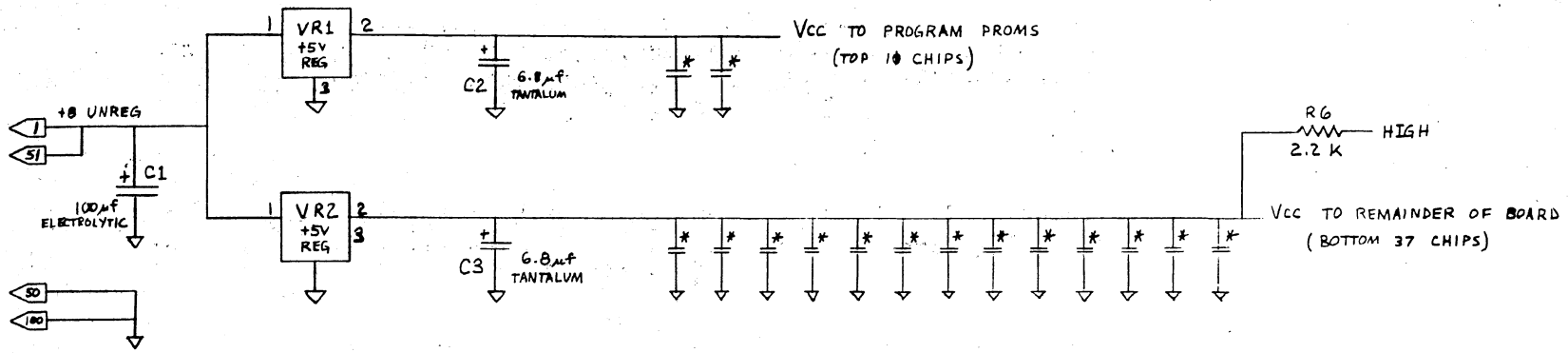
© NORTH STAR COMPUTERS, INC 1976

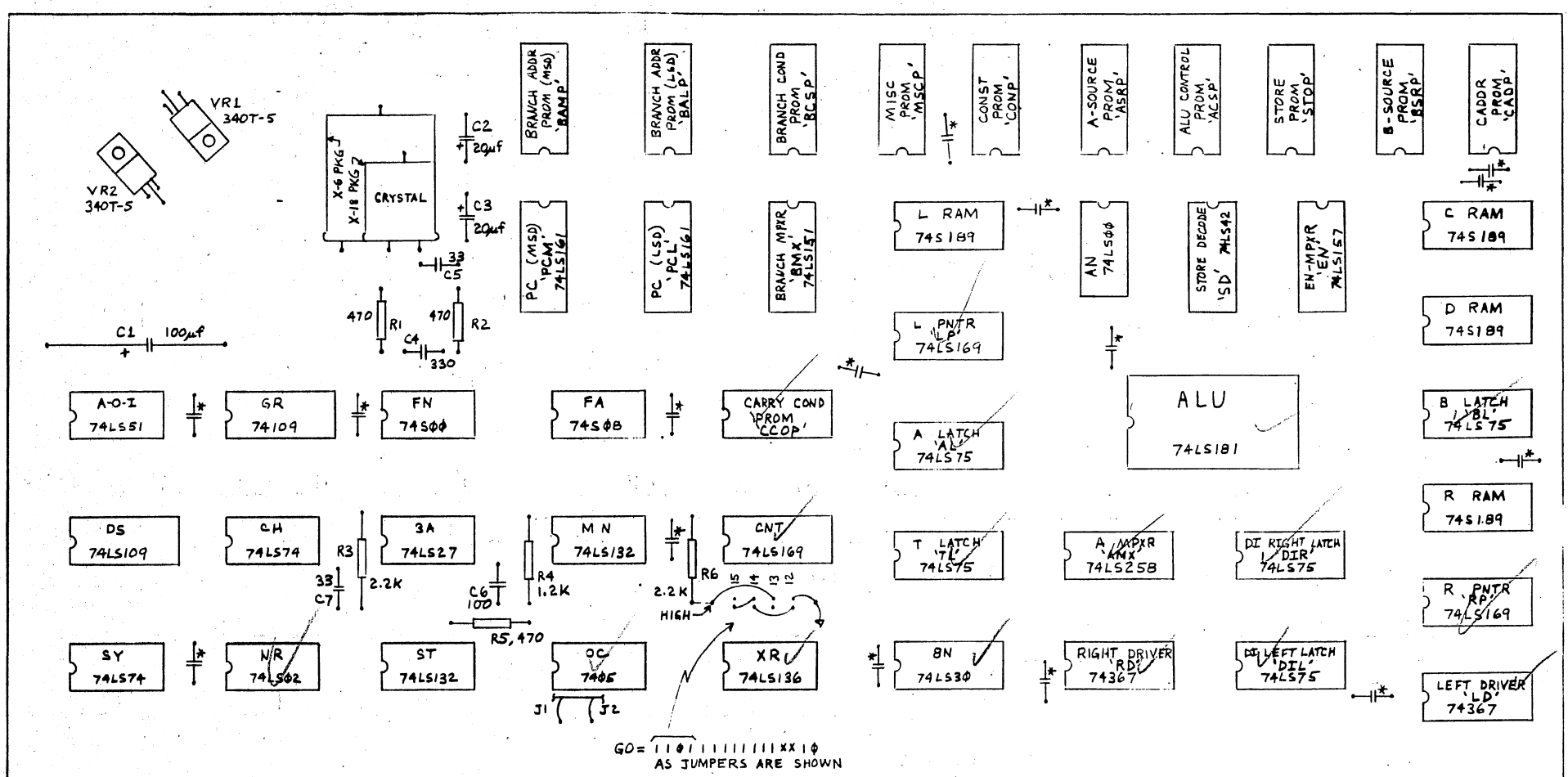


SCHEMATIC — CONTROL LOGIC

FLOATING POINT BOARD FPB-A2

© NORTH STAR COMPUTERS, INC 1976

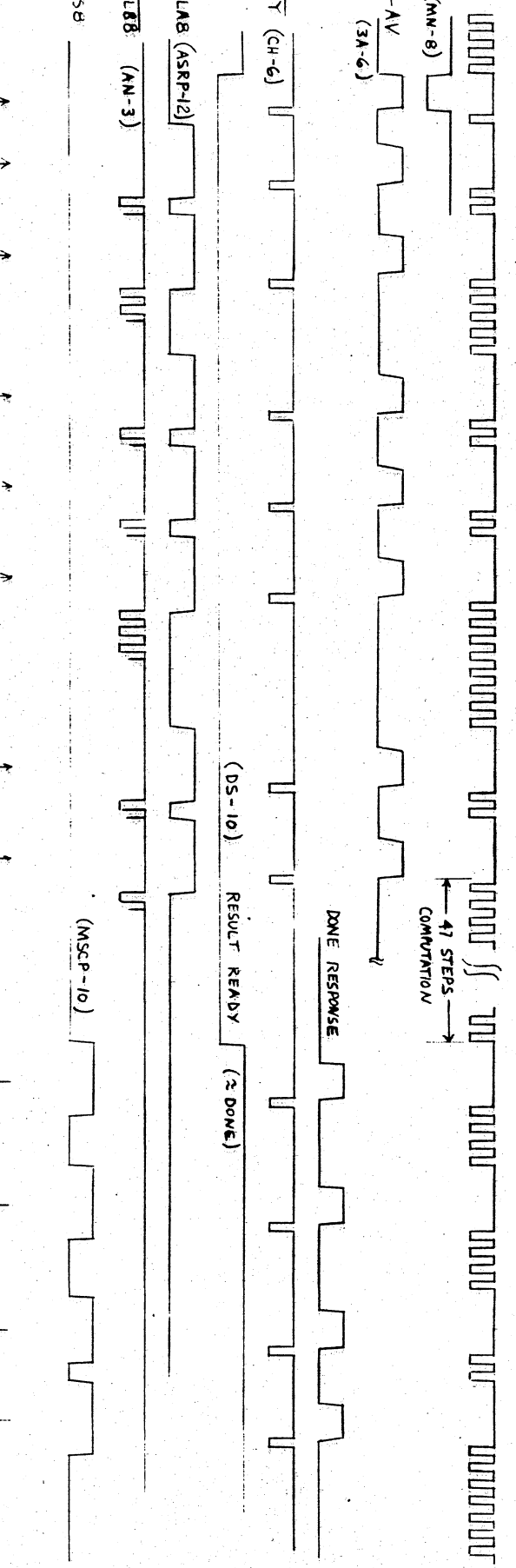




BOARD LAYOUT, REGULATOR SCHEM.

```
0000            *SIMPLE EXAMPLE: 98.76 + 4.321 = 103.1
0000
0000            FPGO  EQU 0DFF2H
0000            FPRES EQU 0DFF1H
0000
0000                  ORG 5F00H
5F00 214E5F     GO    LXI H,DATA      INITIALIZE FPB
5F03 3AF2DF           LDA FPGO        START INITIALIZE
5F06 7E               MOV A,M         COMMAND AND PRECISION
5F07 23               INX H
5F08 7E               MOV A,M         EXPONENT
5F09 23               INX H
5F0A 7E               MOV A,M         DIGITS
5F0B 23               INX H
5F0C 7E               MOV A,M         EXPONENT
5F0D 23               INX H
5F0E 7E               MOV A,M         DIGITS
5F0F 23               INX H
5F10 3E04             MVI A,4         SETUP DELAY LOOP
5F12 3D         IW    DCR A           DELAY LOOP
5F13 C2125F           JNZ IW
5F16 79               MOV A,C         PREVIOUS FLAGS TO A-REG.
5F17 D300             OUT 0           SCOPE SYNC ON "SOUT"
