VLBA Correlator Memo No. 72

VLBA CORRELATOR HARDWARE DESIGN

(860731)

Ray Escoffier and Joe Greenberg

July 30, 1986

I Introduction

This memo will describe the present state of the design of parts of the VLBA correlator. As in VLBA correlator memo 71, information presented here is somewhat preliminary because of the evolving nature of the present design.

II What's new

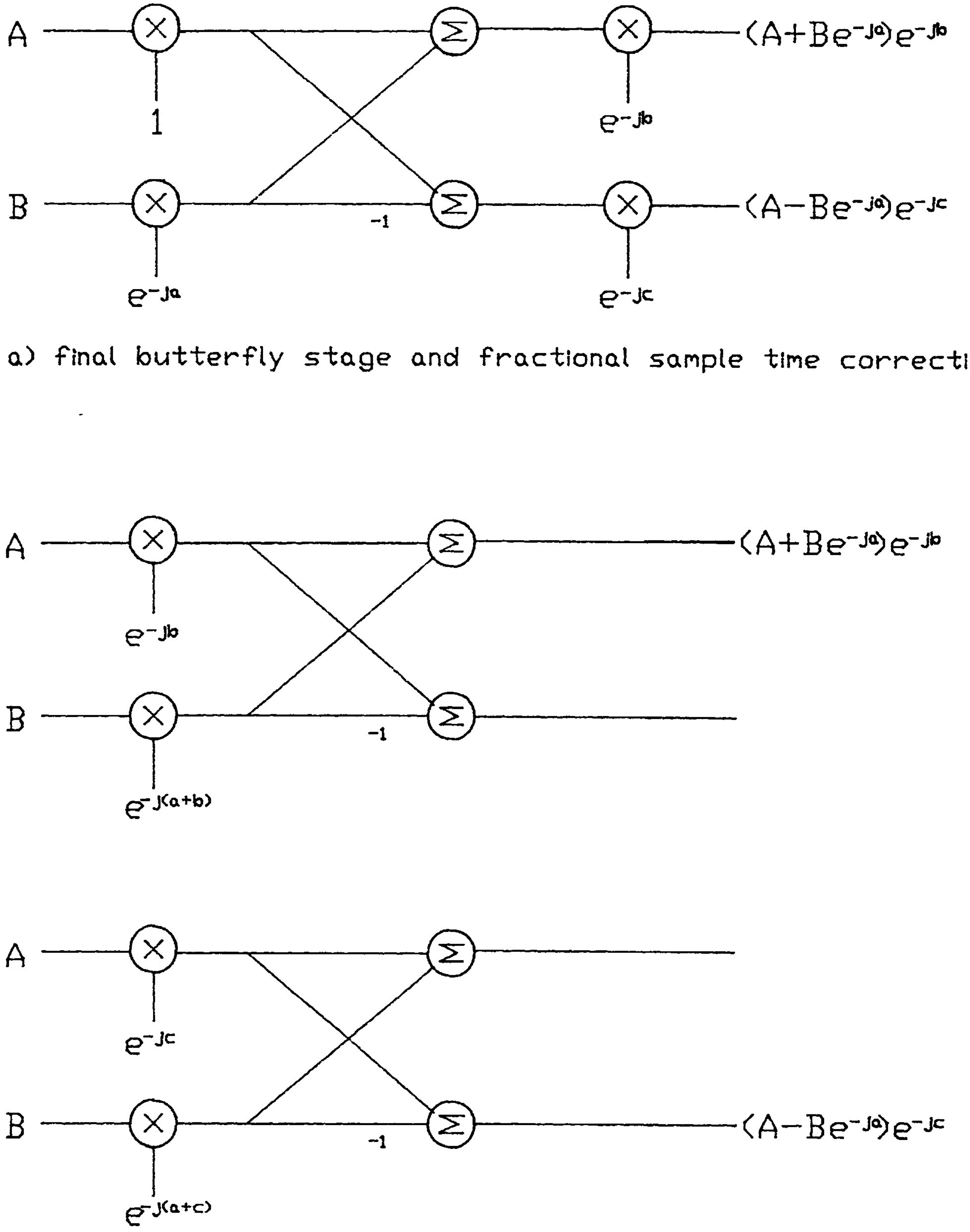
A number of things have changed in the VLBA correlator design since memo 71. Below is a list of such changes;

1) A tentative decision has been made to do 2048 point Fourier transforms. The main disadvantage to this size FFT was stated in memo 71 as the inability to use radix 4 FFT butterflies. In was recently realized, however, that a radix 2 butterfly was sufficiently like the radix 4 butterfly that a gate array chip designed to do a radix 4 butterfly could easily be used for both radix 4 and radix 2. Hence a 2048 point FFT can be perfomed by doing 5 radix 4 butterflies and one radix 2 butterfly.

2) Toshiba recently announced a series of very fast RAMs including a 25 nsec, 2K by 8 static RAM. I have called about this chip and was given a price of $6.00 each in quantity. At this low a price, I don't know if radix 4 butterflies make much sense. The advantages to radix 4 are fewer RAMs and fewer gate array chips (fewer IC's in general). If the RAMS are cheap enough and a smaller gate array (that would result from doing radix 2 butterflies) reduces the cost of the gate array chips, much of the advantage is lost. We intend to carry the correlator design along in both radix 2 and radix 4 for some time until the economic tradeoffs are clearer. Hence, this memo will present gate arrays designs for both.

