VLSI Reconfiguration Architecture of Radix-2 FFT on 130nm Technology

Pham Dang Lam, Nguyen Trong Ngo Nhat Du, Ngo Thanh Dat, Hoang Trang
Faculty of Electrics & Electronics,
Ho Chi Minh city University of Technology
Email: lamd.pham@hcmut.edu.vn, nhatdu.bentre@gmail.com, thanhdat5494@gmail.com, hoangtrang@hcm.edu.vn

Abstract**—** It is irrefutable fact that Fast Fourier Transform (FFT) has applied popularly to a wide range of applications such as Orthogonal Frequency Division Multiplexing (OFDM), Mel Frequency Cepstral Coefficients known as one of methods applying to audio feature extraction in speech recognition systems, and FFT architecture has become an essential component in any digital signal systems. Currently, almost researches have approached FFT feature through software solutions due to flexibility or high accuracy that are also drawbacks as regards hardware architecture while hardware approaches have presented high performance in terms of real-time applications and ability of integration in applied circuits. By approaching hardware development, an effective hardware architecture of radix-2 FFT, in which not only a great variety of FFT points can be reconfigured but high accuracy is also satisfied mainly because of applying floating point operations, is proposed to be able to tackle issues realting to dynamic FFT points and accuracy. Such this structure follows Application-Specific Integrated Circuit (ASIC) design flow so that it can be integrated in any system feasibly.

Keywords*- FFT (Fast Fourier Transform), MDC (Multipath Delay Commutator), OFDM (Orthogonal Frequency Division Multi-plexing), MFCC (Mel Frequency Cepstral Coefficients), dấu chấm động*

I. INTRODUCTION

With increasing development of digital signal processing field, FFT has become essential component so that such FFT feature need to be concerned as regards convenient architecture as well as high accuracy. It has experienced that a majority number of FFT algorithms have approached basing on software applications accompanying certain high performance computers [1]. As regards hardware approach, because of high requirements of ability of integration as well as low cost of implementation, a considerable number of hardware structure of FFT have proposed specially to be able to integrate in MFFC as well as OFDM architectures. Particularly, current survey [2, 3, 4] in table 1 has shown that authors proposed different FFT configurations that is reasonable for obtaining high performance of whole MFCC while another survey [5, 6, 7, 8, 9] described in table 2 witnessed a narrow range of FFT points with the highest number of FFT points recoded at 256. Through such surveys, it can be concluded that significant change of the number of FFT points bases on certain applications or to be able to achieve final high performance of whole system. The second worth mentioned is that hardware architectures of FFT have referred to small number of FFT points causing of cost of hardware-base implementation. In order to cover such current issues, a dynamic architecture of FFT with ability of reconfiguring FFT points from 8 up to 4096 as well as integration of 32-bit floating point operations flowing IEEE 754 standard is proposed in this research.

The rest of paper is organized follows. Section II introduces detailed architecture of proposed. Next, Section III experiences obtained results regarding to highest frequency, area report and comparison to other designs, and Section IV concludes the paper and future work.

Table 1 FFT configuration applied in MFCC hardware architecture

Authors	Number of FFT points	Hardware approach	Integrated in whole structure	Timing consumption
GIN-DER WU [2]	256 (radix-2 FFT)	ASIC (0.18µm)	MFCC in speech recognition	10,4 µs
Chin-Teng Lin [3]	256 (radix-16 FFT)	ASIC (0.13µm)	MFCC in speech recognition	-
Dongsuk Jeon [4]	1024 (radix-4 FFT)	ASIC (65 nm)	MFCC in speech recognition	6,7 µs

Table 2 FFT configuration applied in OFDM system

Author	Number of FFT points	Hardware approach	How to implement	Integrated in whole tructure	Timing consumption
Lihong Jia [5]	128 (radix-2/4/8 FFT)	ASIC (0.6µm)	Pipeline	OFDM	3 µs
Atin Mukherjee [6]	8 (radix-2 FFT)	FPGA (Xilinx Virtx-6)	Butterfly	OFDM	19,598 ns

