Reconfigurable FFT System On Chip (SOC)

G. Venkataramana Sagar
Department Of ECE,
G.P.R. Engg. College,
Kurnool, India, 518002

Dr. K. Srinivasa Rao,
Department of ECE,
TRR College of Engg.
RR District, India

ABSTRACT
With onset of paradigms of System On Chip (SOC) to design a module for real time applications or voice codec's, The SOC's have different requirements for operands precision we propose a reusable FFT [2] using reconfigurable multiplier [6]. How ever, the FFT perform either combining N and N/2 bit multiplications in the same N bit tree multiplier. The key challenges in designing a reusable FFT are to limit the impact of flexibility on power operations that are needed for FFT butterfly to perform better than a conventional, dedicated FFT butterfly.

KEYWORDS: System on Chip, Reconfigurable, reusable, Butterflies.

1. FAST FOURIER TRANSFORMS

1.0 Introduction:

A broad class of applications requires the analysis of signals by determining their sinusoidal components that is achieved by transforming a sequence {x[n]} of length L < N into a sequence of frequency samples {X[k]} of length N. These Frequency samples are obtained by evaluating the Fourier transform X(ω) at a set of x[n]. DFT provides the decomposition of input into N periodic components.

Discrete transform of x[n] is given by DFT as

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{-j2\pi kn/N} \quad k = 0,1,..., N-1$$

Thus the transform provides a mutually unique way of correspondence between the sequence of N samples in the original domain and the sequence of frequency domain (spectral) coefficients of the same length.

Discrete Fourier transform is also defined by the core matrix

$$A = [W^{kn}] = \begin{bmatrix} W^0 & W^0 & . & W^0 \\ W^0 & W^1 & . & W^{N-1} \\ . & . & . & . \\ W^0 & W^{N-1} & . & W^{(N-1)^2} \end{bmatrix}$$

$$where \quad W = e^{-j2\pi/N}$$

The basis functions are sampled complex exponential functions. The essence of the Fourier transform of a wave form is to decompose or separate wave form into a sum of sinusoids of different freq. In other word, the Fourier transform identifies or distinguishes the different freq sinusoids, and their amplitudes, which combine to form an arbitrary wave form. Freq domain representation of a function contains exactly the same information as that of the original function.

The N separate DFT [1] analysis frequencies or set of frequency samples which define the spectrum on frequency axis, whose discrete frequency locations are given by

$$f_k = \frac{kF}{N} \quad k = 0,1,..., N-1$$

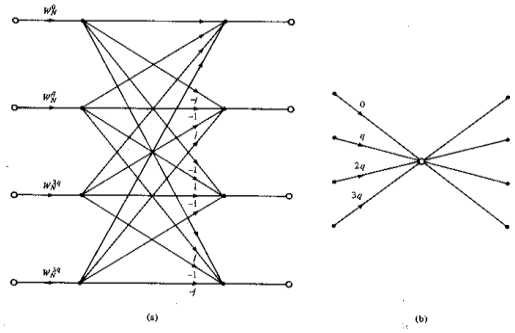

Figure 1.1 Basic butterfly computations in a radix-4 FFT algorithm.

The frequency resolution of the DFT is equal to the frequency increment F/N and is also referred as bin spacing. The X(0) DFT term corresponds to magnitude "DC" term. X(1) term corresponds to magnitude of fundamental freq. When a real input signal contains a sine wave component of peak amplitude A_0 with an integral number of cycles over N input samples, the o/p magnitude of the DFT for that particular sine wave is M_r where 1.2

$$M_r = \frac{A_0 N}{2}$$

If the DFT i/p is a complex sinusoid of magnitude A_0 (i.e, $A_0\ e^{j2\pi ft}$) with an integral number of cycles over N samples, the o/p magnitude of the DFT is M_c where $M_c = A_0 N$

$$f_{analysis}(m) = \frac{mf_s}{N} \qquad for \quad m = 1$$

1.1 Radix-4 FFT Algorithm

Let us begin by describing a radix-4 decimation-in-frequency FFT algorithm briefly. We split or decimate the *N*-point input sequence into four subsequences *x*(4*n*), *x*(4*n*+1), *x*(4*n*+2), *x*(4*n*+3), *n* = 0, 1, ... , *N*/4-1.