3) The input multiply of the first butterfly stage in a DIT FFT is trivial and hence this gate array function is available to do whatever we can think of for it to do. The two leading contenders for this multiply are the fringe rotation and the window function. In this memo it will be assumed that the window function multiply will be done in the first FFT butterfly stage.

4) The last butterfly stage must calculate only one half of the points of other stages since one half of the



a) final butterfly stage and fractional sample time correction

b) final butterfly stage doing fractional sample time correction in two steps

FIGURE 1.

spectral points of the FFT output will be discarded. This fact allows the last butterfly stage to also do the fractional sample time error correction. One picture is worth a thousand words and figure 1 is that picture.

5) The numbering system has changed somewhat. The number representation will now have sign-magnitude mantissas external to the chip and one's complement in the chip adder stages. The advantage to the sign-magnitude input is in having smaller hardware multipliers. The advantage to one's complement is the ability to do un-biased truncation. A second change in the numbering system is the change from an integer conception of numbers in the FFT to fractions. A point in the FFT chain will now have the structure S.XXXX * 2**-YYYY where S is the sign bit, .XXXX is the 4-bit mantissa magnitude, and YYYY is the 4-bit exponent. This change has a few advantages in addition to being a slightly more natural form. First, the window generator can now have the full range of the number system and second, numbers will shift to the top of the butterfly adder stages resulting in less truncation error. The next section will try and explain this in more detail.

6) The sin/cos terms into the FFT butterflies will have 6-bit instead of 5-bit precision. This change was mentioned in correlator memo 71 and has now been adopted pending analysis of computer simulations of the system described in this memo.

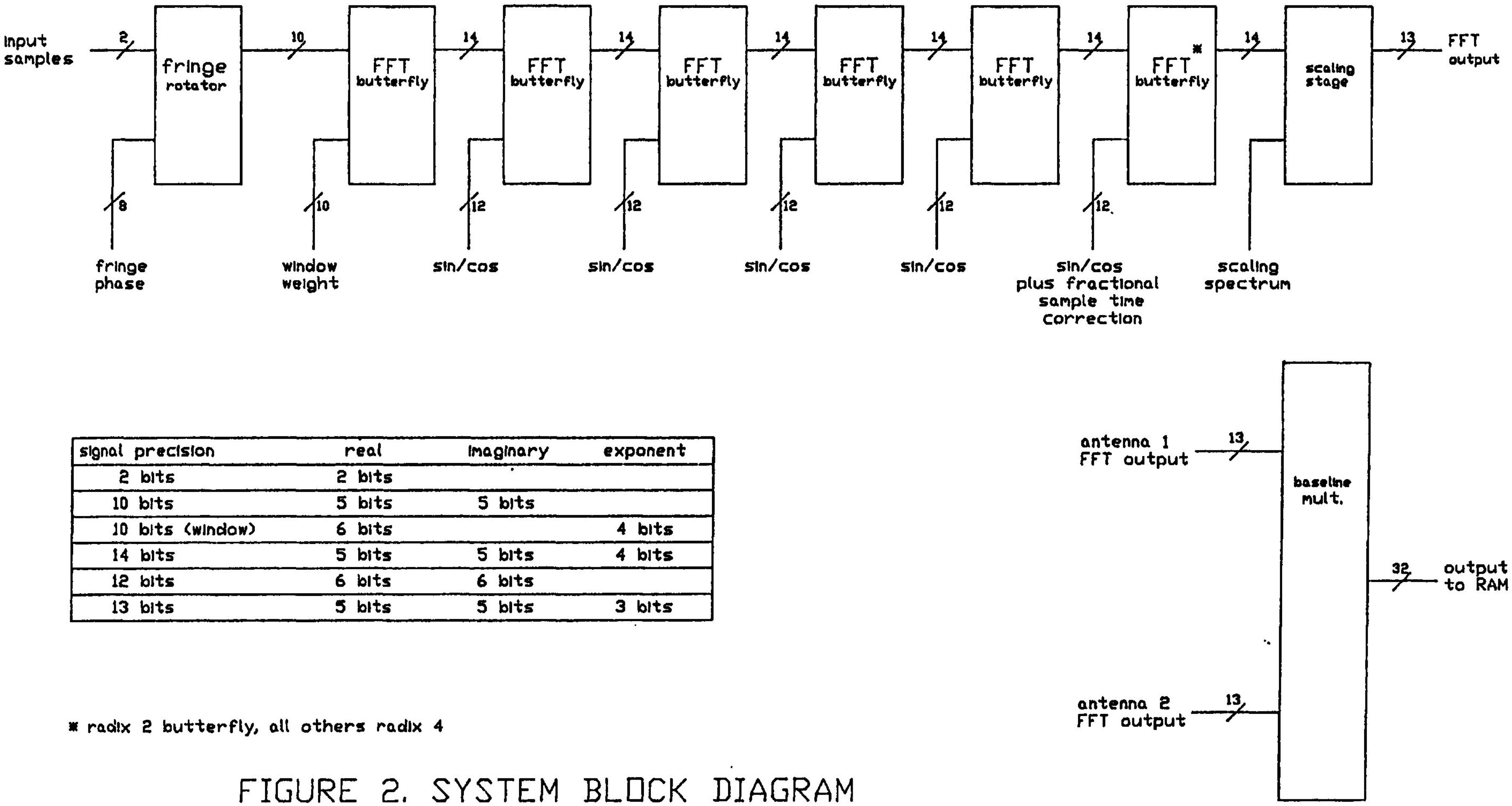
7) The gate counts given in figures 3 and 5 were higher that first anticipated and some thought is now being given to having several small gate array chips made instead of one large chip. The remainder of this memo will, however, consider a single multi-purpose chip.