Jungmin Park[7]	64 – 8K (radix-8 FFT)	FPGA (Xilinx Virtex-5)	Butterfly	OFDM	0,33 µs (64-point FFT) -- 96,20 µs (8000-point FFT)
K. Umapathy[8]	128 (radix-2/4 FFT)	ASIC (90 nm)	Pipeline	MIMO-OFDM	40 µs
Ediz Çetin[9]	256 (radix-2 FFT)	ASIC (0.7µm)	Butterly	OFDM	102,4 µs

II. PROPOSED HARDWARE ARCHITECTURE OF FFT

2.1 FFT algorithm

FFT is one of algorithms applied to calculate DFT quickly in almost digital system popularly. Currently, there are two ways of calculating DFT that base on timing domain or frequency domain in which different radix such as 2, 4, 8 or higher radixes can be selected optionally. Such proposed architecture selects radix-2 configuration in timing domain applied in almost digital system as previous surveys in table 1 and table 2. From such this approach, DFT with N points will be implemented though FFT with m stages where $N = 2^m$. Mathematically, DFT is transferred to sum of two components in which the first component is DFT for odd number while another is even one presented from the first to the fifth statement.

$$X[k] = \sum_{n=0}^{N-1} x(n) W_N^{nk} \quad ,k = 0,1,2,\dots,N-1$$
$$\text{với } W_N^{nk} = e^{-j\left(\frac{2\pi}{N}\right)nk} \tag{1}$$

$$= \sum_{n=0}^{\frac{N}{2}-1} x(2n) e^{-\left(j\frac{2\pi\times(2n)k}{N}\right)} + \sum_{n=0}^{\frac{N}{2}-1} x(2n+1) e^{-\left(j\frac{2\pi\times(2n+1)k}{N}\right)} \tag{2}$$

$$= \sum_{n=0}^{\frac{N}{2}-1} x(2n) e^{-\left(j\frac{2\pi nk}{\frac{N}{2}}\right)} + e^{-\left(j\frac{2\pi k}{N}\right)} \sum_{n=0}^{\frac{N}{2}-1} x(2n+1) e^{-\left(j\frac{2\pi nk}{\frac{N}{2}}\right)} \tag{3}$$

$$= DFT_{\frac{N}{2}}\left[\left[x(0), x(2), \dots, x(N-2)\right]\right] + W_N^k DFT_{\frac{N}{2}}\left[\left[x(1), x(3), \dots, x(N-1)\right]\right] \tag{4}$$

$$= DFT_{\frac{N}{2}}\left[x_{even}(n)\right] + W_N^k DFT_{\frac{N}{2}}\left[x_{odd}(n)\right] \tag{5}$$

According to the fifth statement, butterfly diagram for calculating DFT is proposed as figure 1 bellow in which there are totally m stages. In every stage, different pairs of inputs are calculated basing on rule of butterfly diagram so that total operations after all stages perform statement 5. The operations between two FFT-point values are similar in any butterfly unit as figure 2.