5F19 3AF2DF           LDA FPGO        START ADD OPERATION
5F1C 7E               MOV A,M         COMMAND AND PRECISION
5F1D 23               INX H
5F1E 7E               MOV A,M         RIGHT EXPONENT
5F1F 23               INX H
5F20 7E               MOV A,M         DIGITS
5F21 23               INX H
5F22 7E               MOV A,M         DIGITS
5F23 23               INX H
5F24 7E               MOV A,M         LEFT EXPONENT
5F25 23               INX H
5F26 7E               MOV A,M         DIGITS
5F27 23               INX H
5F28 7E               MOV A,M         DIGITS
5F29 23               INX H
5F2A 3E06             MVI A,6         SETUP DELAY LOOP
5F2C 3D         CW    DCR A           DELAY LOOP
5F2D C22C5F           JNZ CW
5F30 3AF1DF           LDA FPRES       RESULT FLAGS
5F33 77               MOV M,A
5F34 23               INX H
5F35 4F               MOV C,A         RESULT FLAGS TO C-REG.
5F36 3AF1DF           LDA FPRES       RESULT EXPONENT
5F39 77               MOV M,A
5F3A 23               INX H
5F3B 3AF1DF           LDA FPRES       RESULT DIGITS
5F3E 77               MOV M,A
5F3F 23               INX H
5F40 3AF1DF           LDA  FPRES      RESULT DIGITS
