Design implementation low energy fast fourier transform/inverse fast fourier transform (FFT/IFFT) processor based on asynchronous-logic

Chong, Kwen Siong

2007

Chong, K. S. (2007). Design implementation low energy fast fourier transform/inverse fast fourier transform (FFT/IFFT) processor based on asynchronous-logic. Doctoral thesis, Nanyang Technological University, Singapore.

https://hdl.handle.net/10356/3447

https://doi.org/10.32657/10356/3447

DESIGN AND IMPLEMENTATION OF A LOW ENERGY FAST FOURIER TRANSFORM / INVERSE FAST FOURIER TRANSFORM (FFT/IFFT) PROCESSOR BASED ON ASYNCHRONOUS-LOGIC

Chong Kwen Siong

School of Electrical and Electronic Engineering

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

2007

Acknowledgements

Throughout this PhD program, I owe much to my main supervisor, Dr Gwee Bah Hwee, and co-supervisor, Dr Joseph S. Chang. I attribute to them for kindling my interest in research, particularly in the solid-state circuit design and later more specifically, asynchronous-logic design. Dr Gwee gave me many invaluable advice, encouragement, and support throughout my candidature. He has helped me to develop a positive attitude toward difficulty. Dr Chang, a philosopher I could say, inspired me to look at my project and even my life in different perspectives. He has motivated me, through writing and speaking, to think as critically as possible; I consider this critical thinking priceless.

Countless thanks and appreciation to those who have helped me with this thesis – especially those who have instructed and guided me and whom I have worked and studied with. They include Mr Loy Teck Pui, Mr Victor Adrian, Miss Ng Lay Suan, Mr Thet Naing Kyaw, Mr Maung Myo Tun, Mr Chua Chien Chung, Mr Chang Khia Ho, Mr Lim Khoon Aun, Mr Zhao Qingda, Mr Yang Jin, Mr Victor Putra Lesmana, Mr Ricky Setiawan, and Miss Xu Wen. Their friendship is greatly appreciated.

I would like to express my gratitude to my parents and family members for their consistent support and understanding throughout my life. Finally, I would not forget my love, Miss Kee Pei Ling, for her patience, sharing, and support, especially accompanying me to the lab during the weekends and holidays. This thesis is dedicated to her.

This research work was funded by the ASEAN-EU University Network Programme (AUNP), the Singapore Millennium Foundation PhD Scholarship, and the Nanyang Technological University Research Student Scholarship. The support I received from these organizations and foundation is greatly appreciated.

Table of Contents

Acknowledgments i

Table of Contents iii

Summary vi

List of Figures viii

List of Tables xii

Symbols and Abbreviations xiv

Chapter 1 Introduction 1

1.1 Motivation 1

1.2 Objectives 5

1.3 Contributions 6

1.4 Organization 11

Chapter 2 Literature Review 13

2.1 Low Power Low Energy Design Techniques 13

2.1.1 Mechanisms of Power Dissipation 13

2.1.2 Low Power/Energy Design Methodologies 18

2.1.3 Noise Margin 20

2.2 Synchronous-Logic and Asynchronous-Logic Design Approaches 21

2.2.1 Synchronous-Logic Design Approach 21

2.2.2 Asynchronous-Logic Design Approach 25

2.2.2.1 Fundamental Concepts of Asynchronous-Logic Design 26

2.2.2.2 Asynchronous-Logic EDA Tools 34

2.2.2.3 Basic Asynchronous-Logic Microcells 36

2.2.2.4 Potential Advantages of the Asynchronous-Logic Approach 43

2.3 Fast Fourier Transform and Inverse Fast Fourier Transform (FFT/IFFT) 47

2.4 Summary 51

Chapter 3 Asynchronous-Logic Microcells 53
3.1 Introduction 53
3.2 Proposed Latch Adder and Latch Accumulator 56
3.2.1 Latch Adder 56
3.2.2 Latch Accumulator 63
3.3 Asynchronous-Logic Adder Cell 66
3.3.1 Review of Asynchronous-Logic Adder Cells 66
3.3.2 Proposed 2-bit Asynchronous-Logic *Type-*γAdder 71
3.4 Proposed Broad B Latch Controller 75
3.5 Summary 77

Chapter 4 Macrocells: Multiplier and Memory Designs 79
4.1 Introduction 79
4.2 Multiplier Macrocell 82
4.2.1 Review of Multiplier Designs 82
4.2.2 Proposed 16×16-bit Booth Array-Based Multiplier Core 86
4.2.3 Proposed Asynchronous-Logic 16×16-bit Control-Multiplier 98
4.3 Proposed Asynchronous-Logic 128×16-bit Memory 102
4.4 Summary 108

Chapter 5 Synchronous-Logic and Asynchronous-Logic FFT/IFFT Processors 110
5.1 Introduction 110
5.2 Synchronous-Logic FFT/IFFT Processor 112
5.3 Proposed Asynchronous-Logic FFT/IFFT Processor 116
5.3.1 Control Portion 125
5.3.2 Data Paths 128
5.3.2.1 Memory Data Path 128
5.3.2.2 Shifter Data Path 129
5.3.2.3 Multiplier Data Path 131
5.3.2.4 Adder Data Path 132
5.3.2.5 Write Back Data Path 133
5.4 Measurements and Comparisons 134

5.5 Summary 148

Chapter 6 Conclusions and Recommendations 149

6.1 Conclusions 149

6.2 Recommendations: Future Research 152

6.2.1 Development of Advanced Asynchronous-Logic Design Tools 152

6.2.2 Design of Microcells and Macrocells 152

6.2.3 Leakage Current Reduction by Asynchronous-Logic Techniques in Advanced CMOS Processes 154

6.2.4 Adoption of Asynchronous-Logic Circuits in the Sub-Threshold Region 155

Author's Publications 156

Bibliography 158

Summary

This thesis pertains to circuit designs using the promising asynchronous-logic (async) approach as opposed to the prevalent synchronous-logic (sync) approach, with emphases on low voltage operation and low energy dissipation. The circuits designed herein span from low-level circuits (microcells, where the transistor count is < 100), mid-level circuits (macrocells, where the transistor count is > 100) to a complete system – a Fast Fourier Transform/Inverse Fast Fourier Transform (FFT/IFFT) processor that in part embodies these microcells and macrocells. The application of the processor is for energy-critical portable audio biomedical applications, including hearing aids.

The novel microcells include a Latch Adder (LA), a Latch Accumulator, a *Type-*γ adder (a 2-bit async carry completion sensing adder), and an async latch controller in broad data validation. The novelty of these microcells includes the incorporation of different functional features (e.g. latch and delay enhancements), resulting in increased versatility, compactness, and lower energy dissipation. The novel macrocells include a 16×16-bit Booth array-based multiplier core, a 16×16-bit 'Control-Multiplier' specifically for an FFT algorithm, and a 128×16-bit memory macrocell. The novelty of these macrocells includes reduced spurious switching (e.g. by means of latching and gating), resulting in reduced energy dissipation. The functionality and attributes of these microcells and macrocells are verified by computer simulations and on the basis of practical measurements on prototype ICs.

The FFT/IFFT processor is an async 128-point radix-2 decimation-in-time FFT/IFFT processor. The prototype processor IC dissipates 93nJ – 201nJ per FFT/IFFT computation @ V_{DD} = 1.0V – 1.4V, features a delay of < 1.95ms @ $V_{DD} \geq$ 1.0V, and occupies 1.6mm^2 IC area @ 0.35μm CMOS process. When benchmarked against its sync counterpart with and without clock gating, the async processor dissipates ~ 37% and ~ 50% lower energy respectively. The minor drawbacks of the async processor are a largely inconsequential 10% larger IC area and an inconsequential 1.4 times worse delay (inconsequential due to the intended low speed (< 5MHz) hearing aid application).

The work reported in this thesis demonstrates that by means of appropriate design techniques, designs based on the promising async approach can feature superior (lower) energy attributes over designs based on the prevalent sync approach.

List of Figures

Fig. 2.1	Charging and discharging of a CMOS circuit during switching	14
Fig. 2.2	Power/energy methodology hierarchy	18
Fig. 2.3	A sync circuit	22
Fig. 2.4	Clock gating approach	24
Fig. 2.5	(a) Two-phase handshaking protocol , and (b) four-phase handshaking protocol	28
Fig. 2.6	Four channel types: (a) nonput, (b) push, (c) pull, and (d) biput	29
Fig. 2.7	Push channel data valid schemes for (a) two-phase handshaking protocol, and (b) four-phase handshaking protocol	31
Fig. 2.8	A simple four-phase bundled data pipeline	33
Fig. 2.9	*C*-Muller circuit: (a) schematic, and (b) symbol	37
Fig. 2.10	Different *C*-Muller gate designs: (a) three-input asymmetric *C*-Muller gate with low reset, (b) three-input asymmetric *C*-Muller gate with high reset, (c) two-input asymmetric *C*-Muller gate with low reset, and (d) two-input asymmetric *C*-Muller gate with high reset	38
Fig. 2.11	Different latch designs: (a) TG latch, (b) NSP latch, (c) Trans latch, and (d) Mux latch	39
Fig. 2.12	Different async latch controller designs: (a) Broadish, and (b) Broad A	41
Fig. 2.13	The pulse generator (a) schematic, and (b) its timing diagram	42
Fig. 2.14	A radix-2 butterfly	49
Fig. 2.15	Flow chart of the conditional BFP scaling for one FFT stage	51
Fig. 3.1	A symbolic template for library cells	54
Fig. 3.2	Conventional full adders that provide rail-to-rail signal swing at low voltage operation: (a) T28 adder, (b) T18 adder, (c) T16 adder, (d) *N-P* DA, and (e) Domino DA	57
Fig. 3.3	Circuit schematic of the Latch Adder (LA)	59
Fig. 3.4	The simplified block diagram of an accumulator realized by usual design (a standard adder and a separate flip-flop)	64
Fig. 3.5	The proposed Latch Accumulator	65

Fig. 3.6	Microphotograph of the proposed 16-bit Latch Accumulator	66
Fig. 3.7	Conventional carry completion sensing adders (CCSAs): (a) 1-bit dynamic CCSA, (b) 1-bit pass-logic CCSA, (c) 2-bit *Type-*α CCSA, and (d) 2-bit *Type-*β CCSA	69
Fig. 3.8	Circuit schematic of the proposed CCSA, *Type-*γ	72
Fig. 3.9	Block diagram of a 16-bit async adder embodying the proposed *Type-*γ CCSAs	73
Fig. 3.10	The proposed latch controller: Broad B latch controller	76
Fig. 4.1	A simplified block diagram of a parallel multiplier	83
Fig. 4.2	The partial products in a 2's complement 16×16-bit Booth radix-4 multiplier	87
Fig. 4.3	Block diagram of the proposed 16×16-bit Booth array multiplier, SB-Proposed16	88
Fig. 4.4	Delay circuit	90
Fig. 4.5	Power breakdown (in % and μW) of different array multipliers based on simulations @ 1.1V, 1MHz	91
Fig. 4.6	Switching analysis in the Adder Block of various multipliers	94
Fig. 4.7	Microphotograph of the proposed multiplier (SB-Proposed16) and the conventional multiplier (SB-General16)	95
Fig. 4.8	The test jig for the multiplier IC	96
Fig. 4.9	Block diagram of the Control-Multiplier	99
Fig. 4.10	Microphotograph of the Control-Multiplier (part of the FFT/IFFT prototype IC)	101
Fig. 4.11	Block diagram of the 128×16-bit single-port async memory	102
Fig. 4.12	Circuit schematic of the storage cells for one column of a 32×16-bit sub-memory block; 16 columns required for one sub-memory block	103
Fig. 4.13	Circuit schematics of the Row Decoder and the Word Line Controller	104
Fig. 4.14	Circuit schematic of the Column Decoder	104
Fig. 4.15	Circuit schematic of the Transfer Gate	105
Fig. 4.16	Circuit schematic of the Input Buffer	105
Fig. 4.17	Circuit schematic of the completion detection circuit for the memory macrocell	106

