Designs and implementations of ultra-low power, low voltage digital signal processors

Ngoc, Le Ba

2018

Ngoc, L. B. (2018). Designs and implementations of ultra-low power, low voltage digital signal processors. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/89232>

<https://doi.org/10.32657/10220/46221>

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

**DESIGNS AND IMPLEMENTATIONS OF
ULTRA-LOW POWER, LOW VOLTAGE
DIGITAL SIGNAL PROCESSORS**

**LE BA NGOC
SCHOOL OF ELECTRICAL AND ELECTRONIC
ENGINEERING
2018**

**DESIGNS AND IMPLEMENTATIONS
OF ULTRA-LOW POWER, LOW
VOLTAGE DIGITAL SIGNAL
PROCESSORS**

LE BA NGOC

LE BA NGOC

SCHOOL OF ELECTRICAL AND ELECTRONIC ENGINEERING

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirement for the degree of
Doctor of Philosophy

2018

Acknowledgement

This thesis is the end of my joyful journey in obtaining the Doctor of Philosophy degree. Throughout the time, I have been so lucky to be around with so many wonderful people: my professors, my colleagues, my friends, and my family. They have made this journey so memorable with words of encouragement, endless help, fun, and joys. To them, I am deeply indebted.

It is difficult to express my gratitude to my research advisor, Professor Tony T. Kim. His enthusiasm, his inspiration, and his great efforts to explain and help me in my research have encouraged me all the time and brought me over every difficulty I met. During my research, Prof. Kim was always accessible and ready with advice. Whenever I needed helps, he provided with encouragement, good teaching, good company, and lots of excellent ideas. It is my great pleasure and honor to be his student.

I wish to thank the moderators for my Qualifying Examination (QE), Prof. Siek Liter and Prof. Lim Meng Hiot for examining my research works. Their comments and suggestions helped to improve my approach and open up new ideas.

I would like to thank Mr. Sia Liang Poo, Mr. Goh Mia Yong, Jimmy, Ms. Gek Eng and Mr. David Robert for their assistance and technical help. Their fantastic friendliness has created such pleasing environment in Virtus and made me enjoy spending days and nights in the lab.

Finally, I am grateful to countless people who shared with me their difficulties, their knowledge, and their stories. Life would not so pleasant without them.

Table of Contents

Acknowledgement	i
Abstract.....	vi
List of Figures.....	viii
List of Tables	xiv
Chapter 1 Introduction	1
1.1 Motivations	1
1.2 Objectives	3
1.3 Contributions of the Thesis	4
1.4 Organization of the Thesis	5
Chapter 2 Fast Fourier Transform	7
2.1 Discrete Fourier Transform (DFT)	7
2.2 Fast Fourier Transform (FFT) Algorithms	8
2.2.1 Radix-2 Algorithm.....	9
2.2.2 Radix-4 and Higher Radix Algorithms.....	10
2.2.3 Radix-2 ^k Algorithm.....	12
2.3 Fast Fourier Transform (FFT) Architecture.....	14
2.3.1 Delay Commutator Architecture	15
2.3.2 Delay Feedback Architecture	16
2.4 Review and Comparison of existing FFT topologies	18
2.5 Conclusions.....	22

Chapter 3	A 1024-point Low Power High-Performance Radix-2 ² Feed-Forward FFT Processor	24
3.1	Motivations.....	24
3.2	Proposed Feedforward Architecture.....	25
3.3	Proposed Input Scheduling Algorithm	29
3.4	Implementation Details	31
3.4.1	Architecture Optimization.....	31
3.4.2	Butterfly Design	35
3.4.3	Commutator and Memory Design.....	36
3.4.4	Optimum Energy Point	39
3.5	Simulation and Measurement Results	42
3.5.1	Simulation Results	42
3.5.2	Measurement Results	45
3.6	Conclusions	48
Chapter 4	Hand Gesture Sensing Techniques and Available Detecting Algorithms	51
4.1	Motivations.....	51
4.2	Sensing techniques for hand gesture recognition	52
4.2.1	Image sensing.....	53
4.2.2	RF Sensing	53
4.2.3	Ultrasound sensing.....	54

4.2.4	Infrared (IR) sensing.....	56
4.3	Hand Gesture Detecting Algorithms.....	57
4.3.1	Time-of-fly Algorithm.....	57
4.3.2	Template Matching Algorithm	58
4.3.3	Neural Network and Pattern Matching.....	59
Chapter 5	A 256 pixel, 66.8 μ W Infrared Gesture Recognition Processor for Smart Devices	61
5.1	Proposed System Architecture for Hand Gesture Recognition SoC.....	61
5.2	Proposed Hand Gesture Recognition Digital Signal Processor	63
5.3	Gesture Recognition Algorithms	66
5.3.1	Calibration and Motion Detection	66
5.3.2	Wake-up Gesture Detection.....	68
5.3.3	Zooming Detection	70
5.3.4	Sweeping Detection.....	72
5.4	Implementations and Measurement Results	75
5.4.1	Test results for the 16 $\times$ 4 IR sensory array.....	77
5.4.2	Test results for the 16 $\times$ 16 IR sensory array.....	79
Chapter 6	Conclusions and Future Works.....	84
6.1	Conclusions.....	84
6.2	Future Works	86
Author's Publications		88

References 89

Abstract

Conventional Fast Fourier Transform (FFT) using Radix-2² brought a significant improvement in FFT implementations to reduce circuit complexity and computational power. One of the most well-known architectures for VLSI implementation in Radix-2² is Single-path Delay Feedback (SDF) which has simple architecture and requires the smallest amount of memory. However, low throughput due to the single data line is its major drawback. In the first part of this thesis, Radix-2² Multiple-path Delay Commutator (MDC) architecture is proposed to achieve higher throughput while preserving low memory and hardware usage to reduce energy per conversion. In addition, a new input scheduling algorithm, parallel-pipelined architecture together with ultra-low power circuit design techniques are employed to increase the speed and minimize the total power consumption. A 1024-point high speed, ultra-low power Fast Fourier Transform (FFT) was fabricated using STM 65nm technology. The chip consumes 77.2 nJ/FFT with clock frequency of 400 MHz at 0.6V supply and it is able to operate at up to 600MHz at V_{DD}=1V yielding 1.2 Gsample/s. Compared with the existing FFT research works, our proposed FFT processor achieved 8% lower than the lowest reported energy per FFT conversion in high performance domain (Msample and Gsample/s) while occupies only 42% of its area.

In addition to the FFT processor, an ultra-low power hand gesture recognition processor is designed and implemented to further elaborate power saving techniques for system-on-chip implementations. In the past, many hand gesture recognition systems have been invented both for research interest and market demand. These systems require a great amount of power because of using active devices such as

active IR sensor, RF sensor or Ultrasonic sensor which are not suitable for low power and portable applications. In our research, passive infrared (PIR) sensor is proposed to avoid high power demand. New algorithms for gesture recognition including sweeping, zooming and wake-up gesture detection is also presented and verified with real-time input from a customized analog front end as well as recorded input from a Heiman sensor. To further reduce power consumption, the algorithm is able to put the system into standby mode which operates at lower frequencies. Upon sensing a wake-up gesture, its active mode will be triggered to recover full operations. The completed hand gesture recognition SoC which includes an analog front end to process readings from IR sensor array together with two digital signal processors was fabricated in TSMC 65nm. The whole chip occupies an area of 8.1 mm² in which the 16×4 input array DSP takes 580×300 μm and the 16×16 input version uses 580×580 μm. Test chips demonstrate successfully gesture detections for 8 sweeping directions, zooming action and a wake-up gesture. In active mode, the total power consumption of the SoC is 260 μW and it is 46 μW in idle mode. The DSP consumes 28.6 μW for detecting 16×4 input and 66.8 μW for 16×16 input. Power and area specifications of the proposed SoC are suitable for mobile and smart device applications. To our best knowledge, this work marks the first SOC hand gesture recognition processor which is 6.6 times smaller area and 64 times higher resolution in comparison with the state-of-art passive IR sensor from Pyreos.

List of Figures

Figure 1: Fourier Transformation of a time domain signal into a frequency domain signal [7]. A time domain signal is, therefore, a combination of different sinusoidal frequency signals.	7
Figure 2: Radix-2 signal flow graph of an 8-point FFT [15] using divide-and-conquer technique for faster FFT calculations.	10
Figure 3: Butterfly for equation (2.5) from Radix-4 FFT algorithm [15]. Note that input data with a distance of $N/2$ are paired up.	11
Figure 4: A signal flow graph of a 16-point Radix-4 FFT. Note that FFT calculations are divided into four 4-Point FFTs.	12
Figure 5: A signal flow graph of a 16-point Radix- 2^2 FFT Algorithm. Note that butterflies in Radix- 2^2 have a similar architecture as in Radix-2 butterflies.	14
Figure 6: Radix-2 Multiple Delay Commutator Architecture for a 16-point FFT. Note that MDC architecture uses the same butterfly type II as presented in Radix- 2^2	16
Figure 7: Radix-2 Single delay Feedback Architecture for a 16-point FFT. In this architecture, butterflies' outputs are delayed to pair up with the input.	17
Figure 8: Radix-4 Single Delay Feedback Architecture for a 16-point FFT. Similar to Radix-2 DSF, Radix-4 SDF uses multiple delay line to delay butterflies' output.	17
Figure 9: An MDF FFT implementation. (a) The parallel concept of single delay feedback. (b) The actual implementation of a multiple delay feedback FFT.	18
Figure 10: Initial Design of a Radix-4 MDC FFT core. Note that this topology suffers from low utilization rate for multipliers.	19
Figure 11: A modified R4MDC architecture for a 1024-point FFT core. This architecture helps to increase multipliers' utilization rate.	19

Figure 12: (a) Butterfly type-I. (b) Butterfly type-II. (c) R2 ² SDF. Note that BF-II is similar to BF-I but with additional trivial multipliers.....	20
Figure 13: Power, area tradeoff for varying level of parallelism [30]. Note that the horizontal axis shows the level of parallelism while the vertical axis shows the normalized product of power and area (PAP).....	21
Figure 14: Conceptual design of Radix-22 Feedforward FFT topology. Note that commutator circuits are used instead of delay lines.	26
Figure 15: Traditional circuit block diagram for a commutator which provides a distance of N/2 at outputs.....	26
Figure 16: Proposed commutator architecture for Radix-2 ² Feedforward FFT. Note that memory base is used to replace the delay lines.	29
Figure 17: Proposed Radix-2 ² Feedforward FFT architecture. Note that BFI is butterfly type I which is conventional butterfly and BFII is type II butterfly as explained in Figure 12.....	29
Figure 18: A conventional data shuffling approach [25]. In this example, inputs are coming in two parallel paths with a distance of 16 and it is required to have a distance of 4 for the two outputs.....	30
Figure 19: Data flow in our proposed input scheduling algorithm. Note that there is no shifting data and data is accessed directly by its address.....	31
Figure 20: Our proposed 1024-point 2-parallel data path Radix-2 ² feedforward FFT processor block diagram.	32
Figure 21: Parallel-pipelined topologies which are shown in [33]. (a) 2 parallel data paths. (b) 4 parallel data paths.	33

Figure 22: Benefit of parallelism at different percentage of common parts. In this figure, comparison for circuits with different percentage of the common parts is performed and it is clear that with increasing in common parts, parallelism provides much lesser benefit.	34
Figure 23: Butterfly implementation. (a) BF-I. (b) BF-II. Note that BF-I is a trivial Radix-2 butterfly while BF-II requires complex multipliers and modifications in the data path.....	36
Figure 24: Hardware block diagram of our proposed input scheduler. Note that the number of bits in the counter will determine the distance at the output.....	38
Figure 25: Latch based memory design. Note that Read and Write operations can be performed in the same clock cycle.	38
Figure 26: Layout area comparison between register-based memory on the left and an SRAM on the right. The 16-bytes register-based memory requires an area of $160 \times 140 \mu\text{m}$ while it is $73 \times 30 \mu\text{m}$ for a double size SRAM (32-bytes).....	39
Figure 27: Timing diagram in the 5 th stage of our proposed FFT core. Note that it takes n-first clocks to fill data into the memory with size n and after that, a continuous flow is maintained.....	39
Figure 28: Energy characterization curve as a function of supply voltage. Note that at the subthreshold region, dynamic energy is reduced with lowering supply voltage however leakage energy is increased exponentially, therefore, the total energy is increased at an exponential rate when the supply voltage is decreased in the subthreshold region.....	40
Figure 29: Optimum operating point of the devices. (a) Circuit energy consumption at different supply voltages. (b) Circuit delay at different supply voltages. Note that	

$V_{DD} = 0.4$ V is chosen as the operating voltage to achieve minimum energy point and	
$V_{DD} = 0.6$ V is chosen as the best circuit performance operating point.....	41
Figure 30: A 1024-point discrete input signal used for testing: $y(t) = 0.7 \times \sin(100\pi t) + j \times 0.3 \times \sin(300\pi t)$	43
Figure 31: FFT Signal Spectrum. (a) From an embedded Matlab function. (b) From our proposed FFT topology.....	43
Figure 32: ModelSim simulation results for our proposed FFT core. Note that START signal is used to indicate the beginning of new FFT stage.....	44
Figure 33: (a) FFT chip layout design in Cadence Virtuoso. (b) Micrograph of the proposed FFT processor.....	45
Figure 34: FFT output spectrum. (a) Our proposed FFT core with non-uniform bit width. (b) Simulated FFT core with uniform bit width. Note that the output spectrum of uniform bit width design is much noisier than non-uniform bit width design.	46
Figure 35: Measurement results. (a) Power consumption. (b) Maximum clock speed performance.	48
Figure 36: A Rangefinder using Ultrasound Sensors [46]. The working principle of this rangefinder is using TOF calculations to determine the distance.	55
Figure 37: A thermograph of hand gesture captured from a 16×16 PIR array with 4 built-in amplifiers using Molybdenum-gate CMOS with MEMS process [53].....	57
Figure 38: Gesture detection using template matching. (a) Standing up gesture. (b) Six templates to compare with the gesture [56]......	59
Figure 39: Proposed Hand Gesture Recognition SoC which consists of analog front-end to obtain input data from IR sensor and a digital signal processor to analyze input gestures.....	62

Figure 40: A thermograph captured by the Heiman HTPA 16×16d sensor array. Note that red pixels represent higher temperature to differentiate the object from ambient temperature in blue pixels.....	63
Figure 41: Block diagram of our proposed gesture recognition processor. Interface circuit consists of write circuits which are controlled by the control circuit to write 8-bit data into the three memories.	64
Figure 42: Top level design of detecting algorithms. Note that there are total 8 sweeping directions: 4 movements up/down, left/right in straight directions and 4 movements in diagonal directions.	65
Figure 43: The list of recognizable gestures (a) up-down (b) left-right (c) diagonal sweeping (c) zooming in/out (d) a unique wake-up gesture.....	65
Figure 44: Circuit block diagram for calibration module. Note that write and read signal are performed within a half clock cycle.....	68
Figure 45: (a) MHI thermograph without any object's presence or background image (b) MHI thermograph with object's presence.	68
Figure 46: (a) A Wake-up gesture. (b) Thermal graph of a wake-up gesture. Note that the wake-up gesture should be simple to be recognized but unique to avoid being misclassified by the DSP.	69
Figure 47: Zooming detection block diagram which consists of coordinate calculation circuit, corner checking circuit, and a comparator bank.....	71
Figure 48: A zooming action. Note that in the next frame, object shrinks inside its previous locations.	72
Figure 49: Sweeping detection module block diagram. Sweeping gestures are detectable in 8 directions.	74

Figure 50: (a) A vertical sweeping gesture. (b) A diagonal sweeping gesture. Note that the peaks in column sum and row sums will tell the sweeping direction.	74
Figure 51: Sweeping detection mechanism. Note that summation of pixels values in overlapping area is close to zero while it is magnified in the non-overlapping area.	75
Figure 52: Sweeping detection for a curly hand movement. Note that this movement can be recognized with one peak in row summation and two peaks in column summation.	75
Figure 53: The micrograph of our gesture recognition SoC. Note that processor area contains two version for 16×4 input data and 16×16 input data and the AFE part is not in the scope of this report.	77
Figure 54: (a) Measurement temperature of a hand sweeping left to right. (b) Response code from the DSP.	78
Figure 55: Power contribution of the hand gesture recognition SoC in active and power saving mode.	79
Figure 56: (a) Test image for a wake-up gesture with a closed loop highlighted in red (b) output detection with 3 adjacent detected pixels to indicate a closed loop.	80
Figure 57: (a) Output test results for zooming detection, zoom=11 indicates zooming out. (b) Output test results for sweeping detection, sweep=110 indicate a right to left sweep with 2 peaks in total. Note that in MHI image, blue color indicates object's previous position and red color indicates object's current position.	81
Figure 58: Detection accuracy for Right-Left sweeps at different sweeping speed and at different object's distance to the sensor.	82
Figure 59: Power consumption breakdown details. Note that dynamic energy and power consumption from memory circuits are major contributors.	83

List of Tables

Table I: Hardware and throughput comparison for different existing FFT processor architectures.....	22
Table II: FFT cores comparison between our proposed design and existing high throughput designs.....	48
Table III: Comparison with previous gesture detection works. Note that this work is the first gesture detection SoC that has on-chip processor instead of using software processing as in other reported works.	83

Chapter 1 Introduction

1.1 Motivations

Rapid developments in silicon technology have significantly increased transistor density in electronic circuits. According to Moore's law, the number of transistors integrated into a circuit doubles every year [1]. These developments have brought to inventions of many innovative products, however, circuits with exponential increase in transistor density also post a very challenging problem to designers for minimizing power consumption. With more transistors packed in a wafer, not only higher power supply is needed but also more heat will be dissipated from the system and that raises demands for higher-capacity power supplies and better cooling devices which are an additional cost to the final products. Besides, technology scaling reduces transistor size and shorten interconnect hence smaller capacitance to have faster transistor's switching speed. Historically, integrated circuit speed has increased roughly 30% at each new technology node and with doubled transistor density in one die. As a result, circuit power consumption becomes larger and requires better power-performance optimization [2].