5F43 77               MOV M,A
5F44 23               INX H

5F45 3E04             MVI A,4         SETUP DELAY LOOP
5F47 3D         TW    DCR A           DELAY LOOP
5F48 C2475F           JNZ TW
5F4B C3005F           JMP GO
5F4E
5F4E 20         DATA  DB 20H          INITIALIZE COMMAND
5F4F 00               DB 00           RIGHT EXPONENT
5F50 00               DB 00           DIGITS
5F51 00               DB 00           LEFT EXPONENT
5F52 00               DB 00           DIGITS
5F53 41               DB 41H          COMMAND, 4 DIGIT ADD
5F54 41               DB 41H          RIGHT EXPONENT
5F55 21               DB 21H          DIGITS
5F56 43               DB 43H          DIGITS
5F57 42               DB 42H          LEFT EXPONENT
5F58 76               DB 76H          DIGITS
5F59 98               DB 98H          DIGITS
5F5A 00               DB 00           SHOULD GET 80H
5F5B 00               DB 00           SHOULD GET 43H
5F5C 00               DB 00           SHOULD GET 31H
5F5D 00               DB 00           SHOULD GET 10H
```



TIMING CHART

SIMPLE EXAMPLE

98.76 + 4.321 = 103.1

FLOATING POINT BOARD FPB-A2

©NORTH STAR COMPUTERS, INC. 1976