Fig. 4.18	The async 128×16-bit memory macrocell (part of the FFT/IFFT prototype IC)	107
Fig. 5.1	Block diagram of the sync FFT/IFFT processor	112
Fig. 5.2	Data flows for each butterfly operation: (a) to compute the real outputs, and (b) to compute the imaginary outputs	113
Fig. 5.3	Pipeline sequences for each data flow	114
Fig. 5.4	The block diagram of the async FFT/IFFT processor	116
Fig. 5.5	The interface signals in the async FFT/IFFT processor	117
Fig. 5.6	Architecture of the control portion in the FFT/IFFT processor	120
Fig. 5.7	Architecture of the data paths in the async FFT/IFFT processor	121
Fig. 5.8	The simplified signal transition graph for loading new samples	122
Fig. 5.9	The simplified signal transition graph for loading data for butterfly operations	123
Fig. 5.10	The simplified signal transition graph for executing a butterfly operation	124
Fig. 5.11	The simplified signal transition graph for writing outputs into the memory	125
Fig. 5.12	A modeling timing diagram for the Async FFT/IFFT Controller. Note that the pertinent signals are only shown and the delays herein do not reflect the actual circuit delays	126
Fig. 5.13	The control-state signal transition flow (from the current state to the next state) in the Write Back Controller	127
Fig. 5.14	Block diagram of the Memory Data Path	129
Fig. 5.15	Block diagram of the Shifter Data Path	130
Fig. 5.16	Block diagram of the Multiplier Data Path	131
Fig. 5.17	Block diagram of the Adder Data Path	132
Fig. 5.18	Block diagram of the Write Back Data Path	134
Fig. 5.19	Microphotograph of the sync FFT/IFFT processor	135
Fig. 5.20	Microphotograph of the async FFT/IFFT processor	135
Fig. 5.21	The test jig for the async FFT/IFFT processor	136
Fig. 5.22	The real outputs obtained using the logic analyzer	137
Fig. 5.23	Energy dissipation for the async and sync (@ 1MHz) FFT/IFFT processors at 1.0V to 1.4V	139
Fig. 5.24	Delays for the async and sync FFT/IFFT processors	142

Fig. 5.25	Energy-delay curves of the async and sync FFT/IFFT processors	144
Fig. 5.26	The distribution of the number of transistors in various modules in the async FFT/IFFT processor; the total number of transistors is ~ 74K	145
Fig. 5.27	The energy breakdown (in %) of the async FFT/IFFT processor at 1.1V to 1.4V	146

List of Tables

TABLE 1.1	Specifications of the FFT/IFFT Processor	6
TABLE 2.1	Noise Margins for the Targeted Hearing Aid Application in the Chosen 0.35μm CMOS Process	21
TABLE 2.2	Timing Notations for the Sync Circuit	22
TABLE 2.3	Dual-rail Coding	32
TABLE 2.4	Comparisons between Sync and Async Designs (Normalized to a Fully Sync Design)	44
TABLE 3.1	Energy, Delay, Energy-delay-product (EDP) & IC Area of the Proposed Latch Adder Against that of Other Full Adders @ 1.1V, 1MHz	62
TABLE 3.2	Energy, Delay, Energy-delay-product (EDP) & IC Area of the Proposed Latch Adder Against that of Other Adders with Latches @ 1.1V, 1MHz	63
TABLE 3.3	Delay, Energy, Energy-delay-product (EDP) & IC Area of Various 16-bit Accumulators @ 1.1V, 1MHz Based on Post-layout Simulations and Measurements on Prototype ICs	66
TABLE 3.4	Delay (Best-case, Average-case, and Worst-case), Energy (E), Number of PTVs & IC Area of Various Adders @ 1.1V, 1MHz	74
TABLE 4.1	Average Number of Switchings, Energy, Delay, Area & Energy-delay-product (EDP) of Various Array Multipliers @ 1.1V, 1MHz based on Post-layout Simulations and Measurements on Prototype ICs	96
TABLE 4.2	Number of Trivial Multiplications and its % in 8- to 512-point Radix-2 FFT Algorithms	99
TABLE 4.3	The Energy and IC Area for the Control-Multiplier and the Multiplier Cores @ 1.1V, 1MHz	101
TABLE 4.4	The Characteristics of the Async 128×16-bit Memory Macrocell @ 1.1V, 1MHz	108
TABLE 5.1	The Top-level Signals and Functions of the Async FFT/IFFT Processor	118
TABLE 5.2	The Characteristics of the Sync and Async FFT/IFFT Processors	138

TABLE 5.3 Operating Conditions for the Sync and Async FFT/IFFT Processors 139
TABLE 5.4 Characteristics of Recent FFT Processors and the Proposed Async and Benchmarked Sync FFT/IFFT Processors 147

Symbols and Abbreviations

α	—	switching activity
τ_r	—	rise time/fall time
φ	—	truncated portion for the multiplier
Δf	—	frequency resolution
$\Sigma\Delta$	—	sigma-delta
C_{L}	—	load/output capacitance
E	—	energy
f	—	switching frequency
f_{sampling}	—	sampling frequency
I_a, I_b	—	imaginary input data for a radix-2 DIT butterfly
I'_a, I'_b	—	imaginary output data for a radix-2 DIT butterfly
I_{sc}	—	short-circuit current
k	—	transistor gain
L_{eff}	—	effective transistor length
N	—	number of points
NM_{H}	—	noise margin high
NM_{L}	—	noise margin low
P	—	power
P_{leakage}	—	leakage power
P_{sc}	—	short-circuit power
P_{static}	—	static power
$P_{\mathrm{switching}}$	—	switching (dynamic) power
T	—	time period
t_{clk1}, t_{clk2}	—	clock delay
t_{hold}	—	hold time for the register
t_{logic}	—	maximum delay of the logic circuits
$t_{\mathrm{logic,cd}}$	—	contamination delay of the logic circuits
t_{reg}	—	maximum delay of the registers
$t_{\mathrm{reg,cd}}$	—	contamination delay of the register
t_{setup}	—	setup time for the register
R_a, R_b	—	real input data for a radix-2 DIT butterfly
R'_a, R'_b	—	real output data for a radix-2 DIT butterfly
V_{DD}	—	supply voltage
V_{out}	—	output voltage
V_{T}	—	threshold voltage of the transistor
V_{Tn}	—	threshold voltage of the NMOS transistor
V_{Tp}	—	threshold voltage of the PMOS transistor

W_{bit}	—	wordlength
W_N^{nk}	—	twiddle factor
ACK	—	the acknowledge signal
ASIC	—	application-specific-integrated-circuit
Async	—	asynchronous-logic
BB	—	balanced Booth
BFP	—	block floating point
BG	—	bit growth
CCSA	—	carry completion sensing adder
CHP	—	communication hardware process
COMP	—	the completion signal
CRA	—	carry ripple adder
CSP	—	communication sequential processes
DA	—	dynamic adder
DCVS	—	differential cascode voltage swing
DFT	—	discrete Fourier transform
DI	—	delay-insensitive
DIF	—	decimation-in-frequency
DIT	—	decimation-in-time
DSP	—	digital signal processor
ECDL	—	enabled/disable CMOS differential logic
EDA	—	electronic design automation
EDP	—	energy-delay-product
EMI	—	electro-magnetic interference
Exp	—	exponent
FFT/IFFT	—	fast Fourier transform/inverse fast Fourier transform
HCC	—	handshaking control circuit
IC	—	integrated circuit
IP	—	intellectual property
IR	—	current-resistance
LA	—	Latch Adder
LSB	—	least significant bit
MSB	—	most significant bit
Mux	—	multiplexer
NORA	—	no race
NSP	—	N-typed single phase
NTU	—	Nanyang Technological University
PDP	—	power-delay-product
PPG	—	partial product generator

PTV	—	potential timing violation
QDI	—	quasi-delay-insensitive
REQ	—	the request signal
SB	—	standard Booth
SI	—	speed-independent
SoC	—	system-on-chip
SRAM	—	static random access memory
ST	—	self-timed
STG	—	signal transition graph
Sync	—	synchronous-logic
TG	—	transmission gate
VLSI	—	very large scale integration
W/L	—	width/length

Chapter 1

Introduction

1.1 Motivation

Hearing aids (hearing instruments) are assistive high efficacy biomedical devices to improve the speech intelligibility of hearing impaired users. The primary challenges in these subminiature devices are often power dissipation related [1]–[9] due to the need to realize a number of advanced signal processing algorithms, including a filter bank [10], [11], noise reduction [12], acoustic feedback cancellation [13], [14], low-distortion output amplification [15], [16], etc. Physically, the constraints of such devices are largely due to the low voltage (1.1V – 1.4V) low energy capacity (~ 100mA·h) miniature-size battery used, and to aesthetics (e.g. small size [17]). As the life span of the battery in typical hearing aids is usually expected to be ~ 100 hours or more, the current drawn (hence power dissipation) is expected to be ≤ 1mA @ 1.1V – 1.4V. In view of this tight power constraint, simplified versions of highly desirable complex signal processing algorithms (such as the real-time noise reduction [12]) are realized in practical hearing aids.

To satisfy the power constraint in the current-art power-critical mixed-signal hearing aids, low power (and energy) methodologies are routinely adopted to both the analog and digital circuits therein. The analog circuits include a high power-efficient Class-D amplifier [15] (~ 90% power-efficiency over a large range of

modulation index as opposed to the conventional Class-AB amplifier with ~ 30% power-efficiency) and a high signal-to-noise ratio sigma-delta (ΣΔ) modulator [7]. The digital circuits include a low voltage medium-to-low speed but energy-efficient application-specific-integrated-circuit (ASIC) digital signal processor (DSP) [1]–[3]. The DSP therein is usually highly optimized – from the system level down to the transistor level, including a small feature-size fabrication process – and employing many low power/energy design techniques [18]–[23].