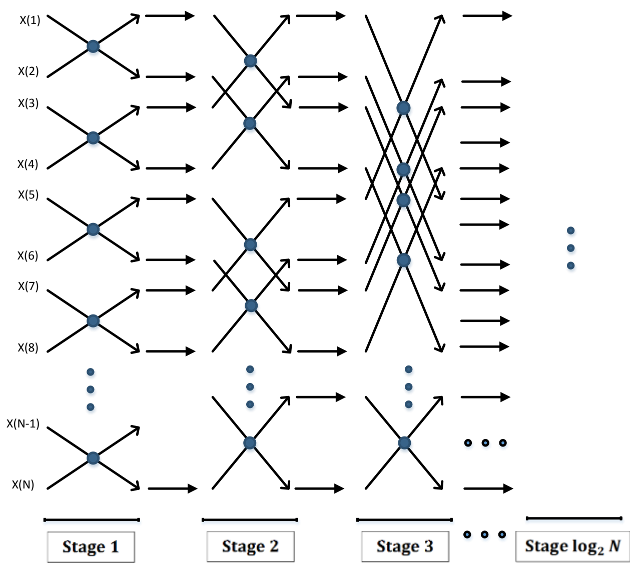
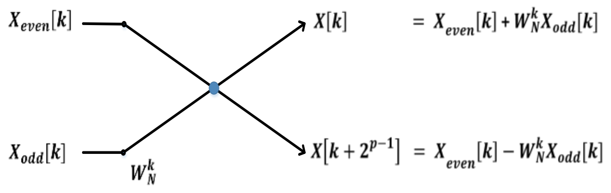


Figure 1 Butterfly diagram for N-point FFT



$X_{even}[k]$
$X_{odd}[k]$
W_N^k
$X[k] \quad = X_{even}[k] + W_N^k X_{odd}[k]$
$X[k+2^{p-1}] = X_{even}[k] - W_N^k X_{odd}[k]$

Figure 2 Operations inside butterfly unit

2.2 Proposed FFT architecture

According to butterfly diagram, it is fact that every butterfly unit is called many times in every stage and that is similar in other stages, but every butterfly unit receives different inputs accompanying different weights. Form such circumstance, an effective FFT architecture in which all operations for a butterfly unit is called FFT core and an FFT-controller structure keeps the role of controlling suitable data inputs transferring into butterfly unit is proposed. As regard flow of input data, initially both value of N points and weight W_N^k, initial integer values of N point is loaded from Input_Memory to *MEM_INIT_1* and *MEM_INIT_4* to be able to calculate for the first stage of butterfly diagram. During operation in every butterfly unit inside FFT core, weight W_N^k is read from *W-Real* and *W_image* memories simultaneously. Real and Image results for every butterfly unit will be written into *MEM_INT_3* and *MEM_IN_6* memories. After finishing all butterfly units per stage and storing all results into such two memories, all data will be read out and written into *MEM_INT_1, MEM_INT_2, MEM_INT_4* and *MEM_INT_5* memories to be used for next stage's operations. In every stage, FFT core is called N/2 times corresponding to N points. Basing on operations of all internal memories, interface of proposed FFT architecture is described in figure 3, figure 4 and table 3.

