R.K. MALIK

V.M. TATKE

ref : rkm7/FXARITH/july92

July 25, 1992

DIGITAL ARITHMETIC INSIDE THE FX-ASIC

INTRODUCTION: Data that enters the FX-ASIC undergoes a series of binary operations which one should be aware of in order to interpret the bit-patterns that appear at the output of the ASIC. This document is intended to make the life a little (and only a little !) easier for a reader interested in knowing the arithmetic operations being performed inside the FX Correlator Chip.

DATA FORMATS:

(a)FFT MODE:

(i) Input:

Real	= 7-bits	Sign-magnitude
Imaginary	= 7-bits	Sign-magnitude
Exponent	= 4-bits	Implied -ve sign

(ii) Input at the Twiddle Port:

(at Stage 0 (Window function))

Real	= 5-bits	Sign-magnitude
Imaginary	= 0-bits	Sign-magnitude
Exponent	= 4-bits	Implied -ve sign

(at other FFT Stages)

Real	= 5-bits	Sign-magnitude
Imaginary	= 5-bits	Sign-magnitude

Exponent	= 0-bits	Implied -ve sign

(iii) Output:

Same as the Input Format at the Normal Port

(b)MAC MODE:

(i) Input (at both the Normal and the Twiddle Ports:

Real	= 4-bit	Sign-magnitude
Imaginary	= 4-bit	Sign-magnitude
Exponent	= 4-bit	Implied -ve

(i) Output:

Real	= 15-bit	**1**'s Complement
Imaginary	= 15-bit	**1**'s Complement
Exponent	= 6-bit	**2**'s Complement

In all the cases mentioned above, the mantissa is always a fraction (¡1).

It must be pointed out here that once the ASIC is configured for the MAC mode, internally some of the input bits are ANDed to 0 and therefore the external input can only be in the 4,4,4 Format even though more pins are physically available on the chip. The main reason for not bringing all the 7,7,4 FFT outputs to the MAC, therefore, is the ASIC Architecture itself rather than anything else.

A block diagram of the various Arithmetical Units is shown in Fig.1. Hereafter we will attempt to describe each and every block separately.

(I) COMPLEX MULTIPLY: Let the two numbers entering the Normal and Twiddle Ports be $(R1 + jI1)$ and $(R2 + jI2)$ respectively.

Here we are considering the mantissas only since the Exponents are processed separately in the ASIC. These two data points are in the format already described.

The COMPLEX MULTIPLY has to perform the following job:

(i) for FFT :

$$\text{output} = (R1 + jI1)\ X\ (R2 + jI2)$$
$$= (R1R2 - I1I2) + j(R1I2 + R2I1)\ ;$$

(i) for MAC :

$$\text{output} = (R1 + jI1)\ X\ (R2 + jI2)^*$$
$$= (R1R2 + I1I2) + j(R1I2 - R2I1)\ ;$$

In the ASIC, the Real and the Imaginary data paths are separate — Top and Bottom in [1]. Firstly all the product terms – R1R2, I1I2 etc– are computed simultaneously. Just before computing the product I1I2 in the Real Data Path, SIGN Bit of I1 is inverted for both FFT and MAC. Alongwith this, SIGN bit of I1 is inverted at the input to the COMPLEX MULTIPLY block itself. With this arrangement, one gets all the desired product terms with appropriate signs.

11-Bit output for each product term is converted to 1's Complement and then SIGN-EXTENDED to **12-Bits** in the 12-Bit Adder which gives **12-Bit** 1's Complement number for (R1R2 - I1I2) or (R1R2 + I1I2) for FFT/MAC as the case may be, in the Real Data Path and similarly one gets **12-Bit** 1's Complement outputs for the Imaginary parts of the terms.