In electronic applications, devices are moving to mobile platforms and biomedical electronics have gained a lot of interests. In these domains, smaller device sizes and long battery life are most critical due to constraints on battery and requirements for continuous operations. For portable and mobile devices which are mainly battery operated, working time depends largely on the battery capacity, however, enlarging battery is not a preferred option as it requires a bigger housing and reduces the portability of products. Lowering power consumption is, therefore, a

critical design requirement to achieve. For biomedical electronics, requirements for power consumption are even more stringent as devices may have to be implanted inside the human body through operations and it would be a life-threatening process if the patient has to undergo another operation to recharge the battery. Although current technology allows charging some devices wirelessly, low power is still a must design requirement to ensure prolonged service as well as avoiding interferences with the human body.

To satisfy constraints on power consumption, many circuit techniques have been discovered. As total power is a summation of dynamic, short-circuit and leakage power, any efforts to reduce these components will bring down the overall power. There are strategies to cut down circuit's power such as trading speed for power, finding lower power operating region and reduce wasted power presented in [3]. One of the very efficient methods for power reduction is operating at lower supply voltage since dynamic power is proportional to the square of voltage. By using ultra-low voltage techniques, circuits are able to function in sub-threshold regions however it becomes much harder to push the limit further as total power is dominated by leakage power in this region. This again leads to new research directions to improve circuit leakage both at the device level and circuit level, but it doesn't come at zero cost. At low voltage supply and leakage control, operating frequency is limited hence it becomes much more difficult for high-performance system design such as digital signal processors. Our research works focus on resolving this problem by incorporating state of the art ultra-low power circuit techniques and top-down approach design from system architecture and algorithms to optimize operations for higher energy efficiency.

This thesis is a part of our research group's efforts to build ultra-low power circuit library for the complex system on chips (SoCs). In this work, two ultra-low power, high-performance digital signal processors which are a 1024-point Fast Fourier Transform (FFT) processor and a hand gesture recognition processor for mobile and portable devices are designed to illustrate our methodology. Fast Fourier transform is chosen because it is a key digital signal processing (DSP) building block and is widely used in digital communication and signal processing [4] therefore improving power consumption in FFT blocks would add a lot of impact to overall system's power. In another aspect, hand gesture recognition processor is a real-time application-oriented design which targets for wearable and portable devices such as mobile phones and smartwatches which have very stringent power requirements. In both designs, novel system architecture and algorithms are proposed to obtain better energy efficiency as well as implementing suitable ultra-low power circuit techniques to achieve optimum energy point.

1.2 Objectives

The objective of this thesis is to design ultra-low power, high-performance fast Fourier transform and hand gesture recognition processor which are able to operate at high frequency while consuming significantly small amount of energy in compared with existing state-of-art designs. Besides using circuit techniques for subthreshold, ultra-low power design, novel system architecture and processing algorithms are also proposed to provide the most suitable platform for energy efficient circuit blocks. In the scope of ultra-low power objectives, the specific requirements are:

- (i) To design a high throughput up to Gsample/s Fast Fourier Transform processing core which has optimum energy point in subthreshold region (200 mV – 500 mV supply voltage) and consumes energy less than the most recent high throughput design in [4].
- (ii) To design a smartwatch application-oriented hand gesture recognition processor which is able to detect and interpret basic hand gestures in real time. The power consumption of the design should be in micro-watts range to meet very stringent power requirement of smartwatches. It is also to note that the focus of this design is on the digital processing part while analog front-end which includes of the sensor array, analog-to-digital converters are designed by another Ph.D. student from the University of Michigan in a collaboration project.

1.3 Contributions of the Thesis

This thesis contributes to ultra-low power design methodology for SOCs, in which two novel processors are designed and implemented:

- (i) A Fast Fourier Transform Processor: This is the first FFT processor implemented at silicon level using a feed-forward architecture which helps to increase the final throughput. Radix-2² algorithm with the new input scheduling scheme is presented to reduce circuit complexity and eliminating switching power from traditional delay lines for better energy efficiency and lower hardware cost. This work was published in [5] (TCAS-I'18).

- (ii) A Hand Gesture Recognition Processor: The proposed digital signal processor (DSP) consists of hardware compatible detecting algorithms for hand sweeping and zooming actions which are featured in energy saving mode for real-time applications. With infrared sensing technology, the DSP is able to recognize basic hand gestures for speedy actions without demanding for high-resolution images. The final design is the first SoC which uses passive sensing elements to minimize power consumptions for integrating into mobile and portable devices. This work was published in [6] (VLSI'16) and applied for a US patent.

In summary, system level architecture design and algorithms are fully utilized to provide better solutions for energy saving. A new FFT architecture and input scheduling algorithms are proposed to achieve higher energy performance in the FFT core while sensing technology and detecting algorithms are investigated and implemented to design the first ultra-low power hand gesture recognition processor for mobile phones and smartwatches.

1.4 Organization of the Thesis

The remaining of this thesis is organized into two main topics: The first topic is about Fast Fourier Transform processor which will be presented in Chapter 2 and Chapter 3. Fundamental FFT calculations and existing topologies will be discussed in Chapter 2 while our proposed FFT core will be described in Chapter 3. The second topic about hand gesture recognition processor will be shown in Chapter 4 and

Chapter 5. In Chapter 4, we review existing sensing methods, detecting algorithms and their advantages as well as disadvantages. Chapter 5 brings our proposed DSP implementation and real-time measurement results. Our conclusion and future works will be summarized in Chapter 6.

Chapter 2 Fast Fourier Transform

2.1 Discrete Fourier Transform (DFT)

Fourier transform (FT) is a technique to transform a signal from time domain into the frequency domain to analyze the signal's frequency components. In the frequency domain, the absolute value at each frequency represents its contributed amount to the original signal and the complex value is the phase offset of the sinusoid at that frequency. By decomposing a time domain signal into a summation of basic sinusoid signals, FT offers a great conversion tool for processing a signal which is difficult or even impossible to handle in the time domain, therefore, it has become one of the most important algorithms in digital signal processing. Figure 1 shows a decomposition using FT of a time domain signal into basic sinusoid frequencies.

Figure 1: Fourier Transformation of a time domain signal into a frequency domain signal [7]. A time domain signal is, therefore, a combination of different sinusoidal frequency signals.

Fourier transform of a discrete time signal is called Discrete Fourier Transform (DFT) which has been widely used and become a fundamental block in various applications including speech processing [8], image processing [9], encryption [10], etc. To compute an N -point DFT, it takes $O(N^2)$ complex arithmetic operations using below formula:

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot W_N^{kn}, k = 0, 1, \dots, N-1 \quad (2.1)$$

where $W_N^{kn} = e^{-j(\frac{2\pi}{N})nk}$ is called twiddle factors, $x(n)$ with $n = 0:N-1$ is a uniformly sampled sequence from a signal $x(t)$ in the time domain and $X[k]$, $k=0:N-1$ is a sequence of complex numbers representing the transformed signal in frequency domain. Equation 2.1 shows that a discrete time signal can be approximated by a summation of N -periodical sequences and with larger N , the approximation becomes better. However, calculation complexity is increased at the second order which means that DFT requires more hardware cost at a lower speed. Therefore, since 1960s many research works have been carried out to calculate DFT at a faster speed and contributed to fast Fourier transform algorithms (FFT).

2.2 Fast Fourier Transform (FFT) Algorithms

There are numbers of FFT algorithms to be named such as Cooley-Tukey [11], Prime-factor [12], Bruun [13], and Hexagonal [14] FFT algorithms. Cooley-Tukey by far is the most commonly used FFT algorithm which recursively breaks a discrete time series into smaller sequences to reduce the number of operations needed for FT calculations at each stage. Depending on the number of decomposed sequences, different algorithms are generated. Each algorithm has its own benefits and drawbacks which will be discussed subsequently. There are two ways to calculate FFT, discrete-in-time (DIT) and discrete-in-frequency (DIF). Both methods have the same time complexity and similar operations hence we only discuss DIF FFT in this thesis.

2.2.1 Radix-2 Algorithm

Discrete sequence $X[k]$ in Equation (2.1) can be decomposed into 2 sequences as shown in Equation (2.2) below:

$$X(k) = \sum_0^{N/2-1} \left[x(n) + (-1)^k x\left(n + \frac{N}{2}\right) \right] \cdot W_N^{kn} \quad (2.2)$$

where $k = 0, 1, \dots, N-1$. Now let $k = 2r$ with $r = 0, 1, \dots, N/2-1$, we have:

$$X(2r) = \sum_0^{N/2-1} \left[x(n) + x\left(n + \frac{N}{2}\right) \right] \cdot W_{N/2}^{nr} \quad (2.3)$$

$X[2r]$ is now the first $N/2$ point FFT of $\left[x(n) + x\left(n + \frac{N}{2}\right) \right]$. Similarly, let $k = 2r+1$ with $r = 0, 1, \dots, N/2-1$, we have $X[2r+1]$ is the second $(N/2-1)$ point FFT of $\left[x(n) - x\left(n + \frac{N}{2}\right) \right] W_N^n$ as shown in the below equation:

$$X(2r+1) = \sum_0^{N/2-1} \left[x(n) - x\left(n + \frac{N}{2}\right) \right] \cdot W_N^n \cdot W_{N/2}^{nr} \quad (2.4)$$

Instead of using $O(N^2)$ operations to calculate $X[k]$ by equation (2.1), only $O(2 \times \log_2 N)$ operations are needed by iterating the conquer and divide procedure in equation (2.3) and (2.4). Figure 2 shows the signal flow graph for an 8-point FFT. From equations and the signal flow graph, it is quite clear that to implement this algorithm $2 \times \log_2 N$ adders and $(\log_2 N - 1)$ multipliers are required. In compared with direct implementation of FT in equation (2.1), using Radix-2 algorithm not only the calculation speed is improved but also a significant amount of hardware cost is saved when smaller FT sequences are calculated then recursively run the algorithm to

achieve bigger sequences. This divide-and-conquer concept can be extended to higher radix for even lesser hardware and faster calculating speed.

Figure 2: Radix-2 signal flow graph of an 8-point FFT [15] using divide-and-conquer technique for faster FFT calculations.

2.2.2 Radix-4 and Higher Radix Algorithms

With the same approach, equation (2.1) can be written as a summation of four sequences as below to further enhance calculating speed:

$$X(k) = \sum_0^{N/4-1} \left[x(n) + (-j)^k x\left(n + \frac{N}{4}\right) + (-1)^k x\left(n + \frac{N}{2}\right) + (-j)^k x\left(n + \frac{3N}{4}\right) \right] \cdot W_N^{kn} \quad (2.5)$$

where $k = 0, 1, \dots, N-1$. Let $k = 4m, 4m + 1, 4m + 2, 4m + 3$ with $m = 0, 1, \dots, N/4 - 1$, we have:

$$X(4m) = \sum_0^{N/4-1} \left[x(n) + x\left(n + \frac{N}{4}\right) + x\left(n + \frac{N}{2}\right) + x\left(n + \frac{3N}{4}\right) \right] \cdot W_{N/4}^{mn} \quad (2.6)$$

this is an $N/4$ -point FFT of $\left[x(n) + x\left(n + \frac{N}{4}\right) + x\left(n + \frac{N}{2}\right) + x\left(n + \frac{3N}{4}\right) \right]$

$$X(4m+1) = \sum_0^{N/4-1} \left[x(n) - j \cdot x\left(n + \frac{N}{4}\right) - x\left(n + \frac{N}{2}\right) + j \cdot x\left(n + \frac{3N}{4}\right) \right] \cdot W_N^n \cdot W_{N/4}^{mn} \quad (2.7)$$

this is an $N/4$ -point FFT of $\left[x(n) + x\left(n + \frac{N}{4}\right) + x\left(n + \frac{N}{2}\right) + x\left(n + \frac{3N}{4}\right) \right] \cdot W_N^n$

$$X(4m+2) = \sum_0^{N/4-1} \left[x(n) - x\left(n + \frac{N}{4}\right) + x\left(n + \frac{N}{2}\right) - x\left(n + \frac{3N}{4}\right) \right] \cdot W_N^{2n} \cdot W_{N/4}^{mn} \quad (2.8)$$

this is an N/4-point FFT of This is an N/4-point FFT of $\left[x(n) - x\left(n + \frac{N}{4}\right) + x\left(n + \frac{N}{2}\right) - x\left(n + \frac{3N}{4}\right) \right] \cdot W_N^{2n}$

$$X(4m+3) = \sum_0^{N/4-1} \left[x(n) + j \cdot x\left(n + \frac{N}{4}\right) - x\left(n + \frac{N}{2}\right) - j \cdot x\left(n + \frac{3N}{4}\right) \right] \cdot W_N^{3n} \cdot W_{N/4}^{mn} \quad (2.9)$$

this is an N/4-point FFT of This is an N/4-point FFT of $\left[x(n) + j \cdot x\left(n + \frac{N}{4}\right) - x\left(n + \frac{N}{2}\right) - j \cdot x\left(n + \frac{3N}{4}\right) \right] \cdot W_N^{3n}$. Each equation (2.5-2.9) can be represented using a signal flow graph as shown in Figure 3 which is called a butterfly and when combining four butterflies we have a signal flow graph for Radix-4 algorithm of a 16-point FFT as in Figure 4.

Figure 3: Butterfly for equation (2.5) from Radix-4 FFT algorithm [15]. Note that input data with a distance of N/2 are paired up.

Figure 4: A signal flow graph of a 16-point Radix-4 FFT. Note that FFT calculations are divided into four 4-Point FFTs.