III Block diagram

A block diagram of the signal path is seen in figure 2. A 2048 point FFT is done using 5 radix 4 butterflies and 1 radix 2 butterfly.

Each butterfly block shown consists in hardware of 5 integrated circuits, one gate array IC and two sets of two (actually 14/8) 2K by 8 RAMs. Two sets of RAMs are used for double buffering. If 15 nsec 2K by 8 RAMs become available in the future, this count can go down by one set of RAMs.

The baseline multiplier shown is a single gate array chip. This chip interfaces directly with an accumulator RAM. The 32 wide output of the baseline multiplier shown is the logical width.



signal precision	real	imaginary	exponent
2 bits	2 bits		
10 bits	5 bits	5 bits	
10 bits (window)	6 bits		4 bits
14 bits	5 bits	5 bits	4 bits
12 bits	6 bits	6 bits	
13 bits	5 bits	5 bits	3 bits

* radix 2 butterfly, all others radix 4

FIGURE 2. SYSTEM BLOCK DIAGRAM

The hardware width will be 8-bits (that is, 32-bit running sums of the multiplier/accumulator will be written to or read from the gate array chip in four 8-bit accesses).

Signal dynamic levels thru the block diagram of figure 2 are given below;

1) Fringe generator. Samples enter the fringe generator in one- or two-bit form and leave as complex fractions with the 5-bit mantissas of the form S.XXXX and zero exponents.

2) Window. Complex points enter in the form S.XXXX for real and imaginary components and zero exponents. Weights enter as real fractions with the form S.XXXXX * 2**-YYYY. Points leave complex with the full dynamic range potential of the 5, 5, 4 numbering system. Because of the limit set by using 15-bit registers and adders at the complex multiplier output in the gate array chip, the smallest effective non-zero number in the system will be S.SSSSSSSSSSSSX or S.SSSX * 2**-10 . If a more negative exponent is encountered, the significant bits of the number will be shifted out of the adder stage and only zero would be left.

3) FFT stage. In the FFT stages the points enter in the form S.XXXX * 2**-YYYY where, as above, YYYY has a effective maximum size of 10. The sin/cos terms enter the FFT butterfly as 6-bit fractions of the form S.XXXXX with zero exponent. All of the inputs to the FFT butterfly stage are in sign-magnitude notation. After the complex multiplier, an arithmetic shifter stage will convert the signals to one's complement fixed point numbers. Since the numbers are considered fractions with a radix point at the top of the subsequent adder stages, any error due to the shifting significant bits below the LSB of the adder is minimized. The FFT butterfly chip allows a 2-bit growth in the maximum size of points in each radix 4 stage. To keep a fractional notation through the FFT an effective scaling of 2**2 will be applied. That is, if a point comes out of a radix 4 butterfly with the same exponent it had at the input, it in actuality grew by two bits in the stage. The samples of a flat spectrum will thus progress down the FFT with exponents that get ever more negative.

4) Scaling stage. The scaling stage will add a number, programmable by spectral point (and antenna if that is necessary), to the exponent of each spectral point out of the FFT. The smallest exponent allowed at the scaling stage output will be -5. Thus, points that are much smaller than the expected value (as reflected by the programmed scaling factor) will have significant bits truncated and points that are much larger that expected will be clipped. Scaling is necessary because without it the output products of the baseline multiplier would have a 2**30 dynamic range.

5) Baseline multiplier. The input points to the baseline multiplier will range between S.XXXX and S.XXXX * 2**-5. Only 16 bits of the product between such numbers from two antennas will be kept. If, for example, the two points to be multiplied were +.000001001 and +.000001100, +.000000000001101100 should be the result. The 16 bits accumulated would be +.000000000001101 and there would be a truncation error of +.000000000000000100.