EXPONENT HANDLING: Both the **4-Bit** Exponents are ADDed to produce **5-Bit** Output including a **CARRY**.*note: both the exponents represent magnitude only with an IMPLIED -ve Sign).* Carry bit is used to indicate Exponent Overflow condition on one of the ASIC pins provided BIT-22 in the Control Word is set to 1. It must be noted that in case Exponent Overflow is detected (provided it has been enabled!), all the **5** output bits are CLEARed to 0. Also EXPOF detection cannot be enabled in the MAC mode.

5-Bit Exponent Sum – referred to as XMULT in NRAO documents– available now is SIGN EXTENDED and converted to a **6-Bit 2**'s Complement number (by adding 1 to 1's Complement).

At the same time, this sum is subtracted from the SUMEXP – the exponent of the accumulated result in **2**'s Complement in MAC mode. *(since before converting to 2's Complement, XMULT has*

an implied -ve sign, the subtraction is in fact addition of SUMEXP with the implied -ve XMULT. This addition is implemented in a 6-Bit 2's Complement Adder). The result of this subtraction decides the amount of shift required for conversion to Fixed Point/Common Exponent (FFT/MAC) and whether to use the POSITIVE SHIFTER or the NEGATIVE SHIFTER in the next stage.

(II) FLOATING TO FIXED POINT or COMMON EXPONENT: Conversion to Common Exponent (in case of MAC) requires retaining the larger (less negative) of the SUMEXP and XMULT as the Common Exponent in 2's Complement form, and Right-Shifting the Mantissa bits of the smaller-exponent number by an amount, SHIFT, equal to the magnitude of the difference between XMULT and SUMEXP .

There are two types of **15-Bit SHIFTERs** to do the job — POSitive and NEGative. Mantissas (real and imag) from the accumulated sum sits in the SHIFTERs designated NEG while POS SHIFTERs contain the mantissa output from "COMPLEX MULTIPLY", SIGN Extended (at RIGHT) from **12 to 15-Bits.** When the subtraction – (SUMEXP - XMULT) gives a negative result, contents of the NEG Shifters are shifted RIGHT and similarly POS SHIFTER contents are shifted RIGHT in case of the subtraction yields a positive result.

At this point, one has to worry about the following:

(i) NEG SHIFTER: If (SHIFT ¿ 16), all the SHIFTER output bits are set to 0.

Else the Bits are shifted RIGHT by an amount=SHIFT with SIGN EXTENSION on the LEFT (MSB side).

(ii) POS SHIFTER: If (SHIFT ¿ 15), all the output bits are set to 0 (all the bits get shifted out). Else the normal SHIFTing operation.

This is the first instance in the Chip where one may lose a few Bits(resolution).

Now, FFT is a special case of the description given above where SUMEXP=0 and thus only the POS SHIFTERs are used – reader can easily verify that (and obviously, NEG SHIFTER Contents have no relevance in FFT !!). Again, since the SHIFT = magnitude of the XMULT, obviously the conversion

is from Floating Point to Fixed Point rather than to Common Exponent as in MAC.

(III) ACCUMULATOR or FFT BUTTERFLY: Here first we describe the ACCUMULATOR Operation. Two numbers of width **15-Bit** each from POS and NEG SHIFTERs are SIGN EXTENDED and added together in a **16-Bit** 1's Complement Adder. The accumulated value always goes thru the register '(A + C)' only as the other three registers are used only in FFT. The result at this stage is a **16-Bit** 1's Complement number each in both the Real and Imag data paths. If two MSBs of the Real Part are Identical and same is the case for the two MSBs of the Imag part, then MSB of the mantissa is rejected to obtain a **15-Bit** output. Else the LSB is rejected. This output LOOPs back, when required, for Polarization Mode, otherwise this is the Accumulation Result that gets stored in the internal RAM – all set to be recycled for further accumulation.