From equations (2.5-2.9) and the above signal flow graph, it can be shown that Radix-4 requires $(\log_4 N - 1)$ iteration stages. In compared with Radix-2, Radix-4 algorithm reduces the number of calculating stages at the cost of butterfly complexity and control logic to decompose input signals. It is also possible to achieve even higher radix algorithms to further decrease the number of iteration stages which help to reduce processing time. However, complexity for control circuits and butterflies also increases and it makes higher radix less attractive for hardware implementations. Therefore, it brings a demand for an algorithm that has simple butterfly but inherits higher radix advantages and Radix-2^k is derived.

2.2.3 Radix-2^k Algorithm

Radix-2² algorithm was developed to inherit the simple control structure of Radix-2 but adopt operation and hardware saving techniques from Radix-4. This advantage makes Radix-2² well suited for VLSI implementation of low power FFT.

In [16], by considering the first two decompositions of Radix-2 together equation (2.1) can be written in another form as in equation (2.10):

$$X[k_1 + 2k_2 + 4k_3] = \sum_{n_3=0}^{\frac{N}{4}-1} [H(k_1, k_2, k_3)] \cdot W_N^{n_3 k_3} \quad (2.10)$$

where $k_1 = 0, 1$; $k_2 = 0, 1$ and $k_3 = 0, 1, \dots, N/4-1$. H-expression can be written as

$$H(k_1, k_2, k_3) = [A + (-j)^{(k_1+2k_2)} B] \cdot W_N^{n_3(k_1+2k_2)} \quad (2.11)$$

in which $A = x[n_3] + (-1)^{k_1} x\left[n_3 + \frac{N}{2}\right]$ and $B = x\left[n_3 + \frac{N}{4}\right] + (-1)^{k_1} x\left[n_3 + \frac{3N}{4}\right]$. A

and B are two conventional Radix-2 FFTs which make up the first stage (BF-I) in Radix-2² and $H(k_1, k_2, k_3)$ is the second butterfly (BF-II) which is also a Radix-2 FFT of A and B with an additional complex multiplier for the twiddle factor $W_N^{n_3(k_1+2k_2)}$. Using this decomposition, the first two stages of Radix-2 can be transferred into one stage of Radix-2² which consists of two cascaded Radix-2 butterflies (BF-I and BF-II). The flow graph of this algorithm is shown in Figure 5 which requires the same number of stages as Radix-4 while has the simplicity of Radix-2 butterfly with an addition of one trivial multiplier in butterfly BF-II.

Figure 5: A signal flow graph of a 16-point Radix- 2^2 FFT Algorithm. Note that butterflies in Radix- 2^2 have a similar architecture as in Radix-2 butterflies.

2.3 Fast Fourier Transform (FFT) Architecture

There are two well-known architectures to implement FFT algorithms. One is the memory based architecture which uses a single processor with one or two memory banks to ping-pong the data and the processing results between the memory and the processor [17]. The memory-based design has the benefit of low hardware cost and simple control circuit however it has a drawback of slow processing speed since the data needs to be stored in the memory and wait idly for its next calculating cycle due to only one processing circuit is used for calculations. With a longer FFT length, memory size can be quite large and FFT speed is also limited by the speed of accessing memory data. Another architecture is pipe-lining [18] which divides FFT calculations into pipe-lined stages, data is processed sequentially in each stage and rearranged at the final stage. By pipelining the process, throughput is increased, leakage is reduced, and total energy consumption is lower. As a result, pipe-lined

FFT architecture has gained much of interests for real-time low power and energy efficiency designs. In this thesis, we will focus only on this architecture.

2.3.1 Delay Commutator Architecture

The most classical and straightforward implementation for pipelined FFT is Radix-2 multiple delay commutator (MDC) [19] shown in Figure 6. The input sequence is broken into two parallel data paths with $N/2$ delay distance in the first stage and $N/2^n$ for the n -th stage. From equation (2.2), two butterfly inputs have a fixed distance at every stage and it is realized using delay lines which delay one of the inputs by a correct distance. Commutator circuits are used to shuffle the data entering the delay lines and butterfly circuits calculate the output of equation (2.2) for the next stages. The butterfly used in MDC is butterfly type II (BFII) which perform calculations in equation (2.11). Radix-2 MDC architecture has simple control circuit and butterfly structure however it has a low utilization rate of 50% for both multipliers and butterflies since it accepts only one input per cycle and waits until the full set of data are available for butterflies [16]. With low utilization rate, the circuit stays in the idle mode more often while still being powered up and this causes not only leakage power to increase but also hardware resources are not fully utilized. Therefore, an improved architecture with better utilization rate is needed and delay feedback path helps to resolve this problem.

Figure 6: Radix-2 Multiple Delay Commutator Architecture for a 16-point FFT. Note that MDC architecture uses the same butterfly type II as presented in Radix-2².

2.3.2 Delay Feedback Architecture

Instead of feeding the butterfly output to the delay line in the next stage, single delay feedback (SDF) [20] architecture keeps the output in the feedback register and directs it back to its butterfly input with a fixed delay hence a correct distance between its inputs can be achieved. With this architecture, a single data stream goes through every butterfly stage which has its output fed back after a fixed cycle of delay, therefore, it improves register and butterfly utilization by continuously using these modules, but multiplier utilization remains at 50% as shown in Figure 7. To improve multiplier utilization rate, more delay feedback lines are required as in Radix-4 SDF implementation shown in Figure 8. In this diagram, three out of four outputs are stored in the delay feedback register therefore 75% utilization rate for the multiplier is obtained.

Figure 7: Radix-2 Single delay Feedback Architecture for a 16-point FFT. In this architecture, butterflies' outputs are delayed to pair up with the input.

Figure 8: Radix-4 Single Delay Feedback Architecture for a 16-point FFT. Similar to Radix-2 DSF, Radix-4 SDF uses multiple delay line to delay butterflies' output.

In compared with MDC architecture, SDF offers better register utilization rate hence smaller memory size, however, there is only one data stream which means only one output is generated every cycle hence its throughput is at one sample per cycle. For higher throughput, more data stream are designed for multiple delay feedback (MDF) architecture [21-23] which is elaborated in Figure 9. However, hardware cost and complexity in control circuits are increased with additional parallel paths and this makes MDF architecture less attractive for hardware implementation. In summary, there are two best-known FFT pipelined architectures: delay feedback and delay commutator. Generally, delay commutator offers high throughput but low hardware utilization while delay feedback improves utilization rate at the cost of lower throughput. However, when the two architectures are combined with different Radix algorithms, better hardware utilization and higher throughput can be achieved at the same time and demonstrations for these topologies will be discussed in the following section.

Figure 9: An MDF FFT implementation. (a) The parallel concept of single delay feedback. (b) The actual implementation of a multiple delay feedback FFT.

2.4 Review and Comparison of existing FFT topologies

R2MDC: With advantages of simplicity as presented in the previous section, this topology is quite attractive for VLSI implementation. To implement N -point FFT, it requires $2 \times (\log_4 N - 1)$ multipliers, $2 \times \log_4 N$ adders and a memory size of $3N/2 - 2$ [24]. There are two output streams, therefore, this design is able to generate 2 sample/cycle throughput.

R4MDC: Throughput of FFT can be further increased by using higher radix with MDC as in the initial design of R4MDC [25]. In this design, $3 \times (\log_4 N - 1)$ multipliers, $8 \times \log_4 N$ adders and a memory size of $(5N/2 - 4)$ are used to obtain 4 sample/cycle throughput. However, multiplier utilization rate for this design is as low as 25% since it accepts only one input every cycle as explained in Figure 10. To overcome this problem a modified architecture is shown in Figure 11 [4] which alters the input scheduling to have four input data per cycle hence a full utilization rate is achieved. Besides, this modified design uses a smaller memory size of $7N/4 - 4$.

With 2 parallel lanes and optimum energy techniques, the proposed FFT core works in the sub-threshold region at 270 mV supply voltage and consumes significantly less energy than other up-to-date reported cores.

Figure 10: Initial Design of a Radix-4 MDC FFT core. Note that this topology suffers from low utilization rate for multipliers.

Figure 11: A modified R4MDC architecture for a 1024-point FFT core. This architecture helps to increase multipliers' utilization rate.

R2SDF: As explained in Figure 7, delay feedback has better hardware utilization due to storing data in the registers. With better efficiency in hardware usage, R2SDF [26] reduced memory size to $N - 1$ registers but using the same number of multipliers and adders as R2MDC.

R4SDF: This is a Radix-4 version of R2SDF to increase multiplier utilization rate to 75% [27]. Similar to R4MDC, R4SDF uses $3 \times (\log_4 N - 1)$ multipliers and $8 \times \log_4 N$ adders but with smaller memory size $N - 1$.

R2²SDF: Radix-2² SDF was first introduced in [16]. Inheriting simple butterfly architecture from R2SDF while preserving benefits of Radix-4, R2²SDF requires the least number of multipliers, memory size and has 75% utilization rate for multipliers.

As shown in Figure 12, butterfly type-I is a Radix-2 butterfly consisting of two adders and two subtractors; butterfly type-II is slightly different with one additional multiplier which is a trivial multiplier as in equation (2.11). This multiplier can be implemented using a simple control circuit shown in Figure 12b with one added AND-gate and a multiplexer. In term of hardware cost, $R2^2SDF$ requires $(\log_4 N - 1)$ multipliers, $4 \times \log_4 N$ adders and $N - 1$ memory cells. This reaches the minimum hardware requirements among all the topologies and $R2^2SDF$ has become one of the most favorable topologies for VLSI implementation [28, 29].

Figure 12: (a) Butterfly type-I. (b) Butterfly type-II. (c) $R2^2SDF$. Note that BF-II is similar to BF-I but with additional trivial multipliers.

Parallel Pipe-lining topology: In order to increase the throughput, more output streams are required, and this can be done by adding more parallel processing paths. Almost all the above FFT architectures can be restructured into parallel configures to multiply the throughput by the number of parallel lanes as presented in [20]. However, the benefit of parallelism diminishes with increasing in parallel levels because of raising complexity in control circuits and additional hardware to move

data between parallel paths. Figure 13 [30] shows optimum parallel level for different FFT lengths and FFT algorithms. Using power area product as a cost function, this study brought a good methodology to determine the level of parallelism for high throughput FFT and by adopting parallel technique many high-performance FFT cores have been reported with throughput up to few Gsample/s [22, 31].

Figure 13: Power, area tradeoff for varying level of parallelism [30]. Note that the horizontal axis shows the level of parallelism while the vertical axis shows the normalized product of power and area (PAP).

To have an overview of different FFT topologies, specifications of these designs are summarized in Table I. Note that for the throughput, it does not consider parallelism to evaluate the advantage of different architectures and algorithms since it is better to compare the true throughput without paralleling technique to improve the processing speed. From the table, it is quite clear that MDC architecture has better throughput, but more multipliers, adders and bigger memory size are required while SDF reduces memory size and Radix-2² algorithm helps to lower both multiplier and adder counts. Radix-2² SDF shows overall advantages in term of hardware cost but has lower throughput than MDC.

Table I: Hardware and throughput comparison for different existing FFT processor architectures.

Architecture	Multiplier Count	Adder Count	Memory Size	Throughput
R2MDC [32]	$2 \times (\log_4 N - 1)$	$2 \times \log_4 N$	$3N/2 - 4$	2
R4MDC [25]	$3 \times (\log_4 N - 1)$	$8 \times \log_4 N$	$5N/2 - 4$	4
R2SDF [26]	$2 \times (\log_4 N - 1)$	$2 \times \log_4 N$	$N - 1$	1
R4SDF [27]	$3 \times (\log_4 N - 1)$	$8 \times \log_4 N$	$N - 1$	1
R2 ² SDF [16]	$(\log_4 N - 1)$	$4 \times \log_4 N$	$N - 1$	1

2.5 Conclusions

In this chapter, Fourier transform technique and fast Fourier transform algorithms are introduced. Various FFT algorithms which include Radix-2, Radix-4 and Radix-2² are derived by using different decompositions. Radix-2 algorithm offers very simple butterfly calculation with only adders and subtractors. Radix-4 algorithm reduces the number of processing stages for better calculating speed however its butterfly is more complex with additional complex multipliers. Radix-2² is the best combination of Radix-2 and Radix-4 as it uses two stage butterfly in each stage. The first stage butterfly (BF-I) is a simple Radix-2 butterfly and the second stage butterfly (BF-II) has very similar architecture to BF-I with an added logic gate and a multiplexer to realize its trivial multiplier. Therefore, Radix-2² has both advantage of simple butterfly and reducing number of iteration stages. Following radix algorithms, pipelined FFT architectures are present. In this section, advantage and disadvantage of delay commutator circuits versus delay feedback circuits are discussed and we found that in general delay commutator architectures tradeoff hardware cost for faster speed hence higher throughput. When matching radix algorithms with architectures,

some better topologies are discovered. Among all, Radix-2² SDF achieves the lowest hardware cost however it has a drawback of low throughput when using without parallelism and this is a very attractive area for further improvements which shall be proposed in the next chapter.

Chapter 3 A 1024-point Low Power High-Performance Radix- 2^2 Feed-Forward FFT Processor

3.1 Motivations

As discussed in Chapter 2, $R2^2SDF$ offers very attractive Radix- 2^2 algorithm to reduce hardware requirements however it has only one data stream to generate one sample/cycle. For high throughput demands, either paralleling techniques are required or the processor has to work at a faster clock frequency which consumes more power. This drawback makes SDF architecture less attractive while considering for high throughput applications.

From reported architecture in Chapter 2, MDC appears to be a better choice since it has multiple output streams that can generate multiple sample/cycle but its Radix-2 and Radix-4 algorithm require a lot more hardware components for multipliers, adders, and memory. Therefore, it leads to a hypothesis of using MDC architecture with Radix- 2^2 . The expected topology would have both hardware saving characteristic from Radix- 2^2 algorithm and high throughput characteristic from MDC architecture hence it could be able to offer high throughput with minimum hardware requirements.

To our best knowledge, there is in fact only one research work [33] which explored in this direction using FPGA as an implementation platform. The reason for this unexplored topology has a simple explanation that $R2^2SDF$ is well suited for low power applications while MDC meets the expectations for high throughput demands, and with MDF architecture throughput of $R2^2SDF$ can be multiplied to have higher sample/cycle output. However, with more and more mobile and internet of things

applications come to the market, designs of basic signal processing blocks such as FFT have to meet higher requirements for ultra-low power and high-performance ICs. In this chapter, a new FFT topology named Radix- 2^2 Feedforward ($R2^2FF$) will be introduced to offer better specifications both in term of hardware and power consumption.

3.2 Proposed Feedforward Architecture

Using MDC structure shown in Figure 6 with Radix-2 butterfly being replaced by Radix- 2^2 butterfly, a conceptual design of $R2^2FF$ topology is presented in Figure 14. Instead of feeding back butterfly's outputs via delay registers, these outputs are rearranged in the commutator circuits to ensure having correct distance between two data streams. BF-I, BF-II are two simple butterflies from Radix- 2^2 and output of BF-II will be multiplied by a twiddle factor fetching from a ROM array as required in equation (2.11). For an N-point FFT, the first commutator in stage 1 must separate its two outputs by a distance of $N/2$ and the second commutator provides a distance of $N/4$ then so on for the n^{th} commutator the required distance is $N/2^n$. In order to make this arrangement, the simplest way is demonstrated in Figure 15 in which two delay register lines and two multipliers are used. The delay register has the length equal to the required distance so that the outputs can be separated correctly.

Figure 14: Conceptual design of Radix-22 Feedforward FFT topology. Note that commutator circuits are used instead of delay lines.

Figure 15: Traditional circuit block diagram for a commutator which provides a distance of $N/2$ at outputs.

We denote the first part of the upper input by $U1$, the second part by $U2$ and similarly for the lower input with $L1$ and $L2$. This simple commutator circuit works in two phases. In the first phase (phase 0), the upper multiplier selects its input from Port 0 and delayed it by $N/2$ cycles at the output and at the same time $L1$ fills up the lower delay line. After completing $N/2$ cycles in phase 0, phase 1 starts with new input selections from two multiplexers. The lower multiplexer switches to Port 1 and $U2$ starts coming out at the lower output while in the upper output stream $U1$ is available after the delay line. Therefore, $U1$ and $U2$ are paired up at the output during phase 1. After phase 1 finishes, multipliers switch again to Port 0 to have $L1$ and $L2$ at the output. Finally, the output streams data are correctly separated by $N/2$.