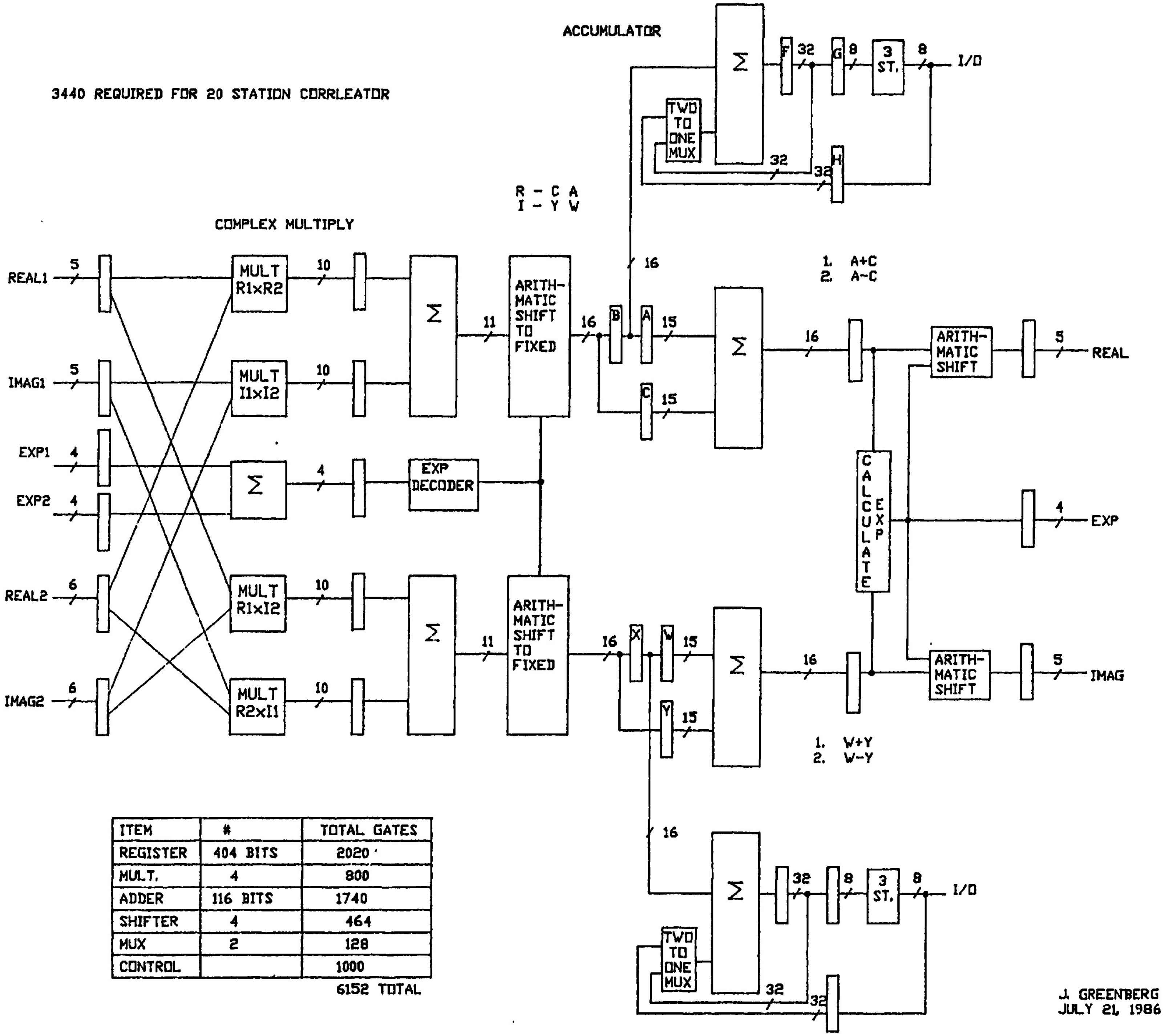
6) Accumulator. The accumulator input will be 16-bit 2's complement numbers (converted from the one's complement output of the complex multiplier). The accumulator will integrate up to the 32-bit level. After accumulation, the scaling factors introduced by the scaling stage will be reunited with the baseline integrations in the correlator back end.

IV) The gate array chip

This section describes a plan to implement a gate array integrated circuit which can perform four distinct. functions. They are fractional sample time error (FSTE) corrections, baseline multiplies, accumulation, and radix two and radix four butterflies. The FSTE function may not be needed since, as discussed above, a final radix 2 butterfly stage can perform this function.

The chip will be driven by a 32 MHz clock.

3440 REQUIRED FOR 20 STATION CORRLEATOR



ITEM	#	TOTAL GATES
REGISTER	404 BITS	2020
MULT.	4	800
ADDER	116 BITS	1740
SHIFTER	4	464
MUX	2	128
CONTROL		1000
		6152 TOTAL

J. GREENBERG
JULY 21, 1986

FIGURE 3. THE RADIX 2 CHIP BLOCK DIAGRAM

Two versions of the chip will be presented. One will perform either radix 2 or radix 4 butterflies. The other will only perform radix 2 butterflies. A butterfly consists of floating point complex multiplies, followed by additions of the products. The complex multiply portion can also be used to implement the fractional sample time error (FSTE) corrections. FSTE is also referred to as fractional bit shift.

An accumulator section after the complex multiply can be used to implement the baseline multiply/accumulate function.

Thus, a single integrated circuit can be selectively used in these four functions.

A) Floating Point, Complex Multiply Function

Figure 3 is the Radix 2 Chip Block Diagram. All the chip functions use the initial complex multiply. Two floating point, complex numbers are input into the chip. The real and imaginary portions of each share a common exponent. To justify this, think of a number in polar form. The larger component is the major contributor to the magnitude. The exponent can be added separately from the multiplications since:

```
(A + jW) * 2**L * (B + jX) * 2**M =
        [(AB - XW) + j(AX + BW)] * 2**(L+M))
```

The resolutions used depend on the function desired. Refer to a 5 bit real mantissa, a 5 bit imaginary mantissa, and a 4 bit exponent as 5, 5, 4.

For the butterfly and FSTE applications, a 5, 5, 4 number will be multiplied by an angle of the form cos a + j sin a. Since the angle has unity magnitude, the exponent will always be zero. B. Clark has suggested that 6 bits of resolution in the mantissa is desirable for the angle. Hence, 5, 5, 4 and 6, 6, 0 numbers will be multiplied together.