The radix-4 butterfly is depicted in Figure 1.1(a) and in a more compact form in Figure 1.1(b). Note that each butterfly involves three complex multiplications since $W_N^{\ 0}$ = 1, and 12 complex additions. A 16-point, radix-4 decimation-in-frequency FFT algorithm is shown in Figure 1.2

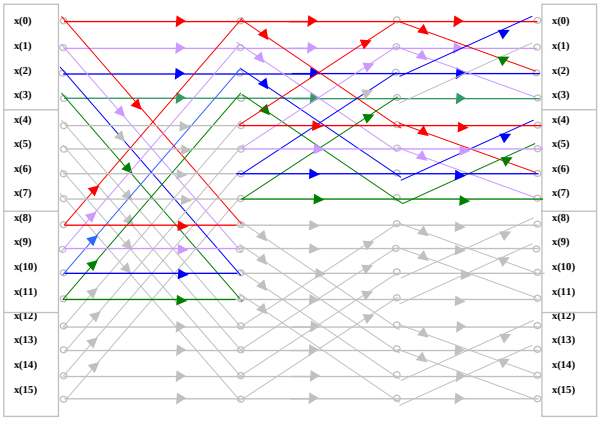

Fig 1.2 16-point,radix-4 decimation-in-frequ-ency algorithm with input in normal order and output in digit-reversed order.

Its input is in normal order and its output is in digit-reversed order. It has exactly the same computational complexity as the decimation-in-time radex-4 FFT algorithm.

2. IMPLEMENTATION OF FFT

2.0 Architectural description

In an FFT implementation there are many parameters that can be made flexible and reconfigurable. In the FFT design of this paper, the focus of flexibility features was on the number of points (*N*), operand scaling and, more interestingly, the twin-precision techni-que. The general architecture for the proposed implementation is a radix-4 DIF FFT butterfly which is fed with complex values from an internal RAM [6]**.** The following figure 2.0 gives the architectural view of the FFT description as detailed below.

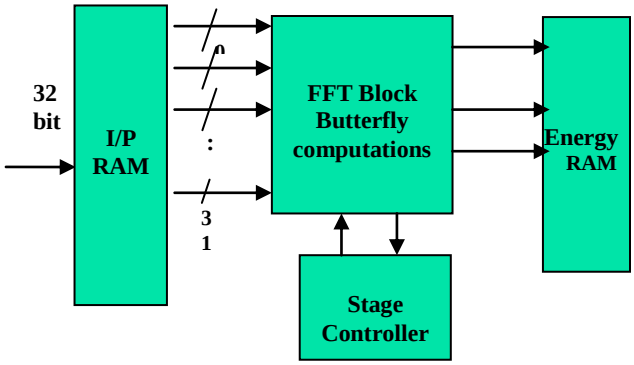

Figure 2.0 Architectural flow diagram of FFT

Input RAM: Input storing unit consisting of 16 locations of 16 bit each (i.e RAM), it takes 12 samples with the offset of 2 and centering . Input RAM takes data parallely with sub-band analysis from the input unit.

FFT Block: Fast Fourier Transform is the main processing block which carries out the transformation for it consisting of three stages with 4X4 butterfly operations [3],[4]& [5]. The twiddle factors for operation are being processed all to the block from the package defined. Samples (1,5,9,13), (2,6,10,14), (3,7,11,15) and (4,8,12,16) are passed down to the first stage of four numbers of 4X4 butterflies respectively and also for second and third stages as shown in figure 1.2. The outputs of First stage are passed on as inputs to the 2nd stage for further processing as shown in figure2.0. The outputs of 2nd stage are passed to 3rd stage which gives the energy spectrum points.

Stage Controller: This is the control block of the reconfigurable FFT unit. This block controls the operation of the each subunits by the generation of the control signals for the individually blocks. It takes the status of each block for the generation of control signal.

Energy RAM: This block is a memory element which stores the obtained energy points. The data stored used to the Psychoacoustic model for calculation of SMR (threshold masking energies).