The Fast Fourier Transform (FFT) [24], [25] is one of the most fundamental signal processing algorithms in audio signal processing and it serves to provide spectrum analysis (frequency resolution) of a given set of input signals (see Chapter 2 later). In some current-art hearing aids and advanced audio devices, the DSP therein embodies the FFT as one of its core macrocells. The computation of the FFT algorithm is relatively complex and essentially comprises multiplications and additions. For high frequency resolution analysis required in some advanced audio applications, the number of points (N) of the FFT algorithm is usually large (e.g. $N \geq 128$), resulting in a large number of multiplication and addition operations and correspondingly, a large number of memory accesses. In the overall hardware perspective, the FFT processor is a complex macrocell and is hence often the most power dissipative cell and that occupies a major portion of the integrated circuit (IC) real estate of the DSP.

In virtually all FFT realizations reported [26]–[34] to date, the synchronous-logic (sync) approach is the approach adopted. This sync approach is the same

approach adopted by the electronics community at large for the majority of complex digital circuit designs; the sync approach will be reviewed in Chapter 2. The sync approach is prevalent arguably due to the maturity of its design methodologies, design simplicity and the availability of many excellent commercial electronic design automation (EDA) tools [35]–[37] and established library cells. Despite the general acceptance of the sync design approach, power dissipation and speed are often difficult parameters to satisfy in ever increasing signal processing demands of advanced electronic devices. To alleviate these parameters, there has been considerable research and intensive on-going research effort on low power dissipation techniques [20]–[23] and since the 1990s, renewed research interest on the 'less accepted' asynchronous-logic (async) approach [38]–[46].

The async approach is an alternative (to sync approach) with potentially lower power dissipation and higher speed advantages largely due to the absence of a global clock infrastructure (required in the sync approach) and due to its inherent fine-grain gating operations (resulting in potentially reduced redundant operations and spurious switching/glitch). The async approach is essentially the design of circuits that are self-timed and does not require a global clock and the signaling therein is hence localized; the async approach will be reviewed in Chapter 2. Other potential merits of the async approach include adaptive voltage scaling [45], data-dependant operations [4], etc [38]–[44].

Despite these potential advantages, only a small number of async designs [38], [41]–[44] have been demonstrated to date. Arguably, the adoption of async approach remains stymied, to a large part, by their unavailability of advanced EDA tools (the async EDA tools are nascent when compared to sync EDA tools) and by the lack of established async library cells. Of the few async designs reported to date, some have indeed demonstrated superior (lower) power dissipation and higher speed attributes [4], [38], [41]–[44], [47]. However, many async designs have not shown to have improved attributes (over sync designs) [38], [46], [48]–[52] largely due to the 'unsophisticated' approach taken. This 'unsophisticated' approach simply involves using standard sync circuits (using synchronous-based EDA tools and standard library digital cells) and modifying their input/output interfaces to satisfy the async signaling protocol. In view of this simple modification of sync circuits to make them asynchronous, such async designs are expected to be inefficient, resulting in high circuit overhead, hence higher power dissipation and slower speed.

The research work reported in this thesis is part of a larger Nanyang Technological University (NTU) research effort on async design, including the development of sophisticated async EDA tools (async design compiler [53]), async library cells, and the integration of the EDA tools and a cell library. The specific research effort in this PhD program delineated in this thesis pertains to an exploitation of the low power/energy attributes of the async approach to realize a relatively complex digital signal processing circuit (> 50K transistors) – an async FFT/inverse FFT (FFT/IFFT) processor for advanced audio devices including hearing aids. This research exercise involves the development of async library

cells with specific emphases on low voltage (1.1V – 1.4V) and power-/energy-critical operations, and the design and handcrafting layout (fully and partially) of the async FFT/IFFT processor. This FFT/IFFT processor herein also serves as a benchmark for comparison when a later async FFT/IFFT processor is designed using the async tools (and the async library cells) that are currently developed by other members of the NTU async research team. For sake of clarity, note that in this thesis, two FFT/IFFT processors are designed, one sync and the other async. These are compared to determine the attributes of the sync and the async approaches.

1.2 Objectives

The overall objective of this thesis is to demonstrate the low power low energy attributes of the async approach (over the prevalent sync approach) in the design and realization of a low voltage low energy FFT/IFFT processor. The specifications of the FFT/IFFT processor are tabulated in Table 1.1.

TABLE 1.1 SPECIFICATIONS OF THE FFT/IFFT PROCESSOR

	Specifications
Wordlength	16-bit
Number of points	128-point
Operating voltage	1.1V – 1.4V
Energy dissipation	< 350nJ per transform
Speed performance	< 2.0ms per transform
IC Area	< 2mm^2 @ 0.35μm CMOS process

In view of the overall objective, the specific objectives are:

(i) To design basic rudimentary low energy async microcells, including novel adders, an accumulator and a latch controller;

(ii) To design specific macrocells, including multipliers and a memory; and

(iii) To design an async FFT/IFFT processor (in part embodying the proposed microcells and macrocells). It is an objective to demonstrate that by means of the async approach, the async FFT/IFFT processor is more energy-efficient than its sync counterpart, specifically for low voltage low-to-medium speed (< 5MHz) energy-critical applications including hearing aids.

1.3 Contributions

The contributions of the work presented in this thesis pertain to circuit designs and async design techniques, and the design of several microcells and macrocells and

an async FFT/IFFT processor, all with emphases on low voltage operation and low energy dissipation. Of the microcells designed, the Latch Adder (LA), Latch Accumulator, *Type-*γadder and Broad B latch controller arguably feature the most significant novelty. The async macrocells designed are a multiplier (embodying a novel multiplier core) and a memory. The async FFT/IFFT processor, in part, comprises these microcells and macrocells. The contributions in these proposed designs will now be described.

(i) Proposed microcell designs

(a) Latch Adder (LA)

The proposed LA is a latch-cum-adder design (an integrated latch and adder) whose novelty is the synchronization (in time) of its adder inputs (with different arrival times) by means of the integrated latch, thereby reducing spurious switching (see Chapters 3 and 4). Its primary attributes are lower energy dissipation, shorter delay and smaller IC area than the conventional adder with latches, which would otherwise comprise an independent adder and three independent latches.

(b) Latch Accumulator

The proposed Latch Accumulator is an accumulator comprising an LA as its core and an added latch. Its novelty is that the integrated LA can latch the output, hence only an added latch is required whilst the conventional accumulator would otherwise comprise an independent adder and two independent latches. Consequently, its primary attributes are slightly shorter

delay, lower energy dissipation and smaller IC area than the conventional accumulator.

(c) *Type-*γ adder

The proposed *Type-*γ adder is an async 2-bit carry completion sensing adder (CCSA) whose novelties are the employment of a transistor sharing technique in its Sum and Carry blocks, and an insertion of delay balancing transistors in its Carry block. It features comparable (slightly higher) energy dissipation to the lowest energy dissipation CCSA, the *Type-*β CCSA [54], [55], but is more robust (against delay variations) than the latter. The *Type-*γ CCSAs can be augmented with the author's earlier disclosed [54] less robust *Type-*α CCSAs (see Chapter 3) to construct a longer wordlength adder. This augmentation results in shorter delay and lower energy dissipation (than an adder design comprising only *Type-*γ CCSAs) without compromising its delay robustness.

(d) Broad B latch controller

The proposed Broad B latch controller is an async latch controller in the broad data valid scheme (see Chapter 2). Its primary novelty is its assurance that the input signals are always successfully latched into the latches before it triggers the successive functional modules, thereby eliminating the possibility of malfunction of the functional modules. Unlike the proposed Broad B latch controller where there is no need to account for any delays to obtain this assurance, other conventional latch controllers may otherwise suffer from

malfunction unless sufficient delay is carefully inserted to accommodate the delays to properly trigger the successive functional modules.

(ii) Proposed macrocell designs

(a) 16×16-bit Booth array-based multiplier core

The proposed 16×16-bit Booth array-based multiplier core is a multiplier design based on the Booth algorithm and whose structure is the array structure. The primary novelties are the embodiment of the abovementioned proposed LAs and timing controls (by means of proposed delay line circuits), thereby significantly reducing the spurious switching (~ 62%) and hence, energy dissipation (~ 32%). A Singapore patent and a USA patent embodying the reduced spurious switching technique have been granted.

(b) Control-Multiplier

The proposed Control-Multiplier is an async 16×16-bit multiplier design and its primary novelties are added control circuits to specially compute trivial multiplications (× 1, × −1, and × 0). The non-trivial multiplications are, as usual, computed by its multiplier core, the abovementioned Booth array-based multiplier core. The primary attribute of the Control-Multiplier is the further reduction in energy dissipation (22% reduction compared to the proposed 16×16 multiplier core) by taking advantage of the fact that > 40% of the multiplications are trivial in the computation of the 128-point FFT algorithm.

(c) 128×16-bit memory design

The proposed 128×16-bit memory macrocell is a single-port async SRAM where several well-established low energy techniques including the partition of the memory into several sub-memory blocks are adopted. The primary novelty of the memory macrocell is the embodiment of a proposed Word Line Controller to activate the particular sub-memory block of interest and this prevents multiple assertions of unselected sub-memory blocks. This results in reduced energy dissipation.

(iii)Proposed async FFT/IFFT processor design

The proposed async FFT/IFFT processor is a dedicated audio processor to compute 128-point radix-2 decimation-in-time (DIT) FFT/IFFT algorithm. The async FFT/IFFT processor embodies established and proposed async microcells, and proposed macrocells. The primary novelty is its extremely low energy attribute (as benchmarked against reported designs to date). The slight drawbacks are a slightly larger IC area and an inconsequential longer delay (inconsequential due to the intended low speed hearing aid application).

The above contributions will be delineated in Chapters 3 – 5 and have been published in literature [56], [61]–[65] or accepted for publication [57], [58]. The summary of these published/accepted papers is delineated in pages 156 and 157 of this thesis.

1.4 Organization

The remainder of this thesis is organized as follows.

Chapter 2 reviews the literature of the work presented in this thesis and three topics are reviewed. The first topic is a review of the mechanisms of power (energy) dissipation in CMOS circuits, low power low energy design methodologies, and noise margin analysis. The second topic is a review of sync and async design approaches. The third topic is an overview of the FFT/IFFT algorithm and this provides a preamble to the architecture adopted for the sync and async realizations.

Chapter 3 describes the design of several proposed microcells, including a Latch Adder, a Latch Accumulator, a *Type-*γ adder and a Broad B latch controller. These microcells serve as basic building blocks for more complex designs, including the designs delineated in Chapters 4 and 5.

Chapter 4 describes the two major async macrocells embodied in the proposed async FFT/IFFT processor. The first async macrocell is an async multiplier design, embodying a novel Booth array-based multiplier core and added control circuits, specifically for the FFT algorithm. The second async macrocell is the proposed async memory design and the techniques to obtain a low energy memory design are presented.

Chapter 5 describes the design and implementation of the sync and the proposed async FFT/IFFT processors. The proposed async FFT/IFFT processor is thereafter benchmarked against the sync FFT/IFFT processor and against other reported designs to depict its attributes.

Chapter 6 concludes this thesis and provides recommendations for future research.

Chapter 2

Literature Review

The research work described in this thesis pertains to the design and realization of async circuits, ranging from microcells and macrocells and to a full FFT/IFFT processor. This chapter will review the literature pertaining to the research work herein. First, a review of low power low energy design techniques and noise margin analysis will be presented. An overview of the sync and async design approaches is thereafter reviewed followed by a review of the FFT/IFFT algorithm. Finally, a summary will be given.