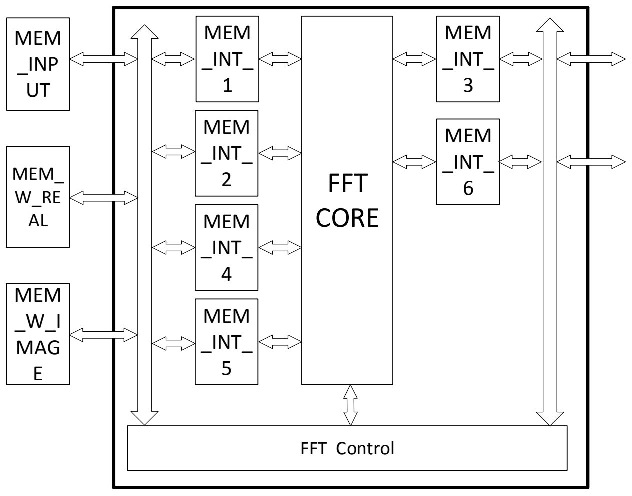


Figure 3. Block diagram of proposed FFT

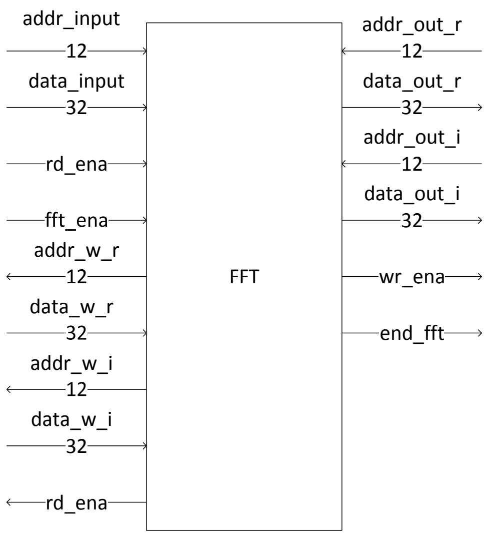


Figure 4. Interface of proposed FFT architecture

Table 3. Detailed interface of proposed FFT architecture

Name	**Type**	**Bit Number**	**Describe**
addr_input	input	12	Address for FFT input data
data_input	input	32	FFT input data
rd_ena_input	input	1	Enable signal for FFT input data
fft_ena	input	1	Enable signal for complete FFT
addr_w_r	input	12	Address for real value of weight
data_w_r	input	32	Real value of weight
addr_w_i	input	12	Address for image value of weight
data_w_i	input	32	Image value of weight
rd_ena_w	output	1	Enable signal for weight input data
addr_out_r	output	12	Address for real value of FFT output data
data_out_r	output	32	Real value of FFT output data
addr_out_i	output	12	Address for image value of FFT output data
data_out_i	output	32	Image value of FFT output data
wr_ena	output	1	Enable signal to write data into internal memories
end_fft	input	1	Notify FFT finished

In order to read or write data during operating butterfly unit, a controller is considered as figure 5.

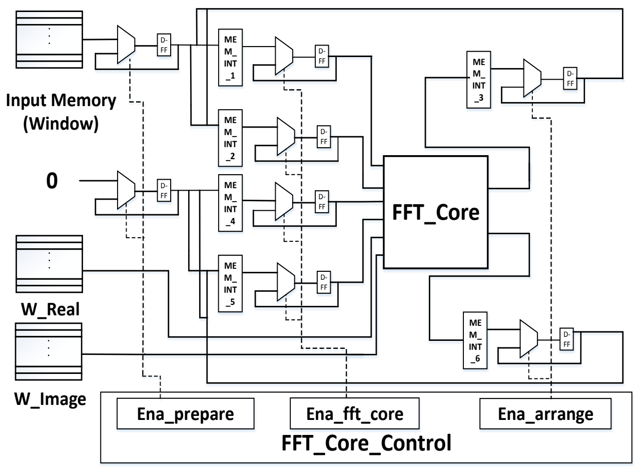


Figure 5 FFT controller

There are three main states comprising of initial prepare for input data, FFT core operation and rearrangement after every stage. Detailed states and related output signals are described in figure 6 and table 4, 5, 6.

Table 4 Function of states in FFT block

Present state	**Function**
RESET	**Reset a system**
INITIAL	**Initial state of internal memories reading**
PREPARE	**Implement the bit-reserved algotithm for the input data**
FFT_CORE	**Implement the butterfly computation**
WAIT	**Until a butterfly computation has been finished**
ARRANGE	**Implement the arrangement of output data of**

	butterfly computation

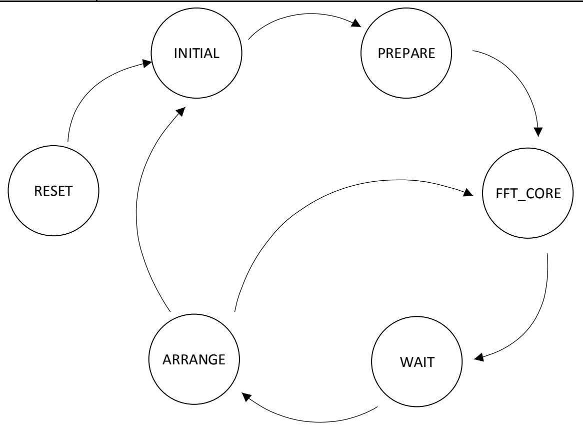


Figure 5 State machine of FFT with N points.