To realize the first commutator, N registers are needed for the two delay lines and for the second commutator, $N/2$ registers are used so on and so forth. Hence, there is a total of $3N/2 - 2$ registers are consumed after $\log_2 N$ stages. The advantage

of this implementation is its simplicity, however, with a long FFT length the number of registers is increased, and the biggest drawback is the switching energy dissipated every clock cycle to shift the data along the delay line. This is a very huge wasted energy since only one data is output every cycle but the whole $3N/2 - 2$ registers must change their content. In fact, energy from memory accounts for 52% of the total power consumption [4] and the majority are used for moving data inside the delay line. Therefore, eliminating the delay line shall bring down the total power consumption significantly.

To redesign the commutator, a memory structure is used to replace the delay line, a read/write control circuit (R/W Ctrl) and address generator are used for reading and writing data into the memory which is design so that read operations are performed in the first half of the clock while writing is done in the second half. In Figure 15, two $N/2$ -register lines are used, one is to store the data, and another is to delay the data. In our proposed commutator shown in Figure 16, instead of delaying the data, the data is written into the memory cell after the cell is read out. During the first $N/2$ cycles, U1 and L1 data are written into the memory and in the following $N/2$ cycles U1 is read from memory for the upper output while U2 goes directly to the lower output. While reading U1, L2 is replaced every memory cell which is read out and after $N/2$ cycles the memory is filled with L2, and the operation is repeated so that L1, L2 are generated at the two outputs accordingly and the memory is always fully utilized to ensure a continuous data flow. By using a memory for holding the data, two delay lines are avoided and $3N/2 - 2$ register cells are used to build the memory. Figure 17 shows the proposed feedforward architecture of an N-point FFT which has butterflies from Radix-2² algorithm and the proposed commutator to

replace conventional multiple delay commutators. With the proposed architecture, two outputs are obtained every cycle. The total memory size is smaller than the architecture in [4] and it has the simplest butterfly structure with only adders and subtractors.

In compared with the commutator proposed in [33] where an FPGA implementation for $R2^k\text{MDC}$ was performed using traditional input scheduling mechanism with shift registers to prove the advantages of the $R2^k\text{MDC}$ architecture over existing parallel FFT topologies, our commutator not only requires lesser registers but also eliminates the delay line and therefore wasted switching energy is removed. Although the memory is accessed every cycle and for a full clock cycle with reading and writing operations for each non-overlapping half cycle, memory-based architecture only needs to change or fetch the content of one cell at a time so the dynamic energy consumed is much smaller than changing all registers as with the delay line. The only additional requirement to implement this proposed architecture is to design an address generator which must generate correct addresses for accessing memory cells. The address generating algorithm or input scheduling algorithm must be simple enough to avoid adding hardware and energy overhead to the overall circuit. The discussion for this algorithm will be discussed in the following section before coming to the detail implementations of the whole FFT core.

Figure 16: Proposed commutator architecture for Radix- 2^2 Feedforward FFT. Note that memory base is used to replace the delay lines.

Figure 17: Proposed Radix- 2^2 Feedforward FFT architecture. Note that BFI is butterfly type I which is conventional butterfly and BFII is type II butterfly as explained in Figure 12.

3.3 Proposed Input Scheduling Algorithm

Data is rearranged inside the commutator to have the correct distance required by the butterfly and the algorithm to arrange the data is called input scheduling algorithm. There are numbers of methods to shuffle the data and the simplest way is to postpone and multiplex the data with a required distance as shown in Figure 18. In this figure, the two inputs are having a distance of 16 and it is required to generate two outputs with a distance of 4. The first step is to delay the lower input by 4 cycles then interchanging a block of 4 data between the upper and lower input. Finally, it requires another delay at the output to align the two data stream. This algorithm is implemented for the MDC architecture in Figure 15 and its drawback was explained in the previous discussion. Nevertheless, this approach is still one of the most popular algorithms for scheduling FFT inputs.

Figure 18: A conventional data shuffling approach [25]. In this example, inputs are coming in two parallel paths with a distance of 16 and it is required to have a distance of 4 for the two outputs.

To work with the proposed commutator, conventional data shuffling is not suitable for memory-based design since data is now obtained by its address instead of getting them at the output sequentially via register lines. Figure 19 shows our proposed input scheduling algorithm which has two operational phases. In phase I, during the first half of the clock data are read from both memories in interchanging manner which means that the 2nd output (Out-2) reads from the 1st memory while the 1st output (Out-1) reads from the 2nd memory, in the second half of the clock data from input stream-1 (In1) is written into the first memory when input stream-2 (In2) goes into the second memory. This procedure continues for n-clock cycles (n = 4 in this example) if a distance of n is required at the output.

After the n^{th} clock, phase II starts by reading data from the 1^{st} memory for Out-1, and Out-2 directly uses coming input from In-1 as its data. Reading operations are completed while the clock is high, during the lower half of the clock cycle, writing operations are performed. In this phase, only data from In-2 is written into the 1^{st} memory. Phase II also carries for other n -cycles then the whole operations are repeated. By interchanging between two phases, correct distance data are coming at the output. In comparison with the conventional input scheduler, the proposed algorithm only changes the content of one memory cell every clock cycle since it can

access every single cell by its address whereas the conventional algorithm changes the content of all FIFOs in the delay line every clock cycle and generates significant switching power, especially with long delay line in the beginning stages of the FFT. Therefore, our proposed algorithm is able to minimize accessing data energy which is a large contribution to the overall energy consumption.

		CLK	In1	In2	Mem1	Mem2	Out1	Out2
Phase I	Out1 reads from Mem2	0	0	16	0	16	~	~
	In2 writes to Mem2	1	1	17	1	17	~	~
	Out2 reads from Mem1	2	2	18	2	18	~	~
	In1 write to Mem1	3	3	19	3	19	~	~
Phase II		4	4	20	20	16	0	4
	Out1 reads from Mem1	5	5	21	21	17	1	5
	In2 write to Mem1	6	6	22	22	18	2	6
	Out2 reads from In1	7	7	23	23	19	3	7
Phase I		8	8	24	8	24	16	20
	Out1 reads from Mem2	9	9	25	9	25	17	21
	In2 writes to Mem2	10	10	26	10	26	18	22
	Out2 reads from Mem1	11	11	27	11	27	19	23
Phase II		12	12	28	28	24	8	12
	Out1 reads from Mem1	13	13	29	29	25	9	13
	In2 write to Mem1	14	14	30	30	26	10	14
	Out2 reads from In1	15	15	31	31	27	11	15
	...	.	.	.	.	.	.	.

Figure 19: Data flow in our proposed input scheduling algorithm. Note that there is no shifting data and data is accessed directly by its address.

3.4 Implementation Details

3.4.1 Architecture Optimization

With the proposed architecture shown in Figure 17 and a suitable input scheduling algorithm in Figure 19, our 1024-point Radix-2² FFT core is proposed with the block diagram in Figure 20. Our FFT core consists of nine processing stages, a memory array to store intermediate data and a ROM array to keep complex twiddle factors.

In this design, two 16-bit data streams are used as the inputs, the internal data width has 40 bits after truncations and to minimize truncation error, bit width is allowed to grow up to 40 bits from stage 1 to stage 4 and truncations are applied from stage 5 to the final stage. In each stage, there is a commutator to rearrange data using our input scheduling algorithm, a dedicated memory to store the processing data and a butterfly for performing radix algorithm. Butterfly type-I (BF-I) is used in odd stages and Butterfly type-II (BF-II) is used in even stages which also consist of complex multipliers to multiply twiddle factors from ROM array with BF-II's outputs.

Figure 20: Our proposed 1024-point 2-parallel data path Radix- 2^2 feedforward FFT processor block diagram.

With 2 data streams, this topology can obtain 2 sample/cycles and for higher throughput, more parallel data paths can be used as proposed in [30, 33]. However, there are additional hardware components to interchange data between parallel paths. As shown in Figure 21, there are almost double of memory registers and computational elements although some components can be shared to reduce the hardware cost. These extra circuitries add overhead energy consumption and benefits of parallelism diminish for higher levels.

Figure 21: Parallel-pipelined topologies which are shown in [33]. (a) 2 parallel data paths. (b) 4 parallel data paths.

Without loss of generality, let's assume using p -level of parallelism with the throughput is multiplied by p and the energy consumption is reduced by p since the paralleling design completes the task with only $1/p$ of the original processing time. We denote the original energy as P_f then the reduced energy as a result of paralleling would be: P_f/p . The overhead energy can be modeled as: $p \times k \times P_f$ where $k \times P_f$ ($k < 1$) represents the energy consumption by the duplicated circuitries between parallel parts. Hence, the overall energy consumption for p -level of parallelism topology P'_f can be derived as:

$$P'_f = \frac{P_f}{p} + p \times k \times P_f \quad (3.1)$$

Level of parallelism p is calculated so that $P'_f - P_f$ is minimized or in another way minimizing the second order function $F(p)$ in equation (3.2)

$$F(p) = k \times p^2 - p + 1 \quad (3.1)$$

At $p = \frac{1}{2k}$, $F(p)$ has the minimum value of $F_{\min} = 1 - \frac{1}{2k}$. Clearly, the lesser duplication among parallel parts, the smaller k is and $F_{\min}$ becomes more negative for higher p . Figure 22 shows the plot for $F_{\min}$ at different values of k . For 10% of duplication among parallel circuitries, optimum number for parallelism is 5 and for 20% duplication, it is at 2. With $k \geq 50\%$, there are no more benefits of parallelism since $F_{\min} \geq 0$. As observed in Figure 21, there are many duplicated circuit components, therefore, coefficient k should be large and $p = 1$ is selected due to this observation.

Figure 22: Benefit of parallelism at different percentage of common parts. In this figure, comparison for circuits with different percentage of the common parts is performed and it is clear that with increasing in common parts, parallelism provides much lesser benefit.

To make an estimated comparison with SDF topology in Figure 12, we used the same approach with equation (3.1). For our proposed topology, it requires $(3N/2 - 2)$ memory registers while SDF only requires $(N - 1)$ registers. Comparing two

architectures, we found that registers are doubled in our feedforward architecture for all stages which have two parallel data streams and they are the same for the serial-to-parallel stage, therefore, the overhead energy can be modeled as: $k \times P_f$ where k is the percentage of the total energy consumed by memory registers and P_f is the overall energy. Since two architectures have the same butterfly so they should consume the same amount of energy in both topologies. We have the energy difference is:

$$P_{proposed} - P_{sdf} = \frac{P_f}{2} + k \times P_f - P_f \quad (3.2)$$

In many designs [4, 34], energy consumed by memory parts is less than 50% of the total energy, therefore our proposed architecture has smaller energy than the SDF architecture although using larger memory size. Equation (3.2) also suggests that with more efficient memory design, the proposed Radix-2² feedforward architecture offers even better energy consumption as compared with Radix-2² SDF topology.

3.4.2 Butterfly Design

As mentioned in the previous section, Radix-2² butterfly has a very simple structure with least requirements of hardware. There are two types of Radix2² butterfly: BF-I is the same as Radix-2 butterfly which is implemented by using two adders and two subtractors shown in Figure 23(a) while BF-II is a trivial butterfly with a complex multiplier. For BF-II, the H-expression in equation (2.11) is broken down into two phases which have outputs depicted in Figure 23(b). With this implementation, BF-II requires an additional control signal and four multiplexers for switching data between two phases. The control signal is generated from the most significant bit (MSB) of an n -bit counter where n is $(1 + \log_2 d)$ with d is the distance required between two BF's inputs. For example, the input distance in the 2nd stage is

256 then a 9-bit counter is used for generating the control signal. During operations, phase 1 (switches are at 0) remains for 128 clock cycles then the counter toggles its MSB to switch to phase 2 mode (switches are at 1) for another 128 clock cycles. This counter is also utilized as an address generator to fetch twiddle factors from a pre-programmed ROM memory. The address location in the twiddle memory is set to be the same as the output of the counter so that in each cycle it gets the correct twiddle value to the multiplier. In this design, we employ Baugh-Wooley multiplier [35] which is famous for its regularity in array multiplication and high speed operation to multiply BF-II's output with twiddle factors.

Figure 23: Butterfly implementation. (a) BF-I. (b) BF-II. Note that BF-I is a trivial Radix-2 butterfly while BF-II requires complex multipliers and modifications in the data path.

3.4.3 Commutator and Memory Design

Commutator circuits use our proposed input scheduling algorithm to rearrange the data with correct distances required by the butterfly. Hardware implementation of our input scheduler is shown in Figure 24. As observed, data is read and written by blocks of n -data sequentially in each phase, therefore their addresses in the memory can be arranged in sequential orders. This pattern simplifies

the way to access the memory by using a counter to generate the address of memory cells. With output distance of n , a $(1 + \log_2 n)$ bit counter is required and the MSB is the indicator for phase order (phase-I when MSB = 0 and phase-II when MSB = 1) while the other bits are the data address.

There are two MUXs (one at the input and another at the output) which help to switch data as required by the algorithm and the states of these MUXs are determined by the MSB signal. R/W circuit is just a buffer circuit to generate non-overlapping read (RD) and write (WR) signals from the clock and enable signal. Note that the whole memory interfacing circuit including address generator, decoder, and R/W control are inherited from a conventional SRAM design, therefore, it can be used directly with SRAM structure for lower frequency with lower power budget applications. In this design, latch based memories in Figure 25 are used for high-performance requirements. Using registers, accessing data speed can be as high as few hundred megahertz while with SRAM the maximum speed only reaches 30 MHz at the optimum supply voltage. Another trade-off between register-based memory and SRAM is the layout area, in which SRAM is very famous for its transistor density and has much smaller layout area than the register-based memory as demonstrated in Figure 26. For a double memory size SRAM, the layout area is still 1/10 of the area required by the register-based memory. With many advantages to offer, incorporating high-speed SRAMs would be the next design target for our future works.

Figure 27 shows a timing diagram illustration for the 5th stage of our FFT core. Initially, the two memories are empty, and it takes n -clock cycles to fill an n -data block into two memories whose size is equivalent to the required distance of n .

Once the memories are filled, RD (read) signal reads out a data during the first half of the clock and then WR (write) signal overwrites the data contained in this location with a new data taken from the input. After finishing the first data block, the next data block is already available therefore there is no interruption in the output data stream to ensure a continuous data flow inside the FFT. Two output samples will be generated every clock cycle hence a 2 sample/cycle throughput is obtained with this core.

Figure 24: Hardware block diagram of our proposed input scheduler. Note that the number of bits in the counter will determine the distance at the output.

Figure 25: Latch based memory design. Note that Read and Write operations can be performed in the same clock cycle.

Figure 26: Layout area comparison between register-based memory on the left and an SRAM on the right. The 16-bytes register-based memory requires an area of $160 \times 140 \mu\text{m}$ while it is $73 \times 30 \mu\text{m}$ for a double size SRAM (32-bytes).

Figure 27: Timing diagram in the 5th stage of our proposed FFT core. Note that it takes n -first clocks to fill data into the memory with size n and after that, a continuous flow is maintained.

3.4.4 Optimum Energy Point

To achieve lower energy consumption, supply voltage scaling is a very efficient approach since dynamic power is proportional to the square of voltage. However, when supply falls below the threshold voltage of transistors, the circuit is not totally cut off but operates under sub-threshold region and leakage power starts

dominating dynamic power. Furthermore, as technology scales down the transistor size leakage current becomes even more severe. In [36], it reveals that leakage was estimated to increase from 0.01% of overall power consumption in a 1.0 μm technology to 10% in a 0.1 μm technology.