2.1 Low Power Low Energy Design Techniques

Power (P) and energy (E) [20], [21] are two related figure-of-merits (see (2.1)) often used to qualify the power/energy parameter of digital circuits. This section reviews the mechanisms of power dissipation, discusses low power/energy design methodologies and provides noise margin analysis for the targeted application.

$$P = \frac{\partial E}{\partial t} \tag{2.1}$$

2.1.1 Mechanisms of Power Dissipation

In digital CMOS circuits, there are four mechanisms of power dissipation [20], [21], [66] – the switching (dynamic) power ($P_{\text{switching}}$), short-circuit power (P_{sc}),

leakage power (P_{leakage}), and static power (P_{static}). These mechanisms are well established and well accepted.

Switching power ($P_{\text{switching}}$) is dissipated during transistor switching events, when either charging the load/output capacitance C_{L} to V_{DD} or discharging the load/output capacitance C_{L} to ground, as depicted in Fig. 2.1. The switching power can be expressed analytically [20], [66] as:

$$P_{\text{switching}} = \alpha \cdot \frac{1}{T}\left[\int_0^{T/2} V_{\text{out}}(-C_{\text{L}} \cdot \frac{\partial V_{\text{out}}}{\partial t})\partial t + \int_{T/2}^{T} (V_{\text{DD}} - V_{\text{out}})(C_{\text{L}} \cdot \frac{\partial V_{\text{out}}}{\partial t})\partial t\right] \quad (2.2a)$$

$$= \alpha \cdot f \cdot C_{\text{L}} \cdot {V_{\text{DD}}}^2 \quad (2.2b)$$

where α is the switching activity,

C_{L} is the load/output capacitance,

V_{out} is the output voltage,

V_{DD} is the supply voltage, and

$f = \frac{1}{T}$, the switching frequency.

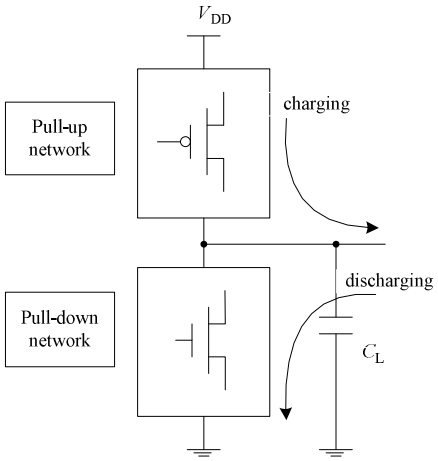

Fig. 2.1 Charging and discharging of a CMOS circuit during switching

For a well designed CMOS circuit based on non-deep-submicron processes, the switching power dissipation is usually dominant (e.g. 80% ~ 90% of the total power dissipation) [20]. However in deep-submicron processes [67], [68], the leakage power dissipation (see later) becomes increasingly significant. From (2.2b), it is apparent that the switching power dissipation can be reduced by reducing the switching activity, load/output capacitance, supply voltage, and the switching frequency.

Switching power dissipation in (2.2a) and (2.2b) can be re-expressed [20], [66] as switching energy required for operations within time T as:

$$E_{\text{switching}} = \alpha \cdot \left[\int_0^{T/2} V_{\text{out}} (-C_{\text{L}} \cdot \frac{\partial V_{\text{out}}}{\partial t}) \partial t + \int_{T/2}^{T} (V_{\text{DD}} - V_{\text{out}}) (C_{\text{L}} \cdot \frac{\partial V_{\text{out}}}{\partial t}) \partial t \right] \quad (2.3a)$$

$$= \alpha \cdot C_{\text{L}} \cdot {V_{\text{DD}}}^2 \quad (2.3b)$$

Short-circuit power (P_{sc}) arises due to the short-circuit current, I_{sc}, when both the PMOS and NMOS transistors are simultaneously turned on. During this short transient interval, I_{sc} conducts from V_{DD} to ground. The short-circuit power dissipation can be expressed [20], [66] as:

$$P_{\text{sc}} = I_{\text{sc}} \cdot V_{\text{DD}} \quad (2.4a)$$

$$\approx \frac{1}{12} \cdot k \cdot \tau_r \cdot f \cdot (V_{\text{DD}} - 2V_{\text{T}})^3 \quad (2.4b)$$

where k is the transistor gain,

τ_r is the rise time (fall time) of the input, and

V_T is the threshold voltage of the transistors (assuming that $|V_{Tp}| = V_{Tn}$).

From (2.4b), it is apparent that the short-circuit power dissipation can be minimized (almost negligible) when the supply voltage (V_{DD}) is close to $2V_T$ or less (i.e. no I_{sc}). Other factors such as the transistor gain, rise time, and switching frequency also influence the short-circuit power dissipation.

Leakage power ($P_{leakage}$) arises when the quiescent (steady state) current flows from V_{DD} to ground even when the transistors do not conduct. The leakage power components [67], [68] include sub-threshold leakage, gate leakage, reversed-biased junction band-to-band-tunneling leakage, and others. For non-deep-submicron CMOS processes (e.g. feature size > 0.18μm), the leakage currents are typically in the order of nano amperes and can usually be neglected. However, if the transistors are of very short channel or nano-scaled lengths, the leakage current becomes significant (relative to switching power). The phenomena of leakage power dissipation [68] are well established and techniques to reduce leakage current are also well investigated and some techniques well established.

Static power (P_{static}) is dissipated only when a continuous current (not short-circuit current during the short transient interval) is established between V_{DD} and ground. In contemporary static CMOS digital circuits (constructed by pull-up and pull-down networks), there is ideally zero static power dissipation. Other circuits such as ratioed logics [20] and analog circuits dissipate static power because a continuous current is needed to provide proper operation, usually for biasing conditions.

For the purpose of comparing different designs, a simplistic comparison based strictly on the power parameter may be a misleading metric if the performance or operating conditions of the circuits are not properly specified. This is because power is a time-averaged metric, and the reading of power dissipation can be increased or decreased by changing the time. In order to fairly compare the power dissipation of different circuits, the circuits have to be quantified for an equivalent (or similar) operation subject to delay constraints. In this respect, power and energy are similar. For a fair comparison, the other parameters such as process technology, V_{DD} (if voltage is not used as a means for power/energy reduction) and other parameters have to be considered and accounted for accordingly.

In view of the above considerations, for the purpose of benchmarking the designs proposed in this thesis against reported designs, energy dissipation (as opposed to power dissipation) is used as a figure-of-merit. In most cases, the proposed designs are constructed and reported designs reconstructed for the same operations, fabricated using the same process technology, and simulated and measured under the same V_{DD} and timing constraints. The μW/MHz (equivalent to μW per million operations per second) or J (Joule) unit measure is used for qualifying the microcells and macrocells, and J unit measure is used for the FFT/IFFT processors (for one complete FFT/IFFT computation).

For completeness, the delay (computation time), energy-delay-product (EDP) [20], [69] and IC area are also used as figure-of-merits. The other prevalent figure-of-merit sometimes used in literature, power-delay-product (PDP) [20], however, is

not considered in this thesis. This is because PDP is usually interpreted as the average energy per switching event [20]. In this respect, the energy dissipation adopted is somewhat similar to PDP but the former is arguably more generic as it qualifies the circuits for a given operation that may comprise many switchings.

2.1.2 Low Power/Energy Design Methodologies

The design methodologies for low power/energy dissipation span all levels of design, from the high level algorithm, architecture, circuit/logic down to the low level device/process parameters [18]–[23] as shown in Fig. 2.2.

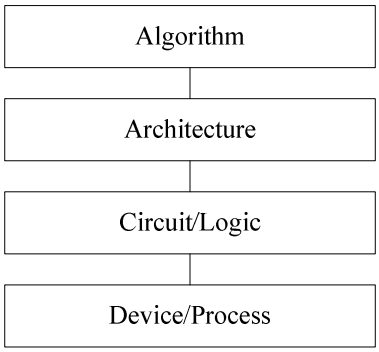

Fig. 2.2 Power/energy methodology hierarchy

The algorithmic-level low power/energy design methodology generally includes a selection of data processing algorithms, specifically to minimize the computational complexity and switching events for a given task. For example, in this thesis, the FFT algorithm [25] is selected over the Discrete Fourier Transform (DFT) [24], and in other designs, considerations include algorithm transformations to match computational resources [21], etc. The architecture-level low power/energy design methodology generally includes smart power

management of the various system blocks [66], utilization of pipelining and parallelism [66], resource sharing [70], etc. The circuit-/logic-level low power/energy design methodology generally includes the proper choice of logic family (e.g. complementary static CMOS vs pass-logic vs dynamic logic [20], [70], [71]), synchronization (e.g. synchronous-logic vs asynchronous-logic [20], [38], [70]), glitch (spurious switching) reduction [70], operand isolation [72], etc. The device-/process-level low power/energy design methodology is often process technology related, including the threshold voltage, device geometries, interconnect properties, etc [21]. In short, a low power/energy design requires concurrent optimization at all levels, taking into consideration the interdependencies among these levels [23].

Many low power/energy techniques (including some of the abovementioned low power/energy methodologies) are employed in the design of the proposed microcells, macrocells and the async FFT/IFFT processor (in part embodying the microcells and macrocells). These techniques include low supply voltage (V_{DD} = 1.1V – 1.4V), low leakage CMOS process, spurious switching reduction, async approach, FFT/IFFT algorithm, low speed and energy efficient architecture, low power standard library cells, etc. Similarly, the same low power/energy techniques (including the V_{DD}, process technology, and the same low power standard library cells adopted in the async design) and clock-related low power/energy techniques (see later) are also applied to the sync FFT/IFFT processor (that serves as a benchmark for comparison).

It can be argued that a circuit operated @ V_{DD} = 1.1V – 1.4V is no longer considered a low voltage design because some current-art designs operate at ≤ 1V (e.g. circuits using 90nm CMOS process) and some designs even operate in the sub-threshold region (e.g. 0.3V) [9], [34]. However, due to cost and availability of fabrication process, the chosen CMOS process for this project is 0.35μm CMOS and the nominal V_{DD} = 3.3V. The V_{DD} of 1.1V – 1.4V, in the context of this fabrication process, is considered low voltage, and this voltage range is the typical low voltage operating range for hearing aid devices.

2.1.3 Noise Margin

It is established that when V_{DD} is low (for low voltage applications), the noise margin is somewhat degraded. Noise margin is a well-recognized problem and its analysis is well-established [20], and for simplicity, the noise margin low (NM_L) and the noise margin high (NM_H) are simplified as follows [20]:

$$NM_L = V_{IL} \geq V_{Tn} \tag{2.5a}$$

$$NM_H = V_{DD} - V_{IH} \approx V_{DD} - |V_{Tp}| \tag{2.5b}$$

where V_{IL} is input voltage low and V_{IH} is input voltage high.

In this thesis, since V_{DD} = 1.1V to 1.4V, V_{Tn} ≈ 0.5V and $|V_{Tp}|$ ≈ 0.7V for the chosen 0.35μm CMOS process, the noise margins for the targeted hearing aid application can be determined. These are tabulated in Table 2.1.