Table 5 State machine in FFT block

Present state	**Next state**	**Condition**
RESET	**INITIAL**	**When receive an enable signal from Main Control**
INITIAL	**PREPARE**	**When receive an enable signal from Main Control**
PREPARE	**FFT_CORE**	**When receive an enable signal from Main Control**
FFT_CORE	**WAIT**	**Butterfly computation has already done (The counter is over)**
WAIT	**ARRANGE**	**After 1 clock**
ARRANGE	**FFT_CORE**	**1 stage has not yet done**
	INITIAL	**Final stage finishes (The counter is over)**

Table 6 Function of internal memories inside FFT

Interal Memory	**Description**
MEM_INPUT	**Contain the value of input data (just has the real part)**
0	**The imaginary part of input data is zero at initial sate**
MEM_INT_1	**Contain the real part of the first complex number participates in butter fly computation**
MEM_INT_2	**Contain the real part of the second complex number participates in butter fly computation**
MEM_INT_4	**Contain the imaginary part of the first complex number participates in butter fly computation**
MEM_INT_5	**Contain the imaginary part of the second complex number participates in butter fly computation**
MEM_INT_3	**Contain the real part of the complex result number after butter fly computation**
MEM_INT_6	**Contain the imaginary part of the complex result number after butter fly computation**
W_REAL	**Contain the real part of a twiddle factor**
W_IMAGE	**Contain the imaginary part of a twiddle factor**

III. EXPERIMENTS AND RESULTS

According to ASIC design flow, initially FFT architecture is developed by Verilog HDL in Register Transfer level, followed by RTL verification with Matlab model to confirm mean of absolute error as table 7.

Table 7 Mean of absolute error between Verilog result and Matlab model

FFT points	**Mean of absolute error for real value**	**Mean of absolute error for image value**
8	$1,034 .10^{-7}$	$2,649 .10^{-7}$
16	$2,915 .10^{-7}$	$6,786 .10^{-7}$
32	$5,278 .10^{-8}$	$2,930 .10^{-8}$
64	$1,286 .10^{-7}$	$2,459 .10^{-8}$
128	$3,083 .10^{-7}$	$1,326 .10^{-8}$
256	$1,009 .10^{-7}$	$1,351 .10^{-8}$
512	$1,107 .10^{-7}$	$7,432 .10^{-8}$
1024	$1,138 .10^{-7}$	$9,462 .10^{-9}$

Where the formulate

$$E = \frac{\sum_{i}^{n} / x_i - y_i /}{n} \tag{6}$$

For the number of FFT points from 8 to 4096, table 8 presents the number of cycles and corresponding timing consumption with achieved frequency at 500 MHz. According to FFT architecture and obtained waveforms, timing consumption can be calculated as statement 7.

Table 8 Timing consumption

Number of FFT points	**The total cycle consumption**	**The total timing consumption (ns)**
8	780	1560
16	1710	3420
32	3760	7520
64	8370	16740
128	18740	37480
256	41910	83820
512	93240	186480
1024	206010	412020
2048	451900	903800
4096	984510	1969020

$$
\begin{aligned}
T &= 10\times[T_{FIRST\ ARRANGE} + (\log_2 N - 2) \\
&\quad \times T_{BUTTERFLY\&WAITE\&LOOP\ ARRANGE} + T_{LOOP\ ARRANGE}] \\
&= 10\left[(3N+15)+(\log_2 N - 2)\times(2N+13)+(N+2)\right] \\
&= 10[(2N+13)\times\log_2 N - 9]
\end{aligned} \tag{7}
$$