The total energy consumption is a summation of dynamic and leakage energy. For a given threshold, when the supply voltage is dropped, the total energy is reduced since dynamic energy starts decreasing as a square function of the power supply. However, once the supply voltage goes into the subthreshold region, the propagation delay increases significantly resulting in a corresponding increase in leakage energy [34]. As a result, total energy which is dominated by subthreshold leakage is increased. This characteristic makes it feasible to find an optimum supply voltage to minimize total energy dissipation as shown in Figure 28.

Figure 28: Energy characterization curve as a function of supply voltage. Note that at the subthreshold region, dynamic energy is reduced with lowering supply voltage however leakage energy is increased exponentially, therefore, the total energy is increased at an exponential rate when the supply voltage is decreased in the subthreshold region.

At the V_{opt} , energy consumption is optimized however the circuit is operating in subthreshold or near subthreshold region and its performance degrades since there is

smaller pull-up or pull-down current causing weaker switching strength therefore besides optimizing for the energy, circuit delay also must be considered carefully in the sub-threshold region. A chain of 64 inverters is used to simulate with HSPICE in order to find a minimum energy point and an optimal circuit performance region that the design should be operated in [34, 37]. Different activity factors are also considered in this simulation. From Figure 29(a), operating region around $V_{DD} = 0.3$ V is the optimum energy region that the processor should be powered up. Furthermore, V_{th} for this process is about 370 mV below which leakage current will dominate dynamic current and degrade the circuit performance. After considering the energy and the delay curves in Figure 29, $V_{DD} = 0.4$ V is selected for optimum energy point and $V_{DD} = 0.6$ V is selected for optimum circuit performance. Above 0.6 V, circuit delay is reduced at a much slower rate and the benefit of trading off between higher voltage supplies for faster speed is diminished.

Figure 29: Optimum operating point of the devices. (a) Circuit energy consumption at different supply voltages. (b) Circuit delay at different supply voltages. Note that $V_{DD} = 0.4$ V is chosen as the operating voltage to achieve minimum energy point and $V_{DD} = 0.6$ V is chosen as the best circuit performance operating point.

3.5 Simulation and Measurement Results

3.5.1 Simulation Results

The proposed 1024-point 2-parallel data path feedforward radix-2² FFT core is first implemented with Matlab to verify the functionality of the overall topology. For verification purpose, a complex signal in Figure 30, $y(t) = 0.7 \times \sin(100\pi t) + j \times 0.3 \times \sin(300\pi t)$ is used. To have a better estimation of hardware circuit's performance, the fixed-point number is employed with truncations applied from the 5th FFT stage. The final Matlab output is plotted against the output of an embedded Matlab FFT function in Figure 31 for comparison purpose. The proposed FFT's spectrum is almost identical to the ideal FFT function from Matlab except having some noises at higher frequencies due to quantization error. Without truncations, the spectrum output is exactly the same. Upon functionality verification in Matlab, twiddle factors and input signals are digitalized into 16-bit data for Verilog simulations with ModelSim. In ModelSim, real numbers and imaginary numbers are considered as separated data, therefore, there are four inputs as well as four output signals in the simulation results in Figure 32. Numeric outputs of ModelSim is compared with Matlab digitalized output for every clock cycles to ensure they are having the same results.

Figure 30: A 1024-point discrete input signal used for testing: $y(t) = 0.7 \times \sin(100\pi t) + j \times 0.3 \times \sin(300\pi t)$.

Figure 31: FFT Signal Spectrum. (a) From an embedded Matlab function. (b) From our proposed FFT topology.

Figure 32: ModelSim simulation results for our proposed FFT core. Note that START signal is used to indicate the beginning of new FFT stage.

In Figure 32, every FFT stage has a flag signal which is named as START signal to indicate completion of the previous stage's calculations and the data is ready to flow into the next stage. As the waveform shows there is no empty gap in the output, our design can generate continuous output data and achieves full utilization of computational elements (CE). After ModelSim verifications, the 1024-point Radix² feedforward FFT core is synthesized in Cadence Virtuoso using low power library from STM 65 nm process. The processor occupies an area of 3.6 mm² (2.0×1.8 mm) using 128,000 cells. A maximum clock frequency of 800 MHz is used for synthesis without any timing violations. The chip layout design and the micrograph of a fabricated IC are shown in Figure 33.

Figure 33: (a) FFT chip layout design in Cadence Virtuoso. (b) Micrograph of the proposed FFT processor.

3.5.2 Measurement Results

Test chips are measured across different voltage supplies from 0.3 V to 1.0 V and clock frequencies from 100 MHz to 600 MHz to evaluate both energy and throughput performance. The same input in Figure 30 is programmed into pattern generator of a Keysight 16860 Logic Analyzer. The output of the logic analyzer is converted to 0.4 V, 0.6 V, and 1 V through resistive dividers for testing the chips at different voltage levels. Finally, the truncated outputs of the FFT processor are collected and compared with the output from Matlab for accuracy assessment.

Due to bit length truncation and finite bit representation in FFT, it causes quantization error and impacts the accuracy which is quantified by Signal-to-Quantization Noise Ratio (SQNR). Figure 34(a) shows the FFT spectrum of the output signal generated from our test chips. Since in each stage the output's bit width increases and it is not possible to accommodate all the length, truncations are required. From simulations in Figure 34(b), using uniform bit-width from the first

stage reduces the accuracy as much as 30 dB from the theoretical value [38], therefore in initial stages of our design data word-width are allowed to grow up until it reaches 40 bits at the 4th stage. Truncations are only applied from stage 5 till the final stage to have 32-bit data path. By employing this strategy, FFT spectrum output of our FFT achieves -1 dB difference from the ideal value. Actual measurements of our FFT core's output show two dominant frequencies at 50 Hz and 150 Hz together with some noisy frequency components in which three pulses are noticeable at 100 Hz, 200 Hz and 450 Hz. These components contribute to quantization noise and reduce our SQNR by 1 dB. This result shows that our proposed FFT core achieves a very high level of accuracy compared with uniform data width designs.

Figure 34: FFT output spectrum. (a) Our proposed FFT core with non-uniform bit width. (b) Simulated FFT core with uniform bit width. Note that the output spectrum of uniform bit width design is much noisier than non-uniform bit width design.

To measure energy consumption, the FFT is supplied at different voltages from 0.4 V to 1 V while sweeping the clock frequency from 50 MHz to 600 MHz. Figure 35 summarizes the measured FFT energy consumption and operating frequency versus voltage supply. At the target operating voltage of 0.4 V and 50 MHz, the FFT consumes 5.8 mW and takes $1024/(10^8)$ seconds to complete one FFT calculation

with 100 Msamples/s, therefore, energy per FFT is 59.3 nJ/FFT. At 0.6 V with a maximum clock frequency of 400 MHz, the proposed FFT consumes 60.3 mW or energy per FFT of 77.2 nJ/FFT for 800 Msample/s throughput. Using normalized energy conversion formula [17] in which normalized energy is directly proportional to the execution time and power consumption but inversely proportional to the data path width and technology node, different FFT designs are comparable across FFT length, word width and technology node.

Compared to the most recent ULV design [4] which requires 84.0 nJ normalized energy for high performance of 0.6 V supply and 290 MHz clock frequency, our proposed processor has better energy efficiency for mid-high range performance while it has slightly higher energy consumption in ultra-low voltage domain (below 0.4 V). The maximum throughput of 1.2 Gsamples/s is achieved at 1 V and 600 MHz clock frequency. In this mode, the proposed processor demands for 213.3 nJ/FFT.

In term of layout area, our design has the best figure of merit. Our processor only occupies 3.6 mm² compared with 8.5 mm² in [4] or 5.1 mm² in [31]. Table II summarizes comparisons for our proposed FFT core with other state-of-art designs. To obtain normalized energy so that designs for different FFT length, word-length and technology nodes are comparable, the equation in [17] is used.

Figure 35: Measurement results. (a) Power consumption. (b) Maximum clock speed performance.

Table II: FFT cores comparison between our proposed design and existing high throughput designs.

Specifications	This work	VLSI '10 [38]	TCAS '10 [21]	JSSC '12 [4]
FFT length (N)	1024	1024	256	1024
Technology	65 nm	65 nm	90 nm	65 nm
Word-length	32 bit	16 bit	10 bit	16 bit
Topology	R2 ² FF	Mix Radix MDF	R2 ⁴ MDF	Modified R4 MDC
Supply Voltage	0.4~1.0 V	0.5~1.0 V	0.625~1.0 V	0.27~1.0 V
Chip Area	2.0×1.8 mm ²	2.39×1.40 mm ²	2.26×2.26 mm ²	2.71×3.15 mm ²
# of Adders	4×log ₄ N	4×log ₄ N	4×log ₄ N	8×log ₄ N
# of Multipliers	log ₄ N - 1	log ₄ N - 1	log ₄ N - 1	3×(log ₄ N - 1)
Memory size	3N/2 - 2	N - 1	5N/2 - 8	7N/2 - 4
Power Consumption	60.3 mW @ 0.6 V	13.83 mW @ 0.45 V	16.4 mW @ 0.625 V	82 mW @ 0.6 V
Max Clock Frequency	600 MHz	20 MHz	500 MHz	290 MHz
Normalize Energy for high throughput	77.2 nJ @ 0.6 V, 800 Msample/s	89.5 nJ @ 0.43 V, 80 Msample/s	167.9 nJ @ 0.625 V, 2.4 Gsample/s	84.0 nJ @ 0.6 V, 2.4 Gsample/s

3.6 Conclusions

In this chapter, a 1024-point Radix-2² feedforward FFT is introduced. The proposed design utilizes advantages of simple butterflies with fewer hardware requirements from Radix-2² algorithm and high throughput characteristic of feedforward architecture. A new feedforward architecture is proposed with delay

lines being replaced by register-based memories which eliminate energy consumed when shifting data along the delay lines. Instead of shifting all data contents sequentially inside delay lines, required data is obtained directly by using its address. Read and write operations are controlled by a special input scheduling algorithm to generate correct memory addresses for accessing the memory.

Our proposed design is the first FFT core using Radix-2² algorithm with feedforward architecture implemented at wafer level to prove its advantages over existing topologies. The architecture and algorithm design are verified with Matlab, ModelSim then implemented using STM 65 nm. The overall design requires an area of 2.0×1.8 mm² which is the smallest among other designs for similar FFT size and data word-length. This achievement is a result of using Radix-2² algorithm with our proposed commutators for hardware savings.

Fabricated chips are measured, and spectrum results are verified with Matlab to ensure a high level of accuracy for our FFT which uses non-uniform bit-width for lowering quantization noise. Optimization methods are also employed in hardware designing process to analyze different parallel-pipelining configurations for finding the optimal settings. Besides, circuit optimizations are carefully studied to locate the minimum energy point and maximum performance region. Measurement results confirmed these benefits. Test chips show reliable performances for high throughput demands with 800 Msample/s while consuming 77.2 nJ/FFT at a supply voltage of 0.6 V. For low power domain, the proposed FFT core uses 5.8 mW or 29.6 nJ/FFT to generate 100 Msample/s at its optimum energy point $V_{DD} = 0.4$ V.

In summary, a 1024-point Radix2² Feedforward FFT is successfully designed and implemented. Measurements of test chips show advantages of the proposed topology over existing high throughput designs and offer promising results to draw more research interests in this direction.

Chapter 4 Hand Gesture Sensing Techniques and Available Detecting Algorithms

4.1 Motivations

Interactions between human and machine are becoming more and more popular with fast evolution in technology. Smart devices are invented to interpret human's instructions through face or hand gestures to bring more conveniences for users. Smartphones with air-gesture features for page flipping, screen rolling have become available. Gesture detecting technology is also widely used in the gaming industry in which game character's movement can be controlled directly by the player's gesture instead of using gamepads. This makes the games more interactive and the player feels being a part of the game. Another very useful application of hand gesture detection is in food and pharmacy manufacturing where hygiene is utmost important. In these factories, many touch screens have been replaced by hand gesture control devices to avoid direct contacts which may introduce contamination to the final products. Recently, with developments in virtual reality hand gesture detection has drawn much of research interest as a core functionality to facilitate real feelings in a virtual world. Besides, hand gesture recognition is also applied in sign language to detect Portuguese, Japanese [39, 40] and open opportunities for human assisting devices for auto-typing, language processing or human-machine communication. As a result of its wide applications, hand gesture detection plays an important part in many today systems however, there are very few reported standalone hand gesture recognition processors since most of the applications use software for processing. With software, many complicated gestures can be processed but it requires having a

platform which includes processors, memories to support and it demands high power consumption too.

For mobile applications, battery working time is limited therefore software processing is not an optimum option. Besides, current developments of smartwatches show a great potential for hand gesture detection. With a very small screen, touch control seems difficult and less interactive actions can be done, as a result, controlling smartwatches using hand gesture is a great option. A hand gesture detection processor is, therefore, a good solution to energy consumption and device's size problem since we can optimize system architecture and detecting algorithms for power and as a standalone SoC, it can be easily integrated inside mobile devices or smartwatches.

In two chapters (Chapter 4 and Chapter 5), hand gesture detecting technologies will be introduced and a completed SoC design will be presented to demonstrate our proposed detecting algorithms as well as its ability to recognize different gestures at ultra-low power consumption which is suitable for mobile and smartwatch applications.

4.2 Sensing techniques for hand gesture recognition

The first step for gesture recognition is gesture sensing in which images of gestures are captured and stored for further processing. There are many ways to capture a gesture such as using image sensors, ultrasound sensors, RF sensors or infrared sensors.

4.2.1 Image sensing

High-resolution images are captured then the software will analyze these pictures to get information about the gesture. The most popular device used to capture image is Kinect which was developed by Microsoft [41]. It includes an infrared camera to measure distance and reconstruct the 3D structure of an object together with a VGA (640×480) video camera to produce 30 frames per second image stream. Kinect device can perform hand and body tracking. At relatively low cost, Kinect has become very attractive tools for gesture detection in vision-based. In [39] and [42], hand gesture recognition system using Kinect sensor was proposed to interpret 3D hand movement and language signs for Portuguese and Japanese respectively. In addition, image sensor brings a great advantage over other techniques when it provides color information of the object. With YCbCr or RGB color space, difference objects such as hand, face, or human body can be differentiated to make decisions about gestures more accurately and have a wide range of complex detectable gestures which are highly desired by gaming applications [40]. As for drawbacks, the image sensor is light dependent, causing it to fail to work in absence of light or not stable under low light environment. Furthermore, image sensors require large memory size and consume a lot of power, therefore, image sensors are more suitable for applications which have powerful processors with high power supplies.

4.2.2 RF Sensing

By transmitting an RF signal which travels toward the object and reflects to the receiver, object's information can be obtained by deriving the distance from the time-of-fly (TOF) of each pulse. Since RF signal works independently without light,

the sensor is able to perform under no light condition. This technique is best used for movement detection where object's appearance or object's absence is interpreted by interruption of RF signal. For gesture recognition, low resolution is the biggest obstacle to reconstruct the object's shape, however, attempt to use RF sensor for gesture sensing has increased recently. With the aid of accelerometer, a system consisting of RF sensors and accelerometers [43] is able to detect 3D hand movements which are programmed to control lights. In [44], a group of MIT researchers proposed a design of wearable hand-held interface using passive RFID sensor tags. The design was successfully demonstrated with sensors attached to user's hand to control a computer. The advantage of this design is using passive RF tags instead of active sensors which continuously generate RF pulses to reduce power consumption. However, it is not convenient when the user must wear the device for interaction. Another significant progress in RF sensing is the work reported in [45], a single scalable RF transmitter/receiver cell that can be constructed into a large array for 3D imaging. The cell consumes only 1.4 W and its working range is from 20 cm to 30 cm. This achievement brings the possibility to design a standalone RF gesture detection system in near future.

4.2.3 Ultrasound sensing

Time-Of-Fly (TOF) principle is also applied for ultrasound signal for sensing as shown in Figure 36. In this figure, a circuit for rangefinder is implemented using an ultrasound sensor which consists of a transmitter and a receiver. The system is able to calculate the distance from the object to the sensor by measuring the time that an ultrasound pulse travels starting when it hits and reflects from the object until it is

received at the receiver. By using an array of ultrasound rangefinder sensors, object's shape and its movement can be detected.