TABLE 2.1 NOISE MARGINS FOR THE TARGETED HEARING AID APPLICATION IN THE CHOSEN 0.35μM CMOS PROCESS

V_{DD}	NM_L*	NM_H	Minimum noise margin
1.4V	0.5V	0.7V	0.5V
1.3V	0.5V	0.6V	0.5V
1.2V	0.5V	0.5V	0.5V
1.1V	0.5V	0.4V	0.4V

* Worst-case scenario

It is remarked that for the targeted application, the maximum tolerable noise (e.g. due to noise in the PCB, voltage variation, ambient noise, etc.) is < 140mV (10% of 1.4V). On the basis of Table 2.1, the minimum noise margin allowable is sufficient for the targeted application.

2.2 Synchronous-Logic and Asynchronous-Logic Design Approaches

This section will first review the sync design approach and then the async design approach. The sync and async design approaches are well-established in literature [20], [38].

2.2.1 Synchronous-Logic Design Approach

The sync approach uses a global clock (global synchronization) to synchronize digital circuits [20]. Fig. 2.3 depicts a basic structure of sync pipeline circuit and Table 2.2 tabulates timing notations.

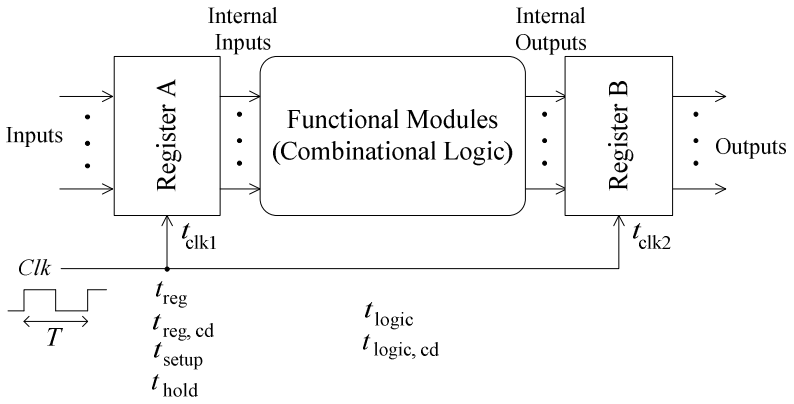

Fig. 2.3 A sync circuit

TABLE 2.2 TIMING NOTATIONS FOR THE SYNC CIRCUIT

Symbol	Notation
t_{logic}	maximum delay of the logic circuits
$t_{logic,cd}$	contamination delay (minimum delay) of the logic circuits
t_{reg}	maximum delay of the registers
$t_{reg,cd}$	contamination delay (minimum delay) of the registers
t_{setup}	setup time for the register
t_{hold}	hold time for the register
t_{clk1}	clock delay to the register A
t_{clk2}	clock delay to the register B
T	clock period

Under the ideal condition when $t_{clk1} = t_{clk2}$, the proper operations of the sync circuits are constrained by the following expressions [20].

$$T \geq t_{reg} + t_{logic} + t_{setup} \tag{2.6}$$

$$t_{reg,cd} + t_{logic,cd} \geq t_{hold} \tag{2.7}$$

Equation (2.6) indicates that the clock period must be long enough for the data to propagate through the registers and logic (including all wire delays) and to be setup at the input registers before the next rising edge of the clock [20]. Equation (2.7) indicates that the hold time of the input registers must be shorter than the sum of the minimum delays of the logic circuits and registers [20]. These restrictions make the sync circuits easy to understand and design. This, in part, explains why most digital circuits are synchronous based.

However, in real-life, the clock signal is never ideal. As a result, two additional clock-related parameters, namely clock skew and jitter, need to be considered in the sync system. Clock skew is referred to as the spatial variation in arrival time of a clock transition on a circuit, and is caused by static mismatches in the clock paths and differences in the clock loads [20]. The clock skew makes the circuit more susceptible to race conditions, which may harm the correct operation of the system. In other words, the design of a low skew clock network is essential. The low skew clock network, however, usually requires a large number of clock buffers to balance the loads at different blocks and may result in higher power dissipation.

Clock jitter refers to the temporal variation of the clock period at a point on the circuit and the clock period may be reduced or expanded on a cycle-to-cycle basis [20] largely due to the process, temperature, and voltage variations. The clock jitter degrades the circuit performance (speed), and to some extent, it makes the inputs (to the combinational circuits) be less synchronous, resulting in spurious switching – hence, increased power/energy dissipation.

Many sophisticated sync EDA tools have been successfully developed to resolve these clock-related issues. For a design running at up to few hundred MHz clock frequency, commercial sync EDA tools (such as tools from Synopsys [36] and Cadence [35]) can easily accommodate these issues. However, for a design running at a very high frequency (e.g. > 1GHz), clock-related issues remain formidable.

To reduce the power/energy dissipation due to the clock in a sync design, two approaches, clock gating and multiple-clock, are widely used. The clock gating approach reduces the power/energy dissipation of the circuit when computation is not required for an extended period of time (idle state). It is a relatively simple concept, as shown in Fig. 2.4. If the functional modules (not shown) do not compute any computations, the *Enable* signal is disabled. As a result, the registers (and the associated functional modules) do not dissipate unnecessary switching power/energy.

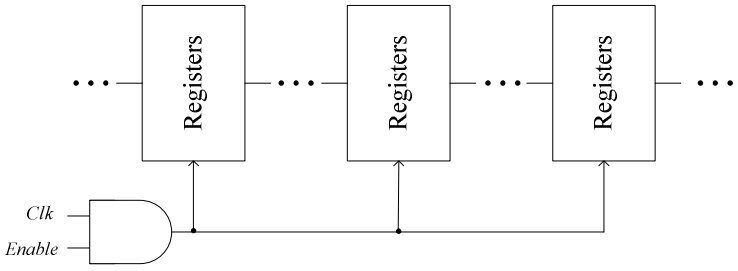

Fig. 2.4 Clock gating approach

The multiple-clock approach can reduce the power/energy dissipation of the registers, especially if some of them need not to be clocked at a high frequency clock rate. The clock signals involve multiple sub-clock signals that are

synchronized by the master clock. Consider an example when there are 500 registers in a system and they are clocked @ master clock of 2MHz. There will be 1 billion register switching activities per second in the system. However, if 400 of them are clocked @ sub-clock of 1MHz (assuming the functionality is not affected), the register switching activities per second therein will be reduced from 1 billion to 0.6 billion, a significant improvement.

Although the abovementioned techniques to reduce power/energy dissipation are effective, the degree of reduction is somewhat constrained by the global clock infrastructure. The clock gating approach, to a large extent, can only be used in a course-grain manner, that is, only a portion of the whole system can be gated. The efficiency of the multiple-clock approach, to a large extent, strongly depends on the pipeline structure and signal flow adopted. However, these two techniques tend to create more skew problems, and in turn complicate the design of the clock infrastructure.

In Section 5.2 later, the sync FFT/IFFT processor that serves as a benchmarked design employs these two techniques for low power/energy dissipation.

2.2.2 Asynchronous-Logic Design Approach

The research work herein pertains to async designs with general objective of low voltage operation and low energy dissipation. This section serves to review four relevant topics, namely (i) fundamental concepts of async designs, (ii) async EDA tools, (iii) basic async microcells, and (iv) potential advantages of the async

approach, and this review preambles the work presented in the subsequent chapters in this thesis.

2.2.2.1 Fundamental Concepts of Asynchronous-Logic Design

In this subsection, four async fundamental concepts will be introduced, namely (i) delay models, (ii) handshaking protocols and channels, (iii) data encoding, and (iv) async pipelines.

(i) Delay Models

Depending on the timing assumptions made [38]–[40], async circuits can be classified by the delay models. The well accepted models are the delay-insensitive (DI), speed-independent (SI), and self-timed (ST) models.

A DI circuit is an async circuit whose functionality is unaffected by the gate delays and the interconnect wire delays. The assumptions here are that both the gate and wire delays are unbounded and arbitrary. The DI circuits are extremely robust for their delay insensitive property. However, these circuits are difficult to realize [73]. Another variation, called quasi-delay-insensitive (QDI), is also delay-insensitive if 'isochronic forks' is accounted for and is accommodated in the design. An isochronic fork is a forked wire where all the branches have the same delay.

An SI circuit is an async circuit whose functionality is unaffected by the delays in its components (e.g. gates). The assumptions here are that the gate delays are unbounded and arbitrary, and that the wire delays are zero. It is clear that the assumption that the ideal zero wire delay is unrealistic (with respect to gate delays), particularly where the relative wire delay is significant in deep-submicron processes. To accommodate this, the SI model has been extended with further assumptions [38] by lumping wire delays into the gate delays.

An ST circuit is an async circuit whose correct operation relies on engineering timing assumptions [38]–[40]. Of the engineering timing assumptions, the most prevalent assumption is arguably the bounded delay model. The bounded delay model is similar to the model used in sync circuits [38], [39]. In this model, the circuit assumes that both the gate and wire delays are known, or at least bounded. Hence, the completion signal (*COMP*) is designed to be generated using a delay line matched to the delay of the gates (including the wire delays), and this completion signal indicates the availability of the outputs of the circuits. Such 'matched-delay' async circuits are usually much simpler than the DI and SI circuits. However, the drawbacks are its worst-case operation and sensitivity to variations (e.g. voltage, temperature, fabrication process, and parasitic effects). For correct operation, the matched-delay components are designed to exceed the worst-case delay of the gates, usually a 5% – 30% safety margin for aggressive designs and $> 100\%$ safety margin for less aggressive designs [74], [75].

In this thesis, a hybrid approach, by both using DI and ST circuits, is adopted for the design of the async macrocells (Chapter 4) and async FFT/IFFT processor

(Chapter 5). This hybrid approach is currently well-accepted and is adopted in many designs [4], [38].

(ii) Handshaking Protocols and Channels

Async circuits operate according to a set of handshaking signals (the request (*REQ*) and acknowledge (*ACK*) signals) [38]–[40], [76]–[78]. In general, they can be classified into two handshaking protocols (two-phase and four-phase) as depicted in Fig. 2.5, and four different channels (nonput, push, pull, and biput) as depicted in Fig. 2.6.