2.1 Operational description:

The 16 bit sample points are fed to the input RAM block where these points get stored in the specified locations the first two and last two locations of the input RAM are stored with zero's to consider the offset of 2 and centering effect. Once the input to the RAM get stored fully i.e., 12 samples are passed down , input RAM generating status to the stage controller to enabling the speed controller operation ,simultaneously input RAM passes the 16 samples to the FFT block. The FFT is implemented as 16 point floating point parallel FFT. The input samples The 16 bit sample points are fed to are stored in RAM [6] at each clock event in a RAM , thus the each packet of speech constituting 12 samples are stored in RAM, after the 12th sample is stored, all the data in RAM are applied to the FFT in parallel. The operation of FFT involves three stage operations and constitutes multitude of multipliers and adders. 4 X 4

butterfly forms basis component which is composed of floating twin-precision multipliers and floating adders, there are altogether 4 basis butterflies, these butterflies are used in both in stage 1 and stage 2 with the output of stage 1 as input to stage 2 using feedback, in stage 3 the output of stage 2 is used to compute the energy of 16 points of the FFT operation. These adders and multipliers are coded in structural style. Stage controller gives the stage count as '01' to the FFT block to indicate the stage to be operated. Once stage 1 has completed its operation the stage unit generates the done signal to the controller unit indicating the completion of the first stage operation.

This results in the increment of the stage count to '10'. Similarly stage 2 followed by the stage 3 operation get carried out. Once done3 is generated the output of the FFT block is passed down to the energy RAM of 16 points. This process iterations are repeated for complete set of speech data passes to the module.

The reconfigurable FFT is implemented as 16 point floating point parallel FFT. The input samples are stored in RAM at each clock event in a RAM with an offset of 2 and centering, thus the each packet of speech constituting 12 samples are stored in RAM, after the 12^{th} sample is stored, all the data in RAM are applied to the FFT in parallel. The operation of FFT involves three stage operations and constitutes multitude of multipliers and adders. 4 X 4 butterfly forms basis component which is composed of floating multipliers and floating adders; there are altogether 4 basis butterflies, these butterflies are used in both in stage 1 and stage 2 with the output of stage 1 as input to stage 2 using feedback, in stage 3 the output of stage 2 is used to compute the energy of 16 points of the FFT operation. These adders and multipliers are coded in structural style. Once stage 1 has completed its operation the stage unit generates the done signal to the controller unit indicating the completion of the first stage operation. This results in the increment of the stage count to '10'. Similarly stage 2 followed by the stage 3 operation get carried out. Once done signal 3 is generated the output of the FFT block is passed down to the energy RAM of 16 points. This process iterations are repeated for complete set of speech data passes to the module. The FFT points are available after third stage and these points are stored in RAM in parallel these points are used to calculate energy of respective points and store them in Energy RAM. This will be used to calculate the masking threshold and signal to masking ratio required in Psychoacoustic model. Of all the components floating point adder and twin-precision multipliers form the core basis components, which are intended to operate on floating data, which components are coded in RTL style of coding.

3. SIMULATION RESULTS

To verify reusable FFT engine design, the randomized test-vector values from Matlab [7] were imported to the VHDL 8.i [8] simulator. The output results of our FFT engine were then compared with the reference results. Due to the complex-valued representation, a 32-bit calculation actually is made up of a 16-bit real and a 16-bit imaginary calculation, in the same way as a 16-bit calculation is represented by an 8-bit real and an 8-bit imaginary one.

The design was synthesized with Design Compiler [9], using a cell library of a commercial 0.13-µm CMOS technology. To obtain estimations on timing and power dissipation, Prime-Time [10] and Prime Power [11] were used.

Delay: Table 3.0 shows the delay for a reconfigurable 32-bit butterfly compared to a dedicated 32-bit butterfly and a dedicated 16bit butterfly.

The delay in the reconfigurable butterfly has two different values depending on in which mode it is operating. When the butterfly is operating in reconfigurable mode, two 16-bit butterflies are calculated simultaneously.

In this mode, the critical path is 4.14 ns this is because the multiplier and the adders is substantially shorter in this mode. Since the 32-bit reconfigurable butterfly can calculate two 16-bit butterflies simultaneously, the throughput is higher in the reconfigurable
butterfly than in the dedicated 16-bit butterfly.

1. **Area:** The area required for a reconfigurable butterfly is more compared to dedicated butterfly. The extra logic area is about 21% more for reconfigurable butterfly as shown in table 3.