For the baseline multiply, two 5, 5, 3 numbers will be multiplied together.

The chip will contain four 5-bit x 6-bit multipliers. These will perform the four multiplications required to perform the complex multiply in a single machine cycle. The output precision of the multipliers is 10 bits, assuming sign-magnitude multiplies.

After the exponents are added, it is necessary to convert to fixed point notation for the subsequent additions. Recall the number are stored in a S.XXXX * 2**-YYYY format, where the exponent, -YYYY, has a maximum value of zero. Thus the peaks of the spectrum will have a value of one minus the LSB. The conversion is implemented by means of an arithmetic shifter. The exponent determines the number of shifts.

B) Radix 2 Butterfly Implementation

To perform a radix 2 FFT butterfly, it is necessary to perform the sum and difference of two, sequential complex products. Figure 3 is the block diagram of a gate array chip that will do a complete radix 2 FFT butterfly. Figure 4 is the flow diagram of a radix 2 FFT butterfly. Referring to figure 3, the registers A, B, and C store the real parts. The registers W, X, and Y store the imaginary parts.

The chart below, shows the radix 2, two port adder timing.

	Clock Cycle							
	1	2	3	4	5	6	7	8
1.	A1	C1	A2	C2	A3	C3	A4	C4
2.			A1+C1	A1-C1	A2+C2	A2-C2	A3+C3	A3-C3
3.	W1	Y1	W2	Y2	W3	Y3	W4	Y4
4.			W1+Y1	W1-Y1	W2+Y2	W2-Y2	W3+Y3	W3-Y3

Each column represents one clock cycle. What is calculated in each entry is strobed in at the end of the entry. Four, two clock cycles are shown. The number associated with each entry tells the cycle number.

Row 1 shows the product terms being fed to the real stage of the adder. The A, B, and C registers are used. Row 2 shows the sums and differences output. The imaginary portion works similarly.

The adder outputs are in fixed point notation. The outputs must be converted back to 5, 5, 4 floating point notation. The real or imaginary component with the largest magnitude is used in calculating the exponent. This exponent is then used to arithmetic shift the fixed point, complex number back to 5, 5, 4 floating point. This forms the output of the butterfly.

C) FSTE implementation

The FSTE function requires a complex multiply, without the two port addition. The addition can be bypassed by having zero always applied to one port of the adder.

For the FSTE function, the complex product is converted back to 5, 5, 4 floating point and output. It was necessary to convert the product first to fixed, then back to floating point to recalculate the exponent.

D) Baseline Multiply Implementation

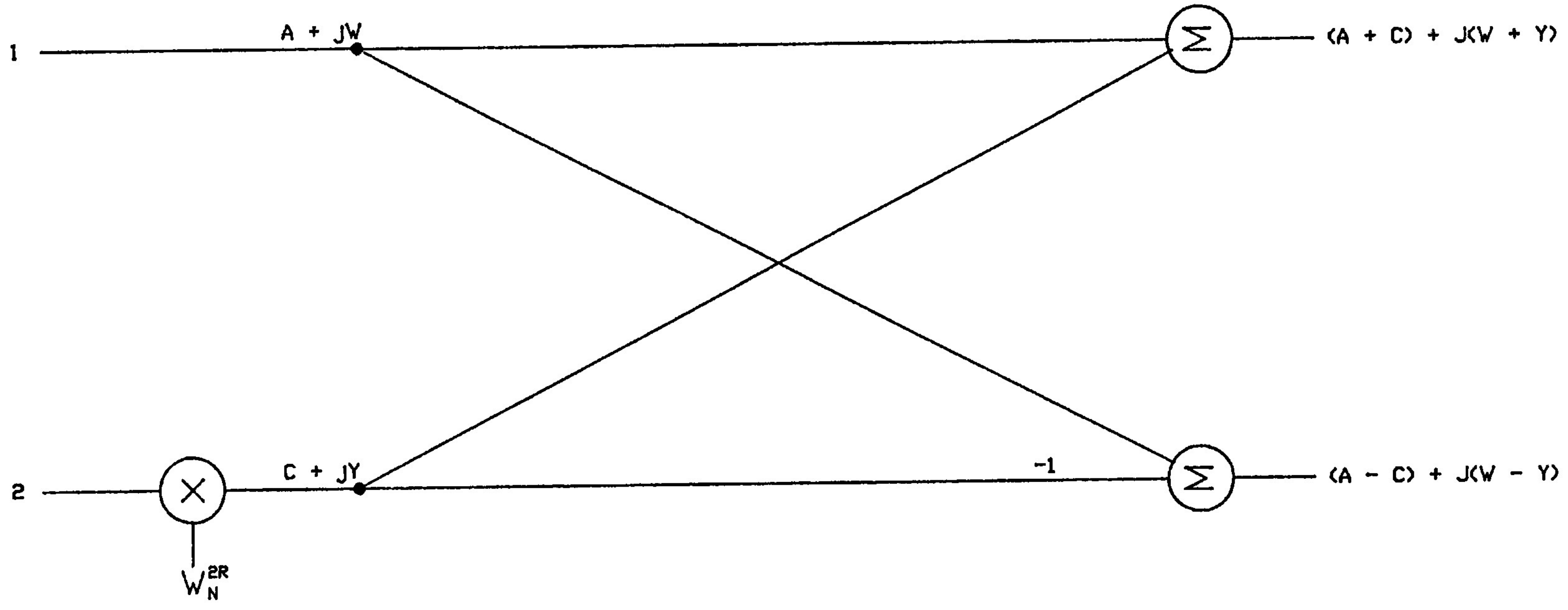


FIGURE 4. RADIX 2 DIT FFT BUTTERFLY

For the baseline multiply function, the 16 bit outputs of the complex multiplier are stored in registers B and X. These then go to the accumulators.