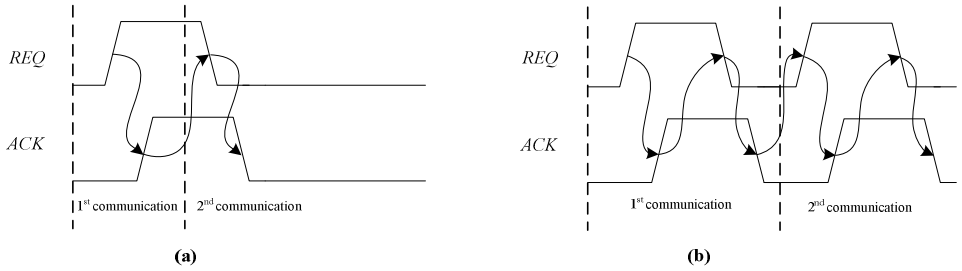

Fig. 2.5 (a) Two-phase handshaking protocol, and (b) four-phase handshaking protocol

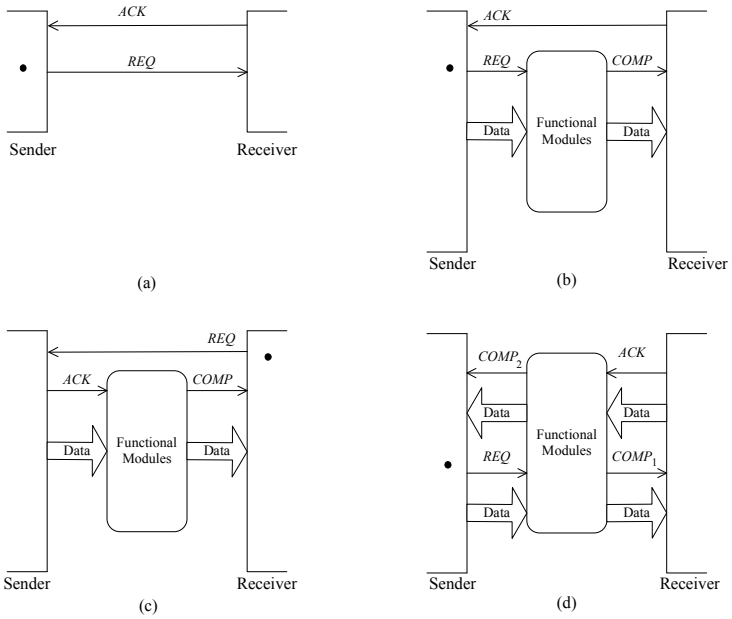

Fig. 2.6 Four channel types: (a) nonput, (b) push, (c) pull, and (d) biput

Figs. 2.5 (a) and (b) depict the two-phase and four-phase handshaking protocols respectively. The two-phase handshaking protocol is an edge-sensitive communication – any transition (rising or falling) on *REQ* or *ACK* is considered a valid communication. One complete asynchronous communication requires one transition on *REQ* and one corresponding transition on *ACK*. The four-phase handshaking protocol, on the other hand, is a level-sensitive communication. One complete asynchronous communication requires the assertion and de-assertion on both the *REQ* and *ACK* signals.

In Fig. 2.6, there are two parties called the sender and receiver, and the black dot represents the active party that initials the *REQ* signal. Either the sender or receiver can be the active party. For the nonput channel type, no data is exchanged between the sender and receiver. For the push channel type, the sender pushes the data (by the *REQ* signal) to the receiver (via functional modules,

if any), and the receiver sends the *ACK* signal once it receives the data. For the pull channel type, the receiver pulls the data from the sender (via functional modules, if any), and the sender will acknowledge the data. For the biput channel type, the sender sends the *REQ* signal together with the data (via functional modules, if any) to the receiver, and the receiver sends the *ACK* signal together with another data (via functional modules, if any) to the sender.

The two-phase and four-phase handshaking protocols can be better interpreted depending on the assumptions of the data valid schemes (and the channel types) [38]. Figs. 2.7 (a) and (b) depict the data valid schemes for a push channel in two-phase and four-phase protocols respectively. In the two-phase handshaking protocol, when *REQ* is toggled, the data is available and remains valid until *ACK* is toggled to indicate the completion of the operation. In the four-phase handshaking protocol, there are four possible data valid schemes, namely early, broad, late, and broadish (extended early). In the early data valid scheme, the data is assumed to be valid when *REQ* is asserted until *ACK* is asserted. In the broad data valid scheme, the data is assumed to be valid when *REQ* is asserted until *ACK* is de-asserted. In the late data valid scheme, the data is assumed to be valid when *REQ* is de-asserted until *ACK* is de-asserted. In the broadish data valid scheme, the data is assumed valid when *REQ* is asserted until *REQ* is de-asserted. The data valid schemes for other channels (pull and biput) can be constructed accordingly and are well established [38].

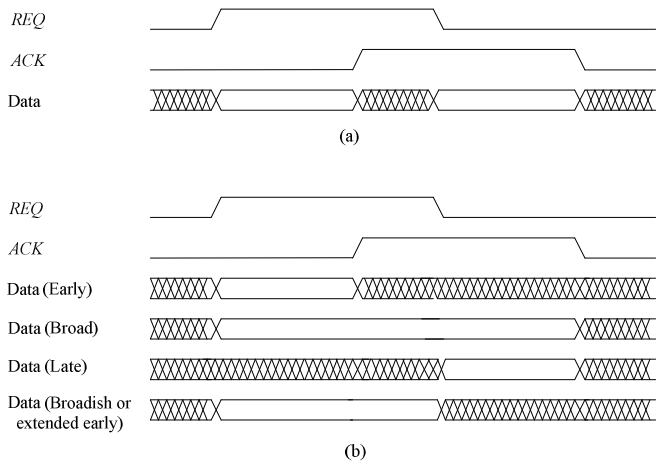

Fig. 2.7 Push channel data valid schemes for (a) two-phase handshaking protocol, and (b) four-phase handshaking protocol

In this thesis, the four-phase handshaking protocol is adopted for the async macrocells and for the async FFT/IFFT processor due to its prevalence and due to its lower power attribute (over the two-phase protocol) [38], [51]. For simplicity and ease of testing, the push channel with broad and broadish data valid schemes is applied to the async macrocells and to the async FFT/IFFT processor.

(iii) Data Encoding

Async circuits are designed to detect the arrival time of data, that is, the data must be encoded with some timing information. For data encoding in async circuits, bundled data and dual-rail data [38] are commonly used.

The bundled data coding uses a bundle of data wires (each wire represents one-bit information) with two control wires, *REQ* and *COMP*, that indicate the timing

information. The bundled data coding is applicable to the bounded delay model discussed earlier.

The dual-rail coding uses two wires for one-bit information. One of the wires is dedicated as a 'TRUE' wire while the other one is dedicated as a 'FALSE' wire. Table 2.3 depicts the truth table for the dual-rail coding technique. Both wires cannot be '1' at the same time. Initially, both wires are '0' (no operation). Once an operation is asserted, only one of the wires can be '1', that is an event. The opposite logic states of the signals not only serve as logic information, but also detect the timing information simultaneously. The dual-rail coding technique is robust, and the circuits employing this technique can be delay-insensitive. However, this technique requires complementary signals which may increase the switching activity, and would require a larger IC area.

TABLE 2.3 DUAL-RAIL CODING

	TRUE wire	FALSE wire
No operation	0	0
Logic '1'	1	0
Logic '0'	0	1
Not used	1	1

In this thesis, both the bundled data and dual-rail coding are adopted for the async FFT/IFFT processor. This hybrid approach is also well-accepted and adopted in many designs [4], [38].

(iv) Asynchronous-Logic Pipelines

Fig. 2.8 depicts a conventional four-phase bundled data pipeline [38]. For simplicity and as an illustration, the circuit in Fig. 2.8 is the pipeline without data processing and functional modules (not shown) can be inserted between the latches in the consecutive stages for data processing.

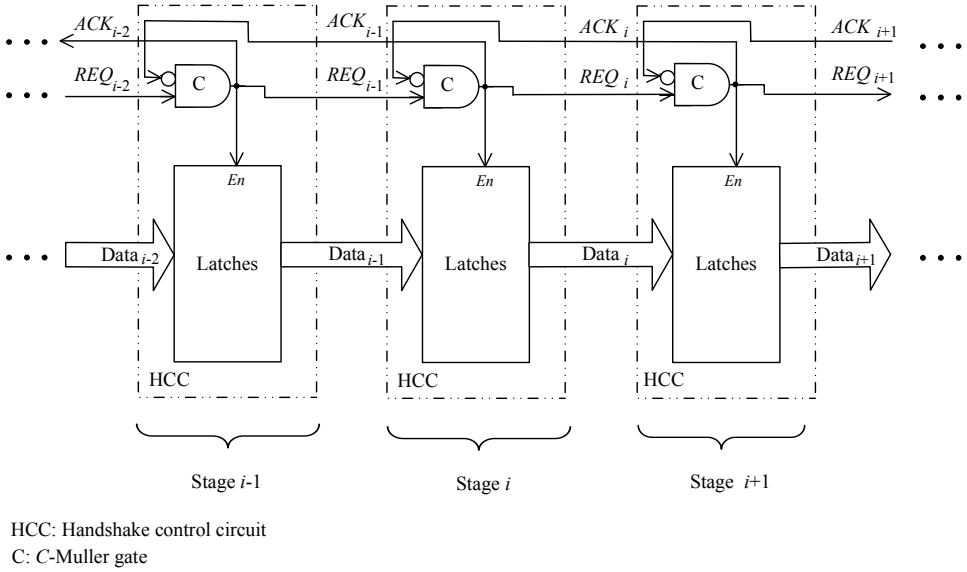

Fig. 2.8 A simple four-phase bundled data pipeline

In the four-phase handshaking protocol, either normally-closed or normally-opened latches [79], [80] can be used and the specific type affects the energy dissipation. Assuming that the initial handshaking signals are '0' and the normally-closed latches are used, when REQ_{i-2} is asserted, REQ_{i-1} (=ACK_{i-2}) will be asserted to open the latches in Stage i-1, and $Data_{i-2}$ is passed to be $Data_{i-1}$. The ACK_{i-2} signal will acknowledge the receipt of the data to the preceding stage. The REQ_{i-1} signal will trigger the C-Muller gate in Stage i and REQ_i (=ACK_{i-1}) is

asserted to open the latches in Stage i, and $Data_{i-1}$ is passed to be $Data_i$. The ACK_{i-1} signal will acknowledge the receipt of the data for Stage i-1, and REQ_{i-2} can be de-asserted, and the latches in Stage i-1 are then closed. $Data_{i-2}$ is now ready to be updated with new data. Thereafter, REQ_{i-1} (=ACK_{i-2}) is de-asserted to close the latches in Stage i, and $Data_{i-1}$ is ready to be updated (if only when ACK_i is asserted). At this time, a new operation can begin by asserting REQ_{i-2}. This scenario applies to the subsequent stages. Pipeline circuits embodying normally-opened latches can similarly be analyzed. The normally-closed latch approach is usually applied for reducing spurious switching (for low power/energy reduction) while the normally-opened latch approach, on the other hand, is usually applied for high speed operations [38], [79].

The two-phase bundled data pipeline is somewhat similar to the four-phase bundled data pipeline [38]. The pipeline structure in Fig. 2.8 is simple and features low throughput. More optimized and elaborate circuit implementations have been reported in literature [38].

In this thesis, for simplicity, testability, and low energy dissipation, the four-phase pipeline (with the broad and broadish latch controllers) and the normally-closed latch approach are adopted for the async FFT/IFFT processor design.