Table 3.0 Delay, Area, & Energy of butterfly element

parameter	32-bit reconfigurable Butterfly	Dedicated Butterfly	
		32 bit	16 bit
Delay (ns)	4.14	4.5	2.65
Area	83582	69548	20158
Energy (nJ)	0.081	0.179	0.054

2. **Energy:** The total electric energy dissipation for one butterfly operation is shown in Table 3.0. The energy has been calculated by running power simulations, in which the dedicated butterflies were operated at maximum speed while the reconfigurable butterfly was operated at its maximum single precision mode. Because the FFT algorithm operates with iteration over the butterfly, values are read and written to a local memory many times throughout the FFT calculation. This reading and writing to the RAM is a great part of the whole energy dissipation in the FFT calculation. By using the twin-precision technique, the number of memory accesses in the calculation phase can be reduced by 50% compared to a dedicated full-precision butterfly.

This will reduce the energy dissipation substantially as shown in the figure 3.0 Thus, considering the ratio of low- to full-precision operations, the break-even point when the usage of a twin-precision FFT makes sense is reduced further.

CONCLUSION

A reconfigurable FFT, the calculation core of the design, the butterfly unit, is shown to be of significance for the final implementation quality in aspects of area, delay and power dissipation. The 32-bit reconfigurable precision butterfly has been compared to both a dedicated 32-bit butterfly and a dedicated 16-bit butterfly. When calculating at least 27% half precision calculations, we can effect savings in electric energy dissipation by using a flexible reconfigurable butterfly, instead of one or two dedicated butterflies.

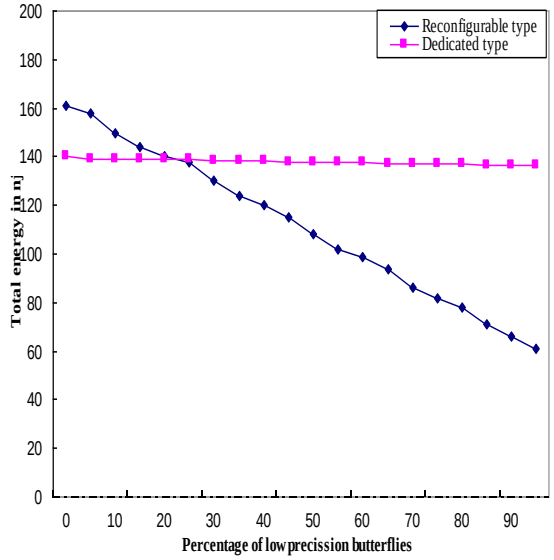

Fig. 3.0 Total energy

The reconfigurable butterfly makes it possible to calculate two half-precision butterflies simultaneously with half the number of accesses to the RAM during calculation, compared to a dedicated full precision butterfly.

REFERENCES

[1] J. W. Cooley and J. W. Tukey, "An Algorithm for the Machine Calculation ofComplex Fourier Series," Math. Compute, vol. 19, pp. 297–301, 1965.

[2] T. painter and A. spanias, "perceptual coding of digital audio" proc. of IEEE,vol.88, no. 4, pp. 451-513, Apr,2000.

[3] M. Sjalander, "Efficient Reconfigurable Multipliers Based on the Twin-Precision Technique," Licentiate of Engineering Thesis, Chalmers University of Technology, 2006.

[4] Leonardo L.de Oliveira, Eduardo. C.Sergio B. Array Hybrid Multiplier Versus Modified Booth Multiplier: Comparing Area and Power Consumption of Layout Implementations of Signed Radix-4 Architectures.

[5] M. Sjalander, H. Eriksson, and P.Larsson-Edefors, "An Efficient Twin-Precision Multiplier," in IEEE International Conferenc on Computer Design, 2004, pp.507–510.

[6] M. Brinck and K. Eklund, "A Flexible FFT/DCT Engine Using the Twin-Precision Technique," Master's thesis, Chalmers University of Technology, 2006.

[7] Matlab version 7.1.

[8] Modelsim SE PLUS 6i.

[9] Design Compiler Users Guide Version 2004.12.

[10] PrimeTime Users Guide X-2005.06.

[11] PrimePowerManualW-2004.12.

[12] David J.Defatta, Joseph G.Lucas, William S. Hodgkiss., Digital Signal Processing A System Design Approach, WI INC

[13] Stefan Brown, Zvonko vranesic., Fundamentals of digital logic with VHDL design, Mc Graw-Hill.