The accumulation takes place in two stages. The sum will be added to a sum stored in RAM.

The chart below shows the accumulator timing when 8 consecutive cross products are summed together before accumulation.

	Clock Cycle							
	1	2	3	4	5	6	7	8
1.	A1	A2	A3	A4	A5	A6	A7	A8
2.	W1	W2	W3	W4	R1	R2	R3	R4

The I/O to the RAM sends 32 bit words over an 8 bit bus in 4 clock cycles.

On the above chart, A1 through A8 represent 8 consecutive cross products. W1 through W4 represent the writing of a 32 bit word to RAM in 4, 8 bit sections. R1 through R4 represent the reading of a 32 bit word from RAM in 4, 8 bit sections.

Referring to figure 3 and the above chart, the accumulator timing is described for an eight clock cycle as follows:

Clock Cycle Number

1. The F register here contains the summation of the previous value from RAM plus the previous 8 cross multiplies. To output the F register, 24 of its 32 bits are temporarily stored in Reg G, while 8 bits are output to RAM as W1. The H register (Reg H) has previously been read from RAM.

    Reg H + A1 -> Reg F.

This adds the value from RAM to the first cross product, and accumulates the sum in register F.

2. W2 is output from Reg G.

    A2 + Reg F -> Reg F.
    This accumulates the cross products.

3. W3 is output from Reg G.

    A3 + Reg F -> Reg F.

4. W4 is output from Reg G.

    A4 + Reg F -> Reg F.

5. R1 -> Reg H.
   This begins the inputting of a 32 bit word from RAM in 4, 8 bit sections.

   A5 + Reg F -> Reg F.

6. R2 -> Reg H.

   A6 + Reg F -> Reg F.

7. R3 -> Reg H.

   A7 + Reg F -> Reg F.

8. R4 -> Reg H.

   A8 + Reg F -> Reg F.

And back to clock cycle 1.

The imaginary values are accumulated, in parallel, in a separate accumulator.

E) Radix 4 Butterfly Chip Implementation

Figure 5 shows the block diagram of a chip which can perform radix 4 butterflies. It is identical to the radix 2 chip, with the addition of the blocks enclosed in the dashed lines. That section implements the second stage of the four port adder function. The output of the four port adder is 16 bits, as was the output of the radix 2, two port adder. The first stage of the four port adder also outputs 16 bits. This allows for when it will be used as a radix two butterfly. Figure 6 is a picture of the radix 4 FFT butterfly.

The four port adder performs 4 sums of the 4 complex products. Assume the four complex products are output in the following order: A+jW, C+jY, B+jX, D+jZ (see figure 6). The four outputs required from the butterfly are:

1. [(A+C)+(B+D)] + j[(W+Y)+(X+Z)]
2. [(A+C)-(B+D)] + j[(W+Y)-(X+Z)]
3. [(A-C)+(X-Z)] + j[(W-Y)-(B-D)]
4. [(A-C)-(X-Z)] + j[(W-Y)+(B-D)]