2.2.2.2 Asynchronous-Logic EDA Tools

Async circuits are highly ‘concurrent’ and the communication between modules is based on ‘handshake channels’. The concurrency and support of the handshake

channels are the two main parameters of async EDA tools. The Communication Sequential Processes (CSP) language has been developed by Hoare [38] to meet these async parameters, and virtually all work on the high-level modeling and synthesis of async circuits relate to the CSP language [38]. Current-art async EDA tools include BALSA [81], Tangram [82], Communication Hardware Process (CHP) [83], and the very recent suite of async tools called TiDE (recently released to universities and research institutions) from Handshake Solutions [84], etc [38], [85].

Tangram and BALSA are very similar and they both use a process called syntax-directed compilation [38] to map the desired circuit into a structure of handshake components. BALSA is developed by the University of Manchester and Tangram is not generally available outside Philips. The recent async TiDE tools include a silicon programming language called Haste and other async simulation and testing tools [84]. Haste and Tangram are similar and both of them are developed by Philips. CHP targets a highly optimized transistor-level implementation of QDI four-phase dual-rail circuits. CHP is also not generally available outside Caltech.

The prevalent commercial tools (such as VHDL or Verilog tools) for sync designs can also be used for async designs. However, as they are not designed for high-level of concurrency and lack built-in primitives for async handshake channels [38], the resulting associated overhead for async handshake components is often heavy. The NTU async research group and several other groups are working on

incorporating concurrency and primitives for async designs into the Verilog and/or VHDL tools.

As in sync EDA tools, async EDA tools require a set of well-established and well-characterized async library cells. This is due to the diversity and complexity of circuit designs and specifications. The quality of a circuit design in part depends on the quality of the library cells and their placement. In view of this and the above, a number of async microcells and macrocells are designed in this PhD program, and they form part of the NTU async cell library.

In summary, the async EDA tools are nascent and embryonic compared to the well-established sync tools, and the support of these async tools is also limited. It is envisioned that the async EDA tools will continue to evolve and consequently mitigate some of the barriers against the general acceptance of async designs.

2.2.2.3 Basic Asynchronous-Logic Microcells

In this section, some of the established async cells [38], [80], [86]–[90] will be described, including (i) *C*-Muller gates, (ii) latches, (iii) async latch controllers, and (iv) a pulse generator circuit. These microcells are pertinent to the realization of the async FFT/IFFT processor (see Chapter 5).

(i) *C*-Muller Gates

Figs. 2.9 (a) and (b) respectively depict the schematic and symbol of the standard two-input *C*-Muller gate. The inputs are *A*, *B*, and $\overline{Reset}$, and the outputs are Q_n and $\overline{Q_n}$. When the $\overline{Reset}$ is '0' (low reset), the output Q_n of the *C*-Muller gate is reset to '0' for initialization. When the $\overline{Reset}$ is '1', the *C*-Muller gate performs an 'AND' operation (for Q_n) if both inputs *A* and *B* are of the same logic state. If inputs *A* and *B* are of opposite logic states, the output Q_n remains unchanged.

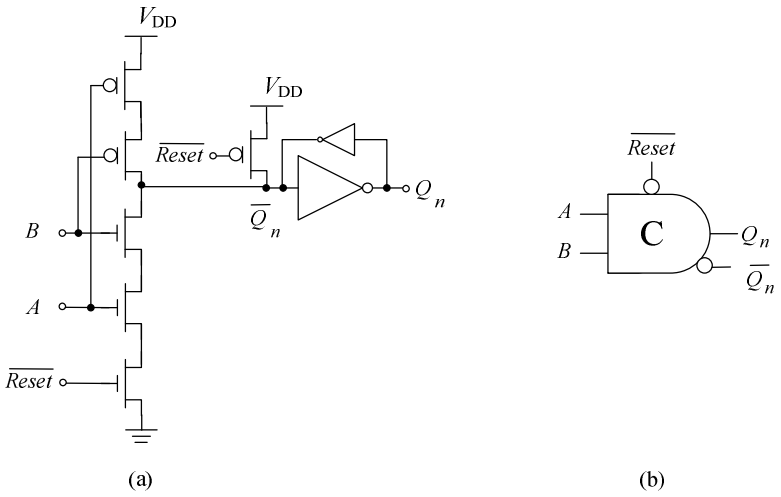

Fig. 2.9 *C*-Muller circuit: (a) schematic, and (b) symbol

Figs. 2.10 (a) to (d) depict four other variations of *C*-Muller gates for different handshake operations [38], [86]. With these different operation properties in various *C*-Muller circuits, various async handshake circuits [38], [80], [87], including the async latch controllers, can be realized.

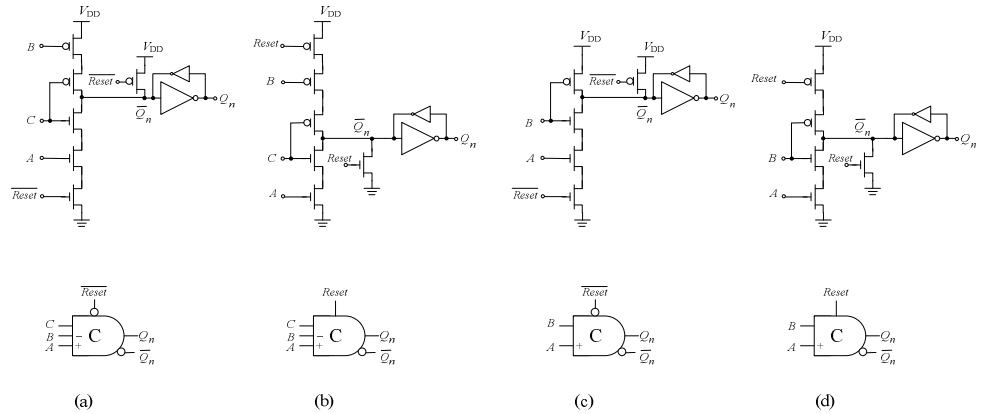

Fig. 2.10 Different *C*-Muller gate designs: (a) three-input asymmetric *C*-Muller gate with low reset, (b) three-input asymmetric *C*-Muller gate with high reset, (c) two-input asymmetric *C*-Muller gate with low reset, and (d) two-input asymmetric *C*-Muller gate with high reset

In this thesis, all five *C*-Muller gates are employed in the async FFT/IFFT processor.

(ii) Latches

Latches are widely used as storage elements in async designs [38], [77], [86]–[89] and are less costly (in terms of IC area and energy dissipation) than flip-flops (commonly used in the sync designs).

Figs. 2.11 (a) – (d) depict four popular latches and they are, in this thesis for brevity, termed as the Transmission Gate (TG) latch [86], N-typed single phase (NSP) latch [88], Transparent latch (Trans latch) [89], and Multiplexer latch (Mux latch) [75] respectively. The TG and NSP latches are inverting latches. The TG latch features a low transistor count and hence occupies a small IC area. Although the NSP latch is more complex (than the TG latch), its *En* signal

features a low input capacitive load [86]. This low input capacitive load on the *En* signal is particularly attractive for low energy dissipation if the input *A* does not change frequently [86]. The Trans latch and Mux latch are non-inverting latches with a $\overline{Reset}$ signal. The Trans latch is a conventional C^2MOS latch whereas the Mux latch is a multiplexed-based latch. The Mux latch is arguably the most robust latch because it is a fully static CMOS design – other latches are semi-static CMOS designs. The merits and disadvantages of these different latches are well established [86], [88], [89].

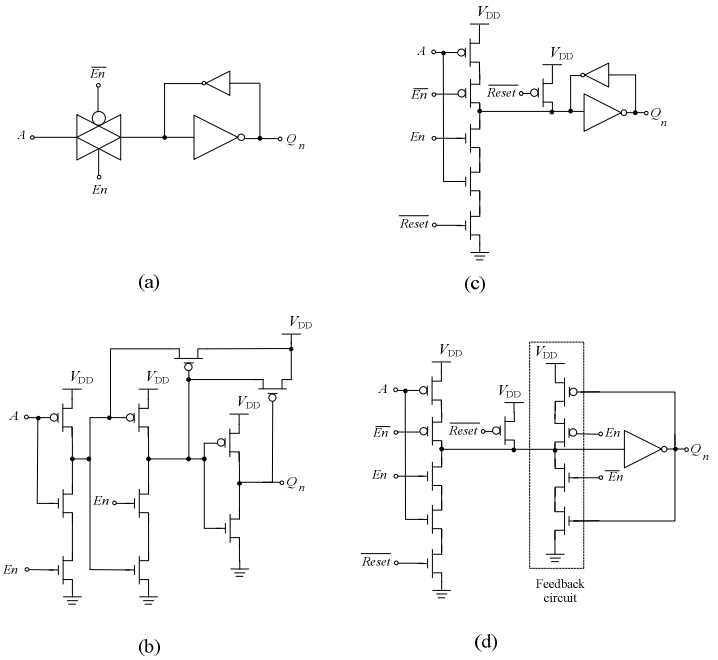

Fig. 2.11 Different latch designs: (a) TG latch, (b) NSP latch, (c) Trans latch, and (d) Mux latch

In summary, the TG latch and the NST latch are respectively (and arguably) most attractive for area-critical applications and for storing highly correlative data for low energy dissipation. The Trans latch is also well accepted for its low energy attribute and the Mux latch for its added robustness.

In this thesis, only the NSP latch and Mux latch are employed in the realization of the async FFT/IFFT processor.

(iii) Asynchronous-Logic Latch Controllers

The async latch controllers serve to coordinate the handshake signals in a pre-defined sequence. Figs. 2.12 (a) and (b) depict two common async latch controllers [80], [90]. The upper figures in Fig. 2.12 are their schematic diagrams and the lower figures are their corresponding signal transition graphs (STGs). The first latch controller in broadish data valid scheme depicted in Fig. 2.12 (a) is, in this thesis for brevity, termed as Broadish latch controller. The second latch controller in broad data valid scheme depicted in Fig. 2.12 (b) is, in this thesis for brevity, termed as Broad A latch controller.