Table 9 Comparison to other designs

Authors	Target Hardware and Technology	Number of FFT points	Achieved Frequency (MHz)	Power (mW)	Timing consumption
GIN-DER WU [2]	ASIC (0.18µm)	256	100	89,18	10.4 µs
Chin-Teng Lin [3]	ASIC (0.13µm)	256	100	22.37	-
Dongsuk Jeon [4]	ASIC (65 nm)	1024	19	-	6,7 µs
Lihong Jia [5]	ASIC (0.6µm)	128	50	400	3 µs
Atin Mukherjee [6]	FPGA (Xilinx Virtx-6)	8	51	-	19.598 ns
K. Umapathy [8]	ASIC (90 nm)	128	40	-	40 µs
Ediz Çetin [9]	ASIC (0.7µm)	256	40	-	102,4 µs
FFT đề nghị	ASIC (130nm)	8-4096	500	3.44	1.5µs-1.969ms

Basing on the table 9 and statement 7, the number of cycle consumption is up to the number of FFT points. However with the achieved highest frequency at 500 MHz, timing consumption is very small with the recorded figure at only 2ms for the largest configuration reported at 4096 points that satisfies any real-time applications. In addition, result of synthesis on target technology at 130nm not only shows high performance in comparison to other designs but it also proves high flexibility with a wide range of number of FFT points. One of concerned issue is that such architecture need considerable number of internal memories that also constrains both area cost and limitation of FFT points. In this proposed architecture 128Kb memory for maximum FFT points at 4096 for every internal memory is compatible to current state of art embedded memory integrated in chip.

IV. CONCLUTION

In this research, an effective hardware architecture of FFT in which user can reconfigure the number of FFT points by input data through internal memory is proposed. According to proposed design, not only flexibility is very compatible to a wide range of application but it has experienced high performance compared with other hardware design. Such this FFFT is the first phase of complete dynamic MFCC architecture that will be included in ASR system in the future.

***Acknowledgement** - The research is funded by Vietnam Government through project with KC.01.23/11-15*

REFERENCES

[1] Teo Cupaiuolo, Daniele Lo Iacono, "A Flexible and Fast Software Implementation of FFT on the BPE platform" in Design, Automation & Test in Europe Conference & Exhibition, March 2012, pp.1467-1470.

[2] Gin-der Wu, Ying Lei, "A Register Array Based Low Power FFT Processor" in Journal of Information Science and Engineering, vol.24, Issue 3, pp. 981-991, 2008.

[3] Chin-Teng Lin, Yuan-Chu Yu, Lan-Da Van, "Cost-Effective Triple-Mode Reconfigurable Pipeline" in IEEE Transactions On Very Large Scale Integration (VLSI) Systems, vol. 16, no. 8, pp. 1058-1071, 2008.

[4] Dongsuk Jeon, Mingoo Seok, Chaitali Chakrabarti, David Blaauw, Dennis Sylvester, "Energy-Optimized High Performance FFT Processor" in ICASSP, 2011, pp. 1701-1704.

[5] Lihong Jia, Bingxin Li, Yonghong Gao, Hannu Tenhunen, "Implementation of A Low Power 128-Point FFT" in Solid-State and Integrated Circuit Technology, Beijing, 1998, pp.369-372.

[6] Atin Mukherjee, Amitabha Sinha, Debesh Choudhury, "A Novel Architecture of Area Efficient FFT" in ACM SIGARCH Computer Architecture News, December 2014.

[7] Jungmin Park, "Design of a radix-8/4/2 FFT processor for OFDM" in Iowa State University of Science and Technology, Ames, Iowa, 2011.

[8] K. Umapathy, D. Rajaveerappa, "Low Power 128-Point Pipeline FFT Processor using Mixed Radix 4/2 for MIMO OFDM Systems" in International Journal of Soft Computing and Engineering (IJSCE), vol. 2, no. 5, pp. 177-179, November 2008.

[9] Ediz Çetin, Richard C. S. Morling, Izzet Kale, "An Integrated 256-point Complex FFT Processor for Real-time Spectrum Analysis and Measurement" in IEEE Proceedings of Instrumentation and MeasurementTechnology Conference, Canada, May 1997, pp.96-101.