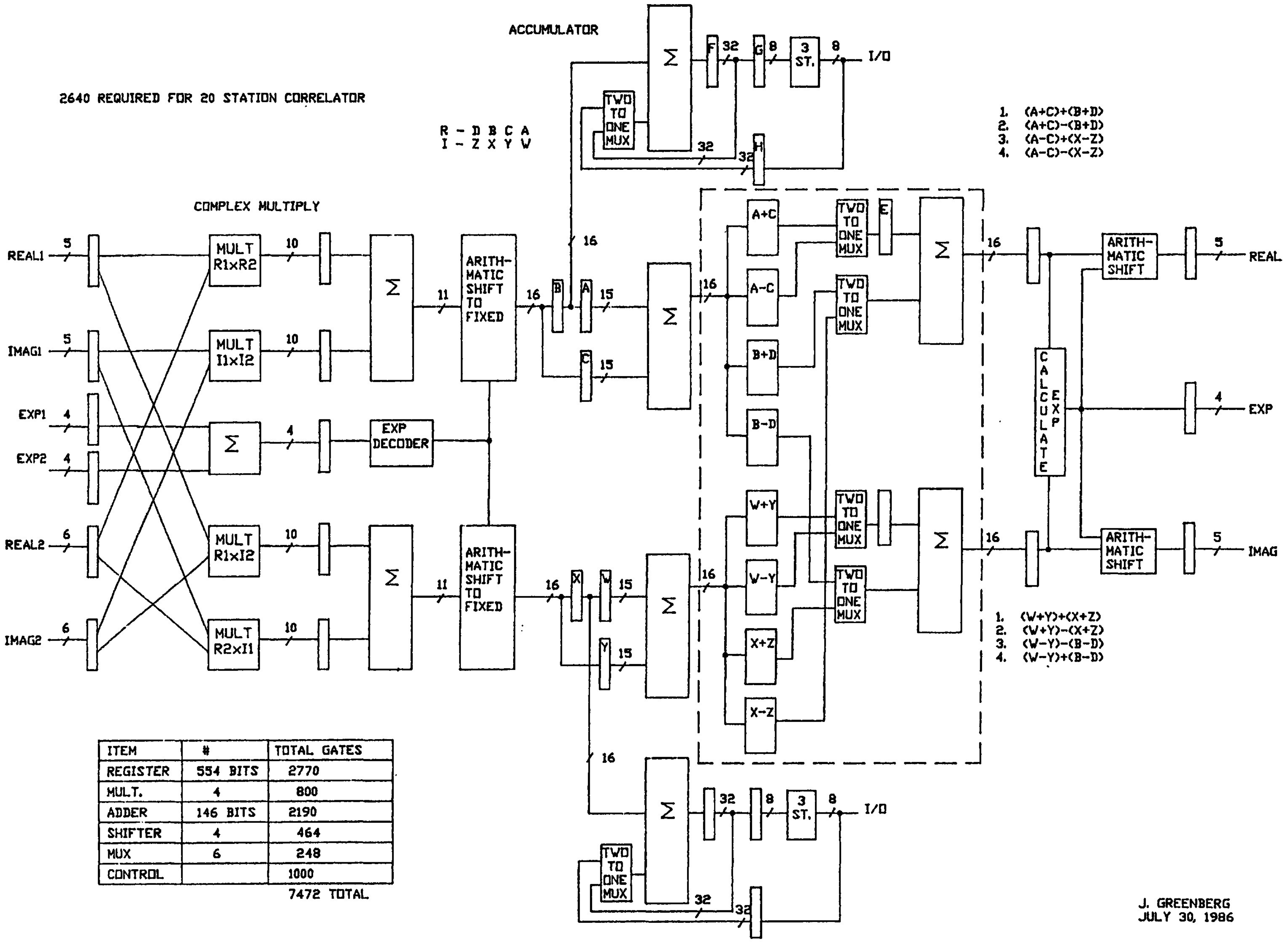
The method to compute the real halves will be described. The imaginary halves are analogous.

On figure 5, registers A, B, and C alternately offer A and C, then B and D to the adder. The adder will calculate the sums and differences and store them in the next row of registers. These will be multiplexed and summed to form the final outputs.

2640 REQUIRED FOR 20 STATION CORRELATOR

R - D B C A
I - Z X Y W

1. (A+C)+(B+D)
2. (A+C)-(B+D)
3. (A-C)+(X-Z)
4. (A-C)-(X-Z)



1. (W+Y)+(X+Z)
2. (W+Y)-(X+Z)
3. (W-Y)-(B-D)
4. (W-Y)+(B-D)