Figure 36: A Rangefinder using Ultrasound Sensors [46]. The working principle of this rangefinder is using TOF calculations to determine the distance.

In comparison with RF sensing, ultrasound is less vulnerable to noise and also requires smaller area which is very important to build a large sensor array, as a result, ultrasound sensor is more widely used. The most popular application of ultrasound is medical ultrasonic imaging to capture images inside the human body for diagnostics and diseases detection. Ultrasound is also used for motion sensing, surface imaging, and recently gesture sensing. A system which consists of ultrasound emitters, sensor were constructed in [47] to perform 3D interface with a computer. Emitters are set at the corners and when the sensor is moved, the distance from the sensor to emitters will be measured by signal's time-of-fly. From these distance, the coordinates of the sensor will be calculated to track the movement. This system is able to imitate a computer wireless mouse accurately. To make a more compact system, ultrasound transmitter and receiver were integrated into a single SoC rangefinder [48] which occupies an area of 1.67 mm^2 in $0.8 \text{ }\mu\text{m}$ CMOS technology. However, transmitter

and receiver were then still mounted on top of the IC and only when micromachining techniques were developed, receiver and transmitter were able to be fabricated in few micrometer-thick PZT films, exhibiting piezoelectric properties [49]. The ultrasound range finder has now become a completely integrated system which has processing circuits, transmitter and receiver all sit on a single IC. Using micromachining, advance ultrasound gesture recognition systems were presented in [46] and [50]. These designs have long detection range, high accuracy and even suitable for gaming applications. The power consumption of these systems is however still in sub-mW range.

4.2.4 Infrared (IR) sensing

For ultra-low power design and portable application, active sensing elements used in above designs are not able to meet power requirements. With active elements, transmitter always consumes a great amount of power as it continues emitting signal even in absence of targeted objects. To save power from the transmitter, the human hand can be considered as a heat source which generates infrared (IR) signal and a passive infrared (PIR) sensor should be used to capture the thermograph for analysis. An array of 4×4 PIR sensors was conducted in [51] to confirm the feasibility of using PIR for hand gesture recognition. In this work, a completed system including 16 PIR Pyreos sensors which requires 500 times less power than active elements [52] and a signal processor was implemented. The final design is able to detect hand movement in 4 directions at 20 cm distance: Up, Down, Left, Right. The sensor array occupies an area of 1.5×1.5 mm and consumes 200 μ W in full working mode. Another development in PIR sensing was reported in [53] which is the first large PIR sensor array that can capture a clear thermal image shown in Figure 37. The sensor was

implemented with 4 built-in amplifiers using Molybdenum-gate CMOS with MEMS process. By successfully capturing good resolution thermograph of the object, this work shows possibilities for using IR sensors to detect complicated gestures.

Figure 37: A thermograph of hand gesture captured from a 16×16 PIR array with 4 built-in amplifiers using Molybdenum-gate CMOS with MEMS process [53].

4.3 Hand Gesture Detecting Algorithms

4.3.1 Time-of-fly Algorithm

Different detecting algorithms are designed to fit different sensing technology and the most popular detecting algorithm is time-of-fly (TOF) which is used for ultrasound sensing and RF sensing. In TOF, a transmitter transmits an ultrasound or an RF signal toward the object then measures the time taken for signals reflecting from the object to derive the distance between the transmitter and the object. By using multiple transmitting sources, the exact location of an object can be detected hence any movement can be tracked easily. TOF algorithm is used in many rangefinders [46, 48, 50] to provide high accuracy, fast response and long working range, however, generating ultrasound or RF pulse requires higher voltage supply which is not desirable for mobile and portable devices. In addition, this algorithm

also has another drawback when transmitters keep emitting pulses for object detection and the processor has to be in active mode all the time for movement tracking, therefore, the total power consumption is increased. Even in absence of any object, transmitters also keep sending signals which generate wasted energy and reduce the battery time. As a result, TOF is only found in high-performance systems which are supplied with high voltage and energy consumption is not a strict constraint.

4.3.2 Template Matching Algorithm

To detect a gesture, predefined templates can be used to match with the captured frame. In order to perform template matching, a series of detectable gestures are recorded from cameras at different views to generate a temporal template library of each gesture. When a new gesture is captured, its frame will be compared with the existing template using Hu moment [54] and statistical models of mean and covariance matrix. Mahalanobis distance [55] is then computed for the gesture to each template and the closest distance will be selected as the detected gesture. Figure 38 shows an example of using template matching for a standing up gesture. The distance of this gesture is calculated against six templates and results show that template No. 6 which is a template for standup up gesture has the closest distance to the recorded gesture, therefore, it is concluded as a standing up gesture. Template matching offers a great way to recognize a predefined set of gestures with good accuracy and fast response which is very suitable for real-time applications. However, it needs to build up a comprehensive template library that needs a big memory to store the images. Besides, the resolution is also a constraint to utilize this algorithm. As the gesture is matched against each template, the resolution of each

template should be good enough to avoid any misclassification and certainly, it demands using high-resolution image capturing devices which will make the circuit more complex and less attractive to small device size applications.

Figure 38: Gesture detection using template matching. (a) Standing up gesture. (b) Six templates to compare with the gesture [56].

4.3.3 Neural Network and Pattern Matching

Using artificial intelligence to recognize the gesture is a promising direction that has drawn research interest recently. In this algorithm [57], the hand image is first gone through an edge detection filter to extract edge filtered image, it is then tokenized to obtain neuronal network usable image form. On the training data, the algorithm uses supervised learning with backpropagation neural network to train the algorithm to detect gestures for numbers from 1 to 5. Upon verification, the network is able to recognize these gestures at accuracy rate as high as 97.36% for number 5 and the lowest detection rate of 66.66% for number 3. These measurements are performed by using an Android device to demonstrate algorithm's ability to work

well with mobile and smart devices. Although having good detection rate and real-time ability, the algorithm still has some limitations in hardware implementation since neural network requires a large amount of memory for training data and classification algorithm consumes high power especially when the number of predefined gestures increases. As a result, the neural network is more suitable for software applications and it still requires more simplifications to have future hardware implementations.

Chapter 5 A 256 pixel, 66.8 μ W Infrared Gesture Recognition Processor for Smart Devices

5.1 Proposed System Architecture for Hand Gesture Recognition SoC

With considerations for ultra-low power applications, IR sensing seems to be the best candidate to implement our sensor array. A hand emits infrared radiation with wavelength representing its temperature and form a thermal image in front of the sensors. Comparing with other sensing technologies, IR sensing is a passive sensing technique which does not require a transmitter to generate sensing signals but using object's radiation directly, therefore, power consumption is significantly less with IR sensors. Especially, in idle mode without object's presence, there is no emitting signal from the sensor, the system only consumes standby power which is much lesser than continuously emitting signals for object detection. When an object appears, its thermal radiation is used as a wake-up signal to trigger the whole detecting procedure. Figure 39 shows our proposed system's block diagram of the hand gesture recognition SoC. At the analog front-end (AFE) an array of IR sensors is used to capture the thermal graph of an object by converting heat to voltage level. Each thermopile signal connects to an analog front-end path that consists of an instrument chopper amplifier and a low pass filter (LPF) for filtering out-band noise [6]. The output of the LPF is digitized using time-division multiplexing ADCs which are used for all the row output signal from the sensor array. Digital signals are then processed and analyzed in the digital signal processor (DSP) to determine the gestures. The DSP is designed based on motion history image (MHI) method which

uses multiple time series frames of an object's movements to trace the object and make conclusions about its gesture. In this design, 8 motion directions which include: up, down, left, right, 4 diagonal directions and zooming in, zooming out are detectable besides a special gesture is designed to resume the system from standby mode by forming a wake-up gesture.

Figure 39: Proposed Hand Gesture Recognition SoC which consists of analog front-end to obtain input data from IR sensor and a digital signal processor to analyze input gestures.

The AFE part is designed and implemented by a Ph.D. student from the University of Michigan using a 16×4 thermopile array and our DSP is designed with two versions. The first DSP version is workable with a small sensor array 16×4 elements, therefore, it has a limited number of features in which only moving in 8 directions and a wake-up gesture are implemented. In our second DSP version, to improve sensing capability, we utilize an HTPA 16×16 infrared array sensor from Heimann Sensor GMBH [58] to capture infrared images then our detecting algorithms are used to analyze these gestures. The images from the Heimann sensor arrays has 16×16 pixels as shown in Figure 40. With better resolution images, we are able to demonstrate capabilities of our detecting algorithm for motion detection, sweeping detection, zooming detection and a special wake-up gesture to resume the system from the sleeping mode and to avoid users from accidentally trigger the system.

Figure 40: A thermograph captured by the Heiman HTPA 16×16d sensor array. Note that red pixels represent higher temperature to differentiate the object from ambient temperature in blue pixels.

5.2 Proposed Hand Gesture Recognition Digital Signal Processor

Our proposed design for the processor's system block diagram is shown in Figure 41. Input integer data which represent pixels' temperature in Kelvin are taken directly from Heiman sensor array and quantized into 8-bit data (DIN). These data are written into three memories by a Write Circuit inside the Interface block. The first memory is to keep motion history image (MHI) which are the difference between two continuous frames. When a motion is detected by our motion detecting algorithm the next two frames will be stored in the 2nd and 3rd memory. The write circuit is controlled by a control circuit which is the control hub to communicate with other sensing modules. The control circuit is also responsible for multiplexing data to each of the sensing module. When a request signal is sent from a sensing module, the control circuit will fetch the data from memories to the requesting module. The analysis will be performed in each module and feedback to the control circuit if any other module needs to be activated. Finally, the control circuit will gather data from each module to make decisions about the gesture.

Figure 41: Block diagram of our proposed gesture recognition processor. Interface circuit consists of write circuits which are controlled by the control circuit to write 8-bit data into the three memories.

Targeting for ultra-low power applications, algorithm complexity has to be minimized to achieve fast response time with least hardware requirements. The proposed algorithms are designed to require only addition, subtraction and shifting operations while it stills ensure high detection accuracy for motion detection, sweeping and zooming detection with a distance up to 30 cm. For detecting wake-up gestures, it requires a shorter distance of below 20 cm as it needs to analyze the image's content and shorter distance also limits users from accidentally waking up the device by some unknown movements. Figure 42 shows the top-level design of gesture detecting algorithm which consists of two working modes. In standby mode, it disables half of its pixels and reduces the frame rate. This mode is triggered when there is no detected motion for a pre-set duration (e.g. 10 seconds). During standby, the processor only senses for motions then analyze if it is a wake-up gesture. If a wake-up gesture is found, the active mode will be recovered then the processor will turn on all the pixels and increase the frame rate. In active mode, the processor is able to detect sweeping actions in 8 directions (Up, Down, Left, Right and 4 diagonal directions) and zooming actions as shown in Figure 43. With two separated working modes, the energy consumption is reduced significantly as in standby mode, only

half of the pixels are used to detect the wake-up gesture and the system also works at reduced clock frequency due to no real-time processing is required for this working mode. In practice, it is observed that most of the time the system will fall into idle mode after having no activity. By having standby option, more than 80% of active power is cut-down to achieve lower overall energy consumption.

Figure 42: Top level design of detecting algorithms. Note that there are total 8 sweeping directions: 4 movements up/down, left/right in straight directions and 4 movements in diagonal directions.

Figure 43: The list of recognizable gestures (a) up-down (b) left-right (c) diagonal sweeping (c) zooming in/out (d) a unique wake-up gesture.

5.3 Gesture Recognition Algorithms

In this section, various gesture recognition algorithms are introduced. Since we are targeting for low power applications, detecting algorithms must have high accuracy but also simple hardware implementations to achieve lower power consumption. Algorithms with complicated features will require more hardware and higher power budget, therefore, our proposed algorithms try to minimize the complexity, hardware cost and only consists of simple components such as shifter, adders, subtractors, and comparators. With advantages of simplicity, the algorithms are more favorable for hardware implementation to achieve minimum size for design layout as well as improve the operating speed at ultra-low power consumption.

5.3.1 Calibration and Motion Detection

In order to differentiate between thermal frames which have motions and the ones without motion, object's reference temperature and background temperature need to be stored and updated so that referenced values are refreshed with changes in the environment. These storing and updating tasks are done by a calibration module which calculates the referenced temperature of the object and the background temperature for motion detection and for wake-up detection. Figure 44 shows the block diagram of the calibration circuit which gauges the background temperature value when there is no motion. This circuit is activated at power-on events and calibrates for 64 frames which are required to be the background images (without any object's presence). Calibration algorithm subtracts the current input from the previous input and updates the maximum pixel's variation reading. Input noise is canceled out by taking the difference of input, therefore, clearer images can be obtained as shown in Figure 45(a) which is a motion history image of the background

during the calibration mode. When the counter reaches 64 frames, the maximum value will be stored as a calibrated reference value for the background temperature. A pixel whose MHI's reading higher than the referenced temperature is considered as having motions. The motion detecting algorithm will count these pixels. If the number of pixels having motions is greater than a preset value, then the difference is significant enough to be considered as a movement. Figure 45(b) is an example of a detected movement. With a hand is moving into the sensor's focus, the number of pixels having temperature value higher than the referenced value keeps increasing and reach the threshold for motion detection. The motion detection circuit then records 75% of the highest pixel's value as the object's temperature and triggers a signal to inform the control circuit about its detected movement for further processing. In the event there is no movement for a period of time, the processor will be put into sleep mode which disables all the blocks but motion detection to save power until a movement is found. When there is a movement, other gesture detecting modules are activated to determine the gesture. The calibration threshold for several having motion pixels are set based on multiple experiments and it is reconfigurable to provide users with options to adjust the sensitivity level.

Figure 44: Circuit block diagram for calibration module. Note that write and read signal are performed within a half clock cycle.

Figure 45: (a) MHI thermograph without any object's presence or background image (b) MHI thermograph with object's presence.

5.3.2 Wake-up Gesture Detection

When there is no activity in front of the sensor for a preset timing, the algorithm triggers standby mode and put the DSP into the idle stage which only consumes a minimum amount of energy. In this stage, only motion detection module is enabled to capture moving activities. If there is any motion detected, the information about this movement will be analyzed by wake-up gesture detection module and if an intended wake-up gesture is found, the DSP will resume its normal operating mode to start more complex gesture recognitions. To avoid accidentally

waking up the DSP by some unknown hand movements, the wake-up gesture has to be unique so that it is not easy to be accidentally triggered, however, it must be simple to be recognized in low-resolution mode using less complex algorithms to optimize for power consumption. With these constraints, the gesture shown in Figure 46 is selected to be our wake-up gesture. This gesture is very common to human being for gesturing an acceptance or an agreement, besides it is also not easy to be unintentionally created as it requires the object to make a closed-loop of pixels which have high temperature as shown in the thermal graph of this wake-up gesture.

Figure 46: (a) A Wake-up gesture. (b) Thermal graph of a wake-up gesture. Note that the wake-up gesture should be simple to be recognized but unique to avoid being misclassified by the DSP.