2's Complement Exponent (**6-Bits**) has already been obtained in the previous section and thus we have the complete **15,15,6** Word.

note: 16-Bit 1's COMPLEMENT ADDER mentioned in this section has a feedback mechanism to avoid oscillation arising out of the ambiguity in the representation of 0 in 1's Complement form. Any 0 is forced to be a +ve 0 unlike in the ADDER in the previous stage.

In case of FFT, four consecutive outputs from the POS SHIFTER are suitably time-delayed to form four Additions in the **16-Bit** ADDER – '(A + C)', (A - C), (B + D) and (B - D). All except (A + C) outputs are **15-Bit**. For (A+C), all the 16 Bits are retained for RADIX2 FFT. In case of RADIX4, however, LSB is rejected and the MSB is made identical to the 15th Bit.

The rest of three outputs viz. (A-C),(B+D) and (B-D) are sign extended later in their progress towards the 16-Bit 1's COMPLEMENT ADDER. This adder is reqd. to perform the various combinations of the (A+C),(A-C),(B+D) and (B-D) outputs mentioned earlier (in radix 4 fft the final output is (A+B+C+D) etc.). The output of this adder is then converted to sign magnitude form which is 16 bits wide. This entire sequence of operations described till this point is valid for the

imaginary data stream. The two sign magnitude outputs pertaining to the REAL and IMAGINARY parts of the data stream are brought together to calculate the common exponent since both the input and output have a common exponent. After the common exponent is derived, the REAL or IMAGINARY mantissas are shifted so that both have the same exponent mentioned above. The data thus conditioned in the 7,7,4 format is routed out of the ASIC

ACKNOWLEDGEMENTS: We thank Chuck/Joe Greenberg at NRAO for discussions over email about the ASIC details and for their efforts to clarify some points.

REFERENCES:

1. "ASIC Block-Diagram – Document No. 56000K001"

2. "VLBA1 SCHEMATIC, 84 PAGES TOTAL"

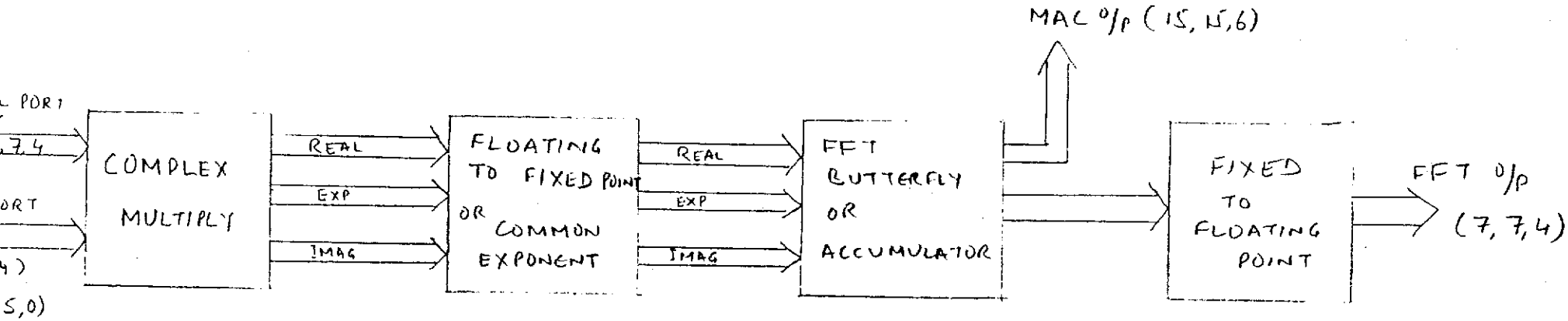
MAC O/P (15, N, 6)
L PORT
7,4
ORT
4)
5,0)
COMPLEX MULTIPLY
REAL
EXP
IMAG
FLOATING TO FIXED POINT OR COMMON EXPONENT
REAL
EXP
IMAG
FFT BUTTERFLY OR ACCUMULATOR
FIXED TO FLOATING POINT
FFT O/P (7, 7, 4)

FIG. 1