ITEM	#	TOTAL GATES
REGISTER	554 BITS	2770
MULT.	4	800
ADDER	146 BITS	2190
SHIFTER	4	464
MUX	6	248
CONTROL		1000

7472 TOTAL

J. GREENBERG
JULY 30, 1986

FIGURE 5. THE RADIX 4 CHIP BLOCK DIGRAM

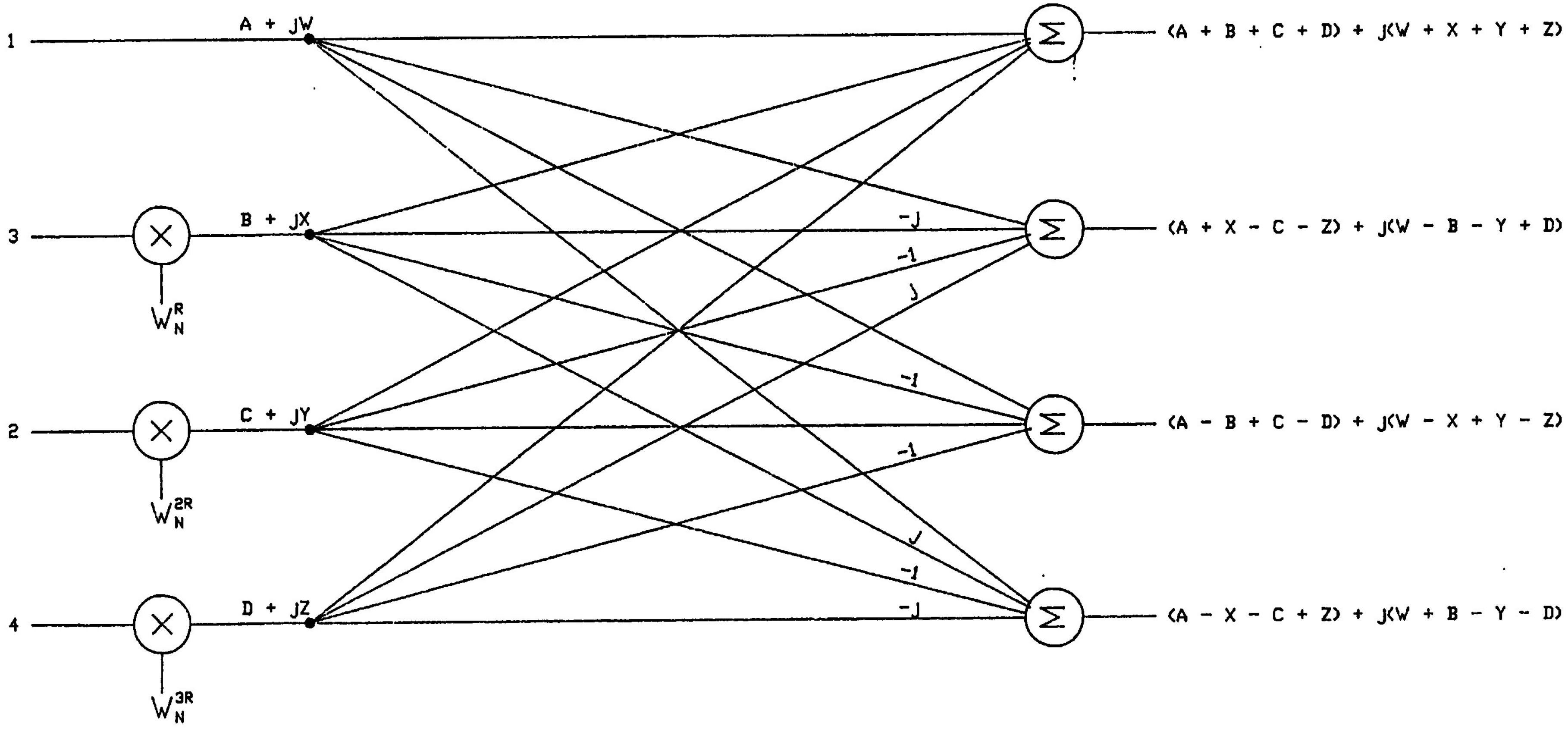

1
A + JW
(A + B + C + D) + J(W + X + Y + Z)
3
B + JX
-J
(A + X - C - Z) + J(W - B - Y + D)
-1
J
W_N^R
-1
2
C + JY
(A - B + C - D) + J(W - X + Y - Z)
-1
W_N^{2R}
J
-1
4
D + JZ
-J
(A - X - C + Z) + J(W + B - Y - D)
W_N^{3R}

FIGURE 6. RADIX 4 DIT FFT BUTTERFLY

The four port adder timing is shown in the chart which follows.

				Clock Cycle						
	1	2	3	4	5	6	7	8	9	10
1.	A1	C1	B1	D1	A2	C2	B2	D2	A3	C3
2.			A1+C1	A1-C1	B1+D1	B1-D1	A2+C2	A2-C2	B2+D2	B2-D2
2a.					A1+C1	A1+C1	A1-C1	A1-C1	A2+C2	A2+C2
3.	W1	Y1	X1	Z1	W2	Y2	X2	Z2	W3	Y3
4.			W1+Y1	W1-Y1	X1+Z1	X1-Z1	W2+Y2	W2-Y2	X2+Z2	X2-Z2
4a.					W1+Y1	W1+Y1	W1-Y1	W1-Y1	W2+Y2	W2+Y2
5.						A1+C1 + B1+D1	A1+C1 - B1+D1	A1-C1 + X1-Z1	A1-C1 - X1-Z1	A2+C2 + B2+D2
6.						W1+Y1 + X1+Z1	W1+Y1 - X1+Z1	W1-Y1 - B1-D1	W1-Y1 + B1-D1	W2+Y2 + X2+Z2

Each column represents one clock cycle. What is calculated in each entry is strobed in at the end of the entry. Two and one half, four entry cycles are shown. The number associated with each entry tells the cycle number.

Row 1 shows the real products from the multipliers. Row 2 shows the sums and differences of the terms from row 1. Row 3 shows the imaginary products. Row 4 shows the sums and differences of the imaginary terms. Row 5 shows the sums and differences of the sums and differences from row 2. This produces the butterfly output terms. Note that for row 5, clock cycle 9, A1-C1 is required. In row 2, at the end of the previous clock cycle, A2-C2 was written. Hence A1-C1 would no longer be available. To resolve this, register E is added to the upper input of the second stage adder. Row 2a shows what is stored in register E. Register E is clocked at the end of the even clock cycles. The multiplexer on the input to the register is toggled at the end of the odd clock cycles. The timing for the imaginary portion is analogous.

Hence, the four complex, radix four butterfly terms are output from the four port adders. One complex term is output each clock cycle.

If a 2048 point FFT is to be implemented, using radix 4 butterfly chips, one stage of chips must do a radix 2 FFT. Since 2048 equals 4**5 * 2, it requires 5 radix 4 stages and one radix 2 stage. The radix 4 chip can be made to perform a radix 2 butterfly by bypassing the last adder stage. For example, for the real portion, have the upper MUX permanently in the A+C position and the lower MUX output zero. This bypasses the second stages of the four port adders, forming two port adders.

F) Possible Multiple Uses of Hardware

The gate count could be reduced by switching hardware between functionalities for different applications of the chip. Some suggestions follow.

The adders in the accumulators could be shared with the adders in the four or two port adders. The four or two port adders are not otherwise used in the baseline multiply function. A counter could implement the most significant bits of the accumulator 32 bit adder, since it would be counting carries from the 16 bit inputs.

The multiplexing of the 32 bits from the accumulator to the 8 output bits could be done by the arithmetic shifters of the final fixed point to floating point conversion. This would also implement a dual use of the output pins between the functions.