To detect this pattern, the algorithm determines if there is any hole created by cool pixels (A cool pixel is defined as a pixel which has a thermal value smaller than the referenced value that is calibrated when the processor starts). Below is the pseudo-code for this algorithm:

- i. *Detecting motion by comparing 2 continuous frames. If having a motion then capture the image otherwise stay in this mode.*

- ii. *Scan all columns in the picture vertically to find an area of cool pixel bounded by hot pixels.*
- iii. *Pick one pixel in that area and scan in 8 directions (vertical, horizontal and diagonal)*
 - a. *If it is surrounded by all hot pixel then find one “hole pixel”*
 - b. *Otherwise, there is no hole in this area*
- iv. *If there are 3 “hole pixels” which are adjacent, then a hole is detected.*

5.3.3 Zooming Detection

When a motion is detected, there will be a sequence of object's location analysis performed by the processor. First, zooming detection module informs the control circuit to fetch the three most recent frames together with referenced temperature values to the coordinate calculation circuit. In this circuit module, object's locations in each frame are determined by finding a rectangular covering the whole object. This rectangular can be formed while scanning for object's pixels to locate its four corner positions. Once corners of the object are located, object's position is bounded by a rectangular and relative position of these rectangular from each frame will reveal the object's movement. After having the boundary information, coordinate calculation circuit passes the 4 sets of object's boundary coordinate to the corner checking circuit which compares if any coordinate has the value that equals to one of the four corners: (0,0), (0,16), (16,0) and (16,16). If it is not at the corner, then the comparator circuit starts comparing coordinates of the three frames. In the event of ascending or descending order is formed by the

coordinates from frame 1 to frame 3 then there is a zooming in or zooming out action. Otherwise, it is another type of gesture and it will be analyzed by other modules. The circuit block diagram for zooming detection module is displayed in Figure 47 and its operating concept is shown in Figure 48 accordingly.

It can be seen that when zooming out action is performed, object location in the first frames will cover its locations in the subsequent frames hence comparing the corner coordinates from each frame can disclose the real gesture. However, our proposed zooming detection has a limitation in interpreting zooming at the corner and when it happens the algorithm is only able to tell there is a movement at the corner but not the correct gesture. This is because when the user sweeps across the corners, the ascending or descending order will be formed, and it is misinterpreted as a zooming action. Therefore, corner checking is executed as the first step to avoid this situation.

Figure 47: Zooming detection block diagram which consists of coordinate calculation circuit, corner checking circuit, and a comparator bank.

Figure 48: A zooming action. Note that in the next frame, object shrinks inside its previous locations.

5.3.4 Sweeping Detection

When a motion is not classified as a zooming action, sweeping detection module will check if it is a sweeping action. In this detecting mode, motion history image (MHI) technique is employed. By taking the difference of two continuous frames, MHI is able to record both object's locations in the two frames. Figure 49 demonstrates operations of the sweep detecting algorithm. Two MHIs are obtained upon requesting data from the three memories then the processor takes an average of all the pixels in every row and in every column. These numbers form two distributions for rows' average pixel values and columns' average pixel values as shown in Figure 50.

For vertical sweeps, the column pixels of MHI cancel each other in a summation and generate no clear peak values as shown in Figure 50(a) while the row pixels of the same MHI are magnified in the summation, therefore, generating two clear peak values. A similar analysis is applied for diagonal sweeps in which overlapping area in both vertical and horizontal directions is canceled out while the non-overlapping area is magnified. This results in two clear peak values in each row average and column average as shown in Figure 50(b). In order to make a decision

about the gesture, peak counter and locator circuit are used in Figure 49 to count the number of peaks and locate the peaks' positions in two MHI frames. By looking at the number of peak values and their relative positions to each other from the two MHI frames, the direction of sweeping can be accurately concluded.

Figure 51 demonstrates the detecting mechanism for a right-left movement. In this example, there is no peak in rows summation suggesting that the object moves in a horizontal direction and in columns summation there are two peaks generated by summation of two non-overlapping objects. The first peak has a negative value and the second one is a positive peak whose position on the left of the first one. With this information, object movement is concluded as a horizontal sweeping gesture from right to left since right peak happens first and two peaks are both in the horizontal direction. For a curly hand movement shown in Figure 52, there is one negative peak formed in rows summation and two peaks formed in columns summation with negative peak happens before the positive peak. Based on the number of peaks and their relative position to each other, it is clear that there is an object's dislocation from right to left with tilted movements in the vertical direction, as a result, this gesture can be seen as a curly sweep from right to left.

In this proposed sweep detecting algorithm, MHI helps to zero out the intrinsic noise in each pixel and magnify the differences when there is a movement. Row averaging and column averaging technique additionally amplify the difference between the previous object's position and its current position. Therefore, applying row and column averaging on MHI frames generates clear detecting signals and accurate results while requires minimum hardware cost. The circuit is implemented in Verilog with only addition, subtraction and comparators to reduce the complexity

in order to increase processing speed and reduce power consumption. From simulations, the circuit is able to work reliably at a maximum clock frequency of 100 MHz.

Figure 49: Sweeping detection module block diagram. Sweeping gestures are detectable in 8 directions.

Figure 50: (a) A vertical sweeping gesture. (b) A diagonal sweeping gesture. Note that the peaks in column sum and row sums will tell the sweeping direction.

Figure 51: Sweeping detection mechanism. Note that summation of pixels values in overlapping area is close to zero while it is magnified in the non-overlapping area.

Figure 52: Sweeping detection for a curly hand movement. Note that this movement can be recognized with one peak in row summation and two peaks in column summation.

5.4 Implementations and Measurement Results

To verify our DSP's functionality, gesture recognition algorithms are programmed in Matlab, the testing input is taken from Heiman sensors which provide raw pixel temperature. Hand movements are then recorded for multiple gestures in

front of the sensor and output file which contains temperature value is processed into matrix format with each matrix represents a thermal graph of each frame. Our Matlab algorithm reads the frame data from sensor's output file and generates its detection for each frame then the results are compared visually with the frame image to ensure the accuracy. Upon successfully verifying the algorithm, the proposed design is implemented in Verilog with ModelSim simulation. Data from sensor's output file is converted into 8-bit binary data for testing and detection results are compared with Matlab before synthesizing the circuit in Cadence.

The final design is implemented with TSMS 65nm, the DSP layout occupies an area of $580 \times 580 \mu\text{m}$ for 16×16 array version and $580 \times 300 \mu\text{m}$ for 16×4 sensor array. Fabricated chips are tested with inputs from our sensor as well as a fixed set of input stream which is recorded from Heiman sensor for comparing with ModelSim's output. The results showed a perfect match between ModelSim's output and test chips' output. In this taped-out, two versions of DSP are produced. The smaller DSP version for testing with our custom AFE using 16×4 IR sensor array while the bigger DSP is tested with 16×16 Heiman IR sensor. Figure 53 is our gesture recognition SoC's micrograph which has a total area of 8.1 mm^2 . The major area is occupied by AFE and ADC part which account for 64% of the total layout area while the two DSPs only require 0.57 mm^2 .

Figure 53: The micrograph of our gesture recognition SoC. Note that processor area contains two version for 16×4 input data and 16×16 input data and the AFE part is not in the scope of this report.

5.4.1 Test results for the 16×4 IR sensory array

The test chips are measured against two sets of input data, one is with our custom AFE for basic functions including sweeping and wake-up gesture and the other is with Heimn sensor for full features. With 16×4 input array, the DSP is able to recognize hand sweeping accurately for input noise density of $31 \text{ nV}/\sqrt{\text{Hz}}$ in active mode and $130 \text{ nV}/\sqrt{\text{Hz}}$ in idle mode. In this measurement, a hand is swept in front of the sensor array and its temperature changes for three continuous frames are shown in Figure 54(a). In the first frame, pixels on the left edge start recording hand's temperature and turn into red then in the second frame with the hand fully covers the sensing array, all the sensor readings go higher to 30°C range. When the hand starts moving out of the focus zone, the number of hot pixels decreases, and right edge pixels record the hand temperature before it exits. The response code from the DSP is shown in Figure 54(b) for further elaboration. When the object enters the focus zone, reading value increases and when it sweeps across the sensors, two peak values in the

motion history image are obtained to confirm the horizontal movement from left to right. In this active or detection mode, the system requires $260\text{ }\mu\text{W}$ in which $28.6\text{ }\mu\text{W}$ or 11% is consumed by the DSP, 55% is from instrumental amplifiers and the remaining is for LPF and ADC. In power saving or idle mode, only $46\text{ }\mu\text{W}$ is required by the whole system and the DSP accounts for 2% of the energy consumptions due to only turning on motion detection circuit at the reduced frame rate. The details of power contribution are presented in Figure 55 with the supply voltage is set at 1.4 V and sampling frequency of 1 kHz.

Figure 54: (a) Measurement temperature of a hand sweeping left to right. (b) Response code from the DSP.

Figure 55: Power contribution of the hand gesture recognition SoC in active and power saving mode.

5.4.2 Test results for the 16×16 IR sensory array

For the 16×16 sensor array version, test data is taken from Heiman sensors and the DSP is verified by detecting different types of gesture. First, a wake-up pattern is sent to the input. Figure 56(a) shows a wake-up gesture with a closed loop highlighted in red and output response is shown in Figure 56(b). The response image taken from processor's memory has three adjacent low-temperature pixels surrounded by a closed loop of high-temperature pixels indicating the wake-up pattern's location which is correctly detected by the algorithm.

Figure 56: (a) Test image for a wake-up gesture with a closed loop highlighted in red (b) output detection with 3 adjacent detected pixels to indicate a closed loop.

After putting the system into active mode, sweeping and zooming gestures are performed in front of Heimann sensors to obtain digital input for the detecting algorithms. Figure 57(a) demonstrates a zooming in detection which has an output response of “11”, for zooming out the response is “00” and “01” is the default mode. This digital pattern is chosen so that circuit only needs to toggle one bit for any detection. For sweeping detection, a three-bit output and the number of peaks is used to indicate the directions. In Figure 57(b), a horizontal sweep is performed therefore there are total 2 peaks and sweep output is “110” which is encoded for the right to left direction.

Figure 57: (a) Output test results for zooming detection, zoom=11 indicates zooming out. (b) Output test results for sweeping detection, sweep=110 indicate a right to left sweep with 2 peaks in total. Note that in MHI image, blue color indicates object's previous position and red color indicates object's current position.

The recognition rate is then evaluated for sweeping actions. A series of Right-Left sweeps were performed at different speeds and distances from the sensors. All the movements are recorded in video format and processed with frame-cutting software provided by Heimann. The number of frames having Right-Left sweep is counted and their indexes are recorded to compare with the results obtained from the processor. Figure 58(b) shows the detection rate of our proposed algorithm. At a very near distance, 10 cm from the sensor it is more suitable for slow motions as the object become bigger than itself at a further distance. From 20-30 cm, the gesture detection processor has an accuracy rate of 96% at its optimal speed which generates 12 frames across the covered area of the sensor's lens. This detection accuracy is

slightly higher than a recent PIR hand gesture detection research work reported in [59]. For very fast motions which produced less than 4 frames crossing the sensor's focus, there is not enough data to process therefore it does not generate any recognition results and we excluded them from the counting.

Figure 58: Detection accuracy for Right-Left sweeps at different sweeping speed and at different object's distance to the sensor.

In term of power consumption, the DSP's measured power consumption at the supply of 1.8 V with 12 fps (frames per second) input is 66.8 μ W and 23.6 μ W for active and idle mode respectively. Figure 59 shows power consumption breakdown details of our DSP in which the switching and internal energy contribute to 96% of the total power while 4% is caused by leakage energy since most of the time DSP is under sleep mode whose energy consumption is dominated by leakage current. For energy breakdown by components, memory takes the largest contribution of 45% followed by 40% from gesture detecting modules and remaining is coming from other interfacing circuits. In comparison with previous state-of-the-art gesture detections as shown in Table III, this work presents the first SoC for gesture sensing applications using passive IR sensors. With low power supply and

low power consumption features, the design is the best fit for emerging smart devices and mobile applications.

Figure 59: Power consumption breakdown details. Note that dynamic energy and power consumption from memory circuits are major contributors.

Table III: Comparison with previous gesture detection works. Note that this work is the first gesture detection SoC that has on-chip processor instead of using software processing as in other reported works.

Specifications	This work	JSSCC '13 [45]	Pyreos [52]	Robio '17 [60]
System Type	SOC	Only Sensor	PCB	FPGA
Sensor type	Passive IR	RF	Passive IR	Image
Area	580 $\mu\text{m} \times 580 \mu\text{m}$	4 mm $\times$ 5 mm	1.5 mm $\times$ 1.5 mm	N.A
Resolution	16 $\times$ 16	400 $\times$ 384	2 $\times$ 2	640 $\times$ 480
Frame rate (fps)	10 - 30	200	72	60
Voltage Supply	1.2V - 1.8 V	1.8 V	3.2 V	N.A
Power consumption	66.8 μW	140 mW	46.2 μW (only for sensors)	N.A

Chapter 6 Conclusions and Future Works

6.1 Conclusions

In this thesis, we tackled ultra-low power design problems from many aspects including system's perspective, architecture design, algorithm design and many circuit optimization techniques. The thesis demonstrates design methodology with two processors, a fast Fourier transform and a hand gesture recognition DSP. FFT processor is one of the most important signal processing modules which is used by many applications, therefore, any improvement with FFT cores shall create a big impact on the overall system level. Hand gesture recognition in another hand is an application-oriented design which is very useful for mobile and smart devices. Having a small, compact ultra-low power hand gesture SoC would change the interaction between human and devices. Instead of direct contact with the devices, this technology offers a more convenient way to control devices by using hand gestures and the proposed SoC is the first chip level to fulfill the demand.

In chapter 2 and chapter 3, we discussed advantages and disadvantages of current FFT topologies. Among different algorithms, Radix-2^k FFT requires minimum hardware components but at lower throughput and to increase the throughput, either multiple parallel processing lanes are designed, or MDC architecture is used. Both approaches demand more hardware and increase circuit complexity. However, it does meet the requirements for low power and high throughput, consequently, there is no or very few research interests to develop better topologies. With recent developments in subthreshold circuit designs, more and more applications are pushed to ultra-low voltage domain and FFT cores need to be improved both in throughput and power

consumption. Existing FFT topologies seem difficult to catch up with the new requirements, therefore, FFT architecture must be re-designed to accommodate faster and low power structure. A new FFT topology is proposed in Chapter 3, named Radix-2² Feedforward FFT. In this architecture, radix-2² algorithm is used to utilize its hardware saving and simple butterfly structure while multiple delay commutator is implemented to further enhance the throughput. A new input scheduling algorithm is also introduced to eliminate the delay line and shifting energy associated with it. As a result, power is reduced significantly, and the speed is improved with register-based memory. Using optimization techniques to determine the optimal level of parallelism, optimal energy point, and best circuit's performance operating region, the final design is able to achieve the lowest energy/FFT conversion at sub-Giga sample/s throughput. The proposed 1024-point Radix-2² feedforward FFT core also marks the first chip level Radix-2² feedforward architecture to open a new research direction in feedforward FFT.

In the last two chapters, chapter 4 and chapter 5 of this thesis, an application-oriented design for a hand gesture recognition processor is presented. As a human-machine interface, hand gesture DSPs offer a new experience for users of smart devices. The users are now able to control the devices by moving hands over the sensor area, this feature is especially attractive for those devices such as smart watches whose display screen is too small for using the touchscreen. Chapter 4 overviews the available sensing technologies and compares their specifications. Active sensing elements such as ultrasound sensors, RF sensors, and image sensors provide very good image resolution and compatible with complex detecting algorithms to perform more advanced gestures. However, they come with the cost of

larger chip layout and higher energy consumption, even in the event when there is no object's appearance the system still emits signals to detect objects and generates wasted energy. To overcome this limitation, passive sensing devices using infrared signal radiated from human hand is used to detect the hand's movement. Without using transmitters, IR sensors detect heat from human hands and generate a thermal image of the object at a lower resolution than active sensing devices but at much lower power consumptions. Especially, when there is no object the sensor does not consume active energy but only a small amount of power to keep the sensors being turned on. With lower power advantage, IR sensors are more suitable for battery devices and hence being selected for implementations with our hand gesture recognition DSP in Chapter 5. In this chapter, the first SoC hand gesture recognition is designed and tested with real-time gestures using input from a customized IR sensor array as well as from a commercial Heiman sensor array. The system successfully demonstrates the ability to recognize sweeping, zooming and wake-up gestures using 260 μW at a frequency of 1 kHz and 1.4V supply for 16×4 images. With higher resolution image of 16×16 , the DSP consumes 66.8 μW when detecting gestures. This feature makes the design very suitable for emerging smart devices such as smartphones and smartwatches.

6.2 Future Works

This thesis concluded with two designs for ultra-low power applications. In spite of successful functionalities and its advantages over existing works, there are still potential improvements which can help current designs achieve better power consumption and operating speed. For the FFT cores, the area for improvements is:

- To optimize memory: When FFT's length and word width grow, there will be more power consumed by the memory. Besides leakage power will contribute significantly when voltage supply is lowered. The current design uses register-based memories which have more transistor count and lower energy density than SRAM design. Therefore, using SRAM shall reduce the layout area and improve energy consumption.
- Redesign multipliers: Multiplier is one of the major components in FFT, a sub-threshold pipeline multiplier should be designed and implemented to shrink down the computation power.

Our proposed hand gesture recognition DSP is the first SoC reported which is able to work at ultra-low power consumption and low voltage supply. However, the number of detectable gesture is still limited and there are many rooms for future development:

- Define more gestures including sign language and develop better algorithms to build a comprehensive set of detectable gestures.
- Optimize energy consumption using dynamic voltage and frequency scaling technique.
- Design new techniques to work in high-temperature environment: Currently, the system is only workable when the background temperate is smaller than human hand temperature, however, the logic could be toggled to make the design more robust and independent of surrounding temperature.

Author's Publications

- N. L. Ba and T. Kim, "An Area Efficient 1024-Point Low Power Radix-2² FFT Processor with Feed-Forward Multiple Delay Commutators," *IEEE Transactions on Circuits and Systems I*, pp. 1-9, May 2018.
- N. L. Ba and T. Kim, "Design of Temperature-Aware Low Voltage 8T SRAM in SOI Technology for High Temperature Operation (25 ~ 300 °C)," *IEEE Transactions on VLSI Systems (TVLSI)*, Vol. 25, pp. 2383-2387, Aug. 2017.
- S. Oh, N. L. Ba, S. Bang, J. Jeong, D. Blaauw, T. Kim, and D. Sylvester, "A 260μW Infrared Gesture Recognition System-on-Chip for Smart Devices," *IEEE Symposium on VLSI Circuits (SOVC)*, pp. 228-229, June 2016.
- T. Kim, and N. L. Ba, "Design of a Temperature-Aware Low Voltage SRAM with Self-Adjustable Sensing Margin Enhancement for High Temperature Applications up to 300°C," *IEEE Journal of Solid-State Circuits (JSSC)*, Vol. 49, pp. 2534-2546, Nov. 2014.
- T. Kim, N. Ba, A. Do, J. Gopal, G. Chua, and P. Singh, "Low Power Memory Design for High Temperature in Ruggedized Electronics," *11th International SoC Design Conference (ISOCC)*, pp. 110-111, Nov. 2014.
- T. Kim and N. L. Ba, "A Low Voltage 8-T SRAM with PVT-Tracking Bitline Sensing Margin Enhancement for High Operating Temperature (up to 300°C)," *IEEE Asian Solid-State Circuits Conference (A-SSCC)*, pp. 233-236, Nov. 2013.

References

- [1] G. E. Moore. (1965) Cramming More Components onto Integrated Circuits. *Electronics Magazine*.
- [2] B. Nikolic, "Design in the Power-Limited Scaling Regime," *IEEE Transactions on Electron Devices*, vol. 55, pp. 71 - 83, 2008.
- [3] M. Horowitz, T. Indermaur, and R. Gonzalez, "Low-Power Digital Design," presented at the IEEE Symposium on Low Power Electronics, San Diego, CA, USA, 1994.
- [4] D. Jeon, M. Seok, Chaitali Chakrabarti, D. Blaauw, and D. Sylvester, "A Super-Pipelined Energy Efficient Subthreshold 240 MS/s FFT Core in 65 nm CMOS," *IEEE Journal of Solid-State Circuits*, vol. 47, pp. 23 - 34, January 2012.
- [5] N. L. Ba and T. T.-H. Kim, "An Area Efficient 1024-Point Low Power Radix-2² FFT Processor with Feed-Forward Multiple Delay Commutators," *IEEE Transactions on Circuits and Systems I*, pp. 1 - 9, 10 May 2018 2018.
- [6] S. Oh, N. L. Ba, S. Bang, J. Jeong, D. Blaauw, T. Kim, *et al.*, "A 260 μ W Infrared Gesture Recognition System-on-Chip for Smart Devices," presented at the Symposium on VLSI Circuits (SOVC), Honolulu, HI, USA, June 2016.
- [7] *Sparse Fast Fourier Transform*. Available: <https://groups.csail.mit.edu/netmit/sFFT/algorithm.html>
- [8] R. H. Seireg and A. E. Barbour, "A new algorithm for pattern recognition of voices," in *IEEE Circuits and Systems*, Washington, DC, USA, 1992.
- [9] H. Buijs, A. Pomerleau, M. Fournier, and W. Tam, "Implementation of a fast Fourier transform (FFT) for image processing applications," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 22, pp. 420-424, 1974.
- [10] S. S. Agaian and O. Caglayan, "Fast Encryption Method Based on New FFT Representation for the Multimedia Data System Security," presented at the IEEE International Conference on Systems, Man and Cybernetics, Taipei, Taiwan, 2006.
- [11] J. Cooley, P. Lewis, and P. Welch, "Historical notes on the fast Fourier transform," *IEEE Transactions on Audio and Electroacoustics*, vol. 15, pp. 76 - 79, 1967.
- [12] I. J. Good, "The interaction algorithm and practical Fourier analysis," *Journal of the Royal Statistical Society*, pp. 361–372, 1958.
- [13] G. Bruun, "z-Transform DFT filters and FFTs," *IEEE Trans. on Acoustics, Speech and Signal Processing (ASSP)* vol. 26, pp. 56-63, 1978.
- [14] J. B. Birdsong and N. I. Rummelt, "The hexagonal fast fourier transform," presented at the IEEE International Conference on Image Processing (ICIP), 25-28 Sept. 2016.
- [15] K. R. Rao, D. N. Kim, and J. J. Hwang, *Fast Fourier Transform: Algorithms and Applications*. Netherlands: Springer, 2010.
- [16] S. He and M. Torkelson, "A New Approach to Pipeline FFT Processor," in *IEEE Proceedings of IPPS*, Honolulu, HI, USA, USA, 1996.
- [17] B. M. Baas, "A Low-Power, High-Performance, 1024-Point FFT Processor," *IEEE Journal of Solid-State Circuits*, vol. 34, pp. 380 - 387, 1999.

- [18] C. Yu, M.-H. Yen, P.-A. Hsiung, and S.-J. Chen, "A low-power 64-point pipeline FFT/IFFT processor for OFDM applications," *IEEE Transactions on Consumer Electronics*, vol. 57, pp. 40 - 40, 2011.
- [19] V. Stojanovic. Course materials for 6.973 Communication System Design [Online].
- [20] E. H. Wol and A. M. Despain, "Pipeline and Parallel-Pipeline FFT Processors for VLSI Implementations," *IEEE Transactions on Computers*, vol. C-33, pp. 414 - 426, May 1984.
- [21] N. Li and N. P. V. D. Meijs, "A Radix 2^2 Based Parallel Pipeline FFT Processor for MB-OFDM UWB system," in *IEEE International SOC Conference*, Belfast, UK, 2009.
- [22] S.-N. Tang, J.-W. Tsai, and T.-Y. Chang, "A 2.4-GS/s FFT Processor for OFDM-Based WPAN Applications," *IEEE Transactions on Circuits and Systems II*, vol. 57, pp. 451 - 455, 24 May 2010.
- [23] H. Liu and H. Lee, "A High Performance Four-parallel 128/64-point Radix- 2^4 FFT/IFFT Processor for MIMO-OFDM Systems," presented at the IEEE Asia Pacific Conference on Circuits and Systems, Macao, China, 2008.
- [24] V. Stojanovic. Fast Fourier Transform: VLSI Architectures, course materials for Communication System Design, MIT OpenCourseWare (<http://ocw.mit.edu>) [Online].
- [25] E. E. Swartzlander, W. K. W. Young, and S. J. Joseph, "A radix 4 delay commutator for fast Fourier transform processor implementation," *IEEE Journal of Solid-State Circuits*, vol. 19, pp. 702 - 709, 1984.
- [26] Wold and Despain, "Pipeline and Parallel-Pipeline FFT Processors for VLSI Implementations," *IEEE Transactions on Computers*, vol. C-33, pp. 414 - 426, 1984.
- [27] A. M. Despain, "Fourier Transform Computers Using CORDIC Iterations," *IEEE Transactions on Computers*, vol. C-23, pp. 993 - 1001, 1974.
- [28] K. Harikrishna, T. R. Rao, and V. A. Labay, "An FFT Approach for Efficient OFDM Communication Systems," presented at the International Conference on Signal Acquisition and Processing, Bangalore, India, 2010.
- [29] S. A. Ajmal and S. L. Gangadharaiyah, "FPGA based area optimized parallel pipelined radix-22 feed forward FFT architecture," presented at the IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT) Bangalore, India, 2016.
- [30] C.-H. Yang, T.-H. Yu, and D. Marković, "Power and Area Minimization of Reconfigurable FFT Processors: A 3GPP-LTE Example," *IEEE Journal of Solid-State Circuits*, vol. 47, pp. 757 - 768, March 2012.
- [31] Y. Chen, Y.-W. Lin, Y.-C. Tsao, and C.-Y. Lee, "A 2.4-Gsample/s DVFS FFT Processor for MIMO OFDM Communication Systems," *IEEE Journal of Solid-State Circuits*, vol. 43, pp. 1260 - 1273, May 2008.
- [32] G. Bi and E. V. Jones, "A pipelined FFT processor for word-sequential data," in *IEEE Trans. Acoust, Speech, Signal Processing*, 1989, pp. 1982-1985.
- [33] M. Garrido, J. Grajal, M. A. Sánchez, and O. Gustafsson, "Pipelined Radix- 2^k Feedforward FFT Architectures," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, pp. 23 - 32, 29 December 2011.

- [34] A. Wang and A. Chandrakasan, "A 180-mV Subthreshold FFT Processor Using a Minimum Energy Design Methodology," *IEEE Journal of Solid-State Circuits*, vol. 40, pp. 310 - 319, January 2005.
- [35] M. Sjalander and P. Larsson-Edefors, "High-speed and low-power multipliers using the Baugh-Wooley algorithm and HPM reduction tree," presented at the IEEE International Conference on Electronics, Circuits and Systems, St. Julien's, Malta, 2008.
- [36] S. Thompson, P. Packan, and M. Bohr, "MOS Scaling: Transistor Challenges for the 21st Century," *Intel Technology Journal*, 1998.
- [37] B. Zhai, D. Blaauw, D. Sylvester, and K. Flautner, "The Limit of Dynamic Voltage Scaling and Insomniac Dynamic Voltage Scaling," *IEEE Transaction On Very Large Scale Integration (VLSI) Systems*, vol. 13, pp. 1239 - 1252, Nov 2005.
- [38] P. Gupta, "Accurate performance analysis of a fixed point FFT," in *Twenty Second National Conference on Communication (NCC)*, 2016, pp. 1 - 6.
- [39] L. Jaemin, H. Takimoto, H. Yamauchi, A. Kanagawa, and Y. Mitsukura, "A Robust Gesture Recognition based on Depth Data," presented at the The 19th Korea -Japan Joint Workshop on Frontiers of Computer Vision, Incheon, South Korea, 2013.
- [40] J. Suarez and R. R. Murphy, "Hand Gesture Recognition with Depth Images: A Review," presented at the IEEE International Symposium on Robot and Human Interactive Communication, Paris, France, 2012.
- [41] Microsoft. (2014). Available: <http://www.xbox.com/en-sg/>
- [42] P. Trindade, J. Lobo, and J. P. Barreto, "Hand Gesture Recognition Using Color and Depth Images Enhanced with Hand Angular Pose Data," presented at the IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), Hamburg, Germany.
- [43] B. Mrazovac, M. Z. Bjelica, D. Simic, S. Tikvic, and I. Papp, "Gesture Based Hardware Interface for RF Lighting Control," presented at the IEEE 9th International Symposium on Intelligent Systems and Informatics, Subotica, Serbia.
- [44] R. Bainbridge and J. A. Paradiso, "Wireless Hand Gesture Capture Through Wearable Passive Tag Sensing," presented at the International Conference on Body Sensor Networks, Dallas, TX, USA, 2011.
- [45] A. Arbabian, S. Callender, S. Kang, M. Rangwala, and A. M. Niknejad, "A 94 GHz mm-Wave-to-Baseband Pulsed-Radar Transceiver with Applications in Imaging and Gesture Recognition," *IEEE Journal of Solid-State Circuits*, vol. 48, pp. 1055 - 1071, 2013.
- [46] R. J. Przybyla, S. E. Shelton, A. Guedes, R. Krigel, D. A. Horsley, and B. E. Boser, "In-Air Rangefinding with an AlN Piezoelectric Micromachined Ultrasound Transducer," *IEEE Sensors Journal*, vol. 11, pp. 2690 - 2697, July 2011.
- [47] R. Carotenuto, R. Ionescu, P. Tripodi, and F. Urbani, "Three Dimensional Ultrasound Gestural Interface," presented at the IEEE International Ultrasonics Symposium, Rome, Italy, 2012.
- [48] Q. H. Christoph Kuratli, "A CMOS Ultrasound Range Finder Microsystem," presented at the IEEE International Solid-State Circuits Conference, San Francisco, CA, USA, 2000.

- [49] P. Muralt, N. Ledermann, J. Baborowski, A. Barzegar, S. Gentil, B. Belgacem, *et al.*, "Piezoelectric Micromachined Ultrasonic Transducers Based on PZT Thin Films," *IEEE Transaction On Ultrasonics, Ferroelectric and Frequency Control*, vol. 52, pp. 2276 - 2288, Dec 2005.
- [50] R. Przybyla, A. Flynn, V. Jain, S. Shelton, A. Guedes, I. Izyumin, *et al.*, "A Micromechanical Ultrasonic Distance Sensor with >1 Meter Range," presented at the Proc. Solid-State Sensors, Actuators and Microsystems Conference Beijing, China, 2011.
- [51] P. Wojtczuk, A. Armitagey, T. D. Binnie, and T. Chamberlainz, "PIR Sensor Array for Hand Motion Recognition," presented at the The Second International Conference on Sensor Device Technologies and Applications, 2011.
- [52] *Pyreos Ltd: Touchless Gesture Control*. Available: <http://www.pyreos.com/>
- [53] H. Kawanishi, K. Hishinuma, M. Kimura, N. Motai, Y. Soutome, T. Tsukii, *et al.*, "64/256-Element Thermopile, Infrared Sensor Chip with 4 Built-In Amplifiers for use in Atmospheric Pressure Conditions," presented at the IEEE International Solid-State Circuits Conference, San Francisco, CA, USA, 2008.
- [54] M. Hu, "Visual Pattern Recognition by Moment Invariants," *IRE Trans. Information Theory*, vol. 8, pp. 179-187, 1962.
- [55] Mahalanobis, "On the generalised distance in statistics," *Proceedings of the National Institute of Sciences of India* pp. 49-55, 1936.
- [56] A. F. Bobick and J. W. Davis, "The Recognition of Human Movement Using Temporal Templates," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 23, pp. 257-267, 2011.
- [57] A. Saxena, D. K. Jain, and A. S. , "Hand Gesture Recognition using an Android Device," presented at the International Conference on Communication Systems and Network Technologies, 2014.
- [58] H. S. GmbH. (2014). *Thermopile Arrays and Imaging*. Available: http://www.heimannsensor.com/products_imaging.php
- [59] T. D. Binnie, A. F. Armitage, and P. Wojtczuk, "A passive infrared gesture recognition system," presented at the IEEE Sensors, Glasgow, UK, 2017.
- [60] W. Zhou, C. Lyu, and X. Jiang, "Real-time implementation of vision-based unmarked static hand gesture recognition with neural networks based on FPGAs," presented at the IEEE International Conference on Robotics and Biomimetics (ROBIO), Macau, China, 2017.