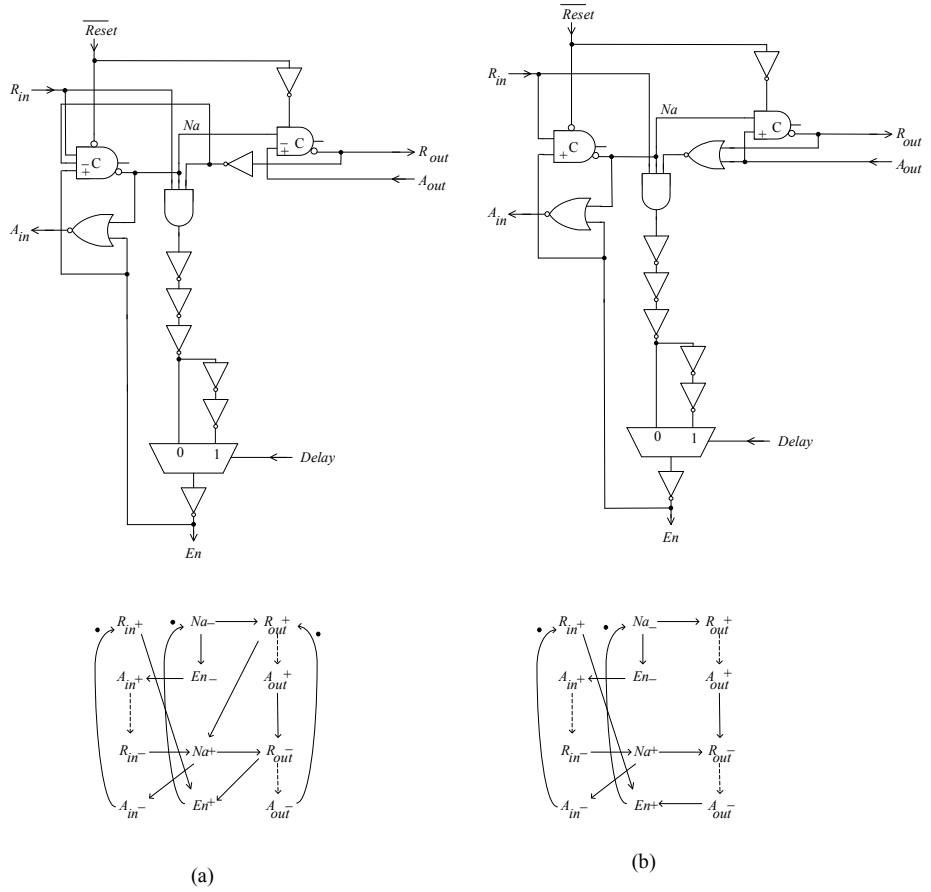

Fig. 2.12 Different async latch controller designs: (a) Broadish, and (b) Broad A

In this thesis, the Broadish and Broad A latch controllers are employed in the design of the async FFT/IFFT processor.

(iv) Pulse Generator

The pulse generator [20] generates a short pulse after the rising (or failing) edge of the signal and its pulsewidth is usually determined by delay lines. In some cases, it behaves like a local 'clock' signal. However, in the context of async circuit functionality, it operates asynchronously depending on the handshake and control signals. The self-reset property (high to low) of the pulse generator makes it very

easy to control registers, particularly for the controllers based on the four-phase protocol.

Figs. 2.13 (a) and (b) depict a pulse generator circuit and its timing diagram respectively. This pulse generator is based on the glitch generator [20] but slightly modified. The inputs are *A* and *B*, and the output is *Q*. Input *B* is the delayed signal from output *Q* (through a delay line). There are two scenarios for this pulse generator: (a) the pulsewidth of input *A* is longer than the duration of the delay line, and (b) the pulsewidth of input *A* is shorter than the duration of the delay line.

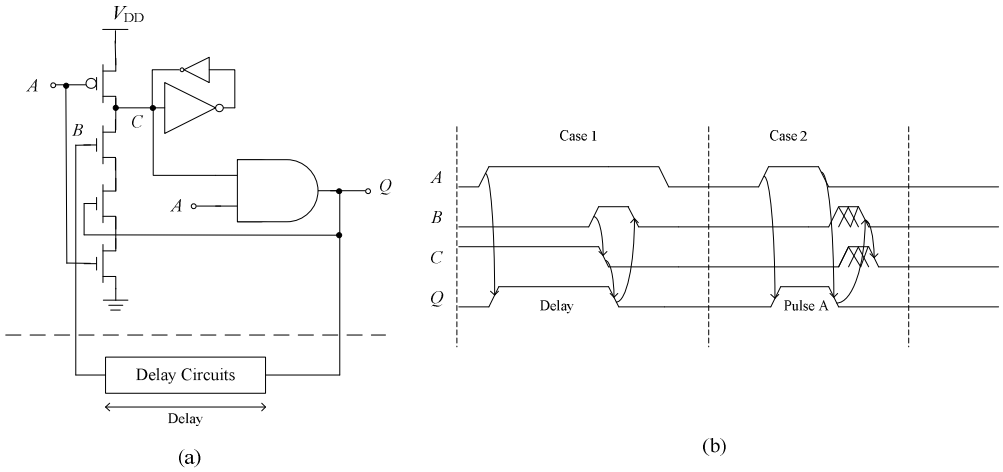

Fig. 2.13 The pulse generator (a) schematic, and (b) its timing diagram

In the first scenario, the pulsewidth of output *Q* is determined by the delay of the delay line. Initially, *A*, *B*, and *Q* are '0'. When *A* is asserted, output *Q* will become '1'. After a predetermined delay (through the delay line), signal *B* will be asserted. As a result, the intermediate node *C* will become '0' (through the

NMOS transistors) and the output will be reset to '0' (signal *B* will also become '0').

In the second scenario, the pulsewidth of output *Q* is determined by the pulsewidth of signal *A* itself. For the same initial conditions of *A* and *B* in the first scenario, output *Q* will be set to '1' if signal *A* is '1'. The output *Q* is hence asserted and set to '1'. Subsequently, signal A becomes '0' and it resets output *Q* to '0' (before signal *B* to be '1'). Finally, signal *B* will be stabilized at '0'.

In this thesis, the pulse generator is employed in the async Pulse Circuit in the FFT/IFFT processor (see Chapter 5).

2.2.2.4 Potential Advantages of the Asynchronous-Logic Approach

Many researchers [38]–[40] have suggested that due to the inherent properties of async circuits, there are many potential advantages (over the sync counterpart), including low power/energy dissipation, high speed operation, less electro-magnetic interference (EMI), robustness against variations (e.g. V_{DD} variation, temperature variation, etc), better modularity, no global clock distribution and skew problems, etc. The mechanisms behind for these advantages were alluded to in the reviews earlier and discussed elsewhere [38]–[40].

The NTU async research group has debated the potential advantages, disadvantages and challenges of async designs. Table 2.4 tabulates a comparison between sync and async designs (present and future) based on discussions within

the NTU async group. Although this comparison, to varying degrees, is highly contentious, it does provide some direction for research.

TABLE 2.4 COMPARISONS BETWEEN SYNC AND ASYNC DESIGNS (NORMALIZED TO A FULLY SYNC DESIGN)

	Sync	Async	
		Present	Future
Design effort	1.0	2.5 – 5.0	1.5 – 2.0
Energy dissipation	1.0	0.25 – 2.0	0.1 – 1.2
IC area	1.0	1.1 – 2.0	1.0 – 1.5
Delay	1.0	0.1 – 2.5	0.1 – 1.2
EMI	1.0	< -25dB	< -30dB
Modularity	1.0	> 1.0	>> 1.0
Battery life	1.0	1.0 – 4.8	1.0 – 12

As discussed in Chapter 1 earlier, the design effort of async circuits is substantially more considerable (estimated ~ 2.5 – 5.0 times) than that of sync designs. This is largely due to the lack of advanced async EDA tools and established async library cells. With more advanced async tools [53], [91], [92] and library cells developed in the future, the design effort of async design is likely to be reduced. Nonetheless, async designs have yet to be accepted as a serious alternative to sync designs and the maturity of async designs will likely lag sync designs in the immediate term.

Depending on the application, the range of the energy dissipation of async designs is widely estimated to be between ~ 0.25 – ~ 2.0 times that of sync designs, that is, the async design may dissipate from only one quarter to two times that of an equivalent sync design. This large range of energy dissipation have been reported

[4], [38], [42], [44], [46], [48]–[52], [84], [93]–[95] and in the proposed design (see Chapter 5). These comparisons are highly contentious as some of the reported designs embody novel low energy techniques that are not strictly applicable only to async circuits – they are applicable to sync designs as well. It is envisioned that the energy dissipation of async designs may be as little as 10% of their sync counterparts [93], [94] for highly energy–critical applications by fully exploiting its inherent properties.

In general, the IC area required [4], [38], [42] for async circuits is usually larger than that of sync circuits due to the added handshaking overhead. However, in view of the noise and IR (current-resistance) drop issues, it appears that the IC area required for the clock infrastructure and decoupling capacitance (in a typical complex sync design) is significant [96], hence the IC area required for async circuits may be comparable or slightly larger than that of sync circuits in some cases and possibly in future.

Depending on the application, the range of the delay of async designs can be very wide, estimated to be between ~ 0.1 – ~ 2.5 times that of sync designs. This large range have been reported [38], [42], [46]–[51], [93], [97] and in the proposed design (see Chapter 5). The speed improvement of async designs is more apparent in high-performance SoC designs, and processor interconnects for different IPs (intellectual properties) and I/Os running at different speeds [93].

Async designs have been demonstrated to emit lower electro-magnetic energy than their sync counterparts (e.g. 8051 microcontrollers in [84], [95], and ARM

processor in [98]). This is largely due to the absence of the complex clock infrastructure. The lower EMI attribute means that async circuits may emit lower interference and arguably better 'stealth' properties, hence their lower electronic presence.

Async circuits theoretically feature higher modularity due to their localized handshake interfaces and the local timing [38], [78]. This high modularity translates to simpler design iterations, hence lower redesign cost and shorter design cycles. With the standard handshake protocols and handshake circuits becoming increasingly more standardized, it is envisioned that the modularity of async circuits would be further enhanced.

In addition to the possible lower energy dissipation attributes of async circuits that translate to a longer battery life, a further advantage of async circuits in battery-operated applications is that async circuits can operate at a lower V_{DD} albeit at a slower speed. This is because async circuits are essentially self-timed circuits.

In summary, although the async design approach innately has many potential advantages over the prevalent sync designs, only a few practical designs have been demonstrated to date. It is envisioned that with the limitations of heat and speed of current sync designs in state-of-the-art fabrication processes, the async design may serve as a potential alternative.

2.3 Fast Fourier Transform and Inverse Fast Fourier Transform (FFT/IFFT)

The FFT is essentially derived from the DFT [24], [25] and an N-point DFT is expressed in (2.8), and its inverse (IFFT or DFT) expressed in (2.9):

$$X(k) = \sum_{n=0}^{N-1} x(n) \cdot W_N^{nk} \tag{2.8}$$

$$x(n) = \frac{1}{N} \sum_{n=0}^{N-1} X(k) \cdot W_N^{-nk} \tag{2.9}$$

where $W_N^{nk} = e^{-j2nk\pi/N} = \cos(2nk\pi/N) - j\sin(2nk\pi/N)$,

$W_N^{-nk} = e^{j2nk\pi/N} = \cos(2nk\pi/N) + j\sin(2nk\pi/N)$,

$X(k)$ is the output (frequency) sample,

$x(n)$ is the input (time) sample, and

$k = 0$ to N-1.

As discussed in Chapter 1 earlier, the frequency resolution is an important parameter in spectrum analysis performed by a DFT or FFT. Frequency resolution is expressed in (2.10) where $f_{sampling}$ is the sampling frequency of the signal. The frequency resolution is used to resolve the differences between frequencies for a set of given input signals. The larger number of points N, the better is the frequency resolution. In typical audio signal processing applications, $N \geq 128$.

$$\Delta f = \frac{f_{sampling}}{N} \tag{2.10}$$

It is well-established that the FFT algorithm (for $N=2^L$, L is a positive integer) decomposes the N-point DFT successively into a number of smaller-sized DFTs, known as butterflies, by taking advantage of the periodicity and symmetry of the coefficient, W_N^{nk} , known as the twiddle factor [25]. With this decomposition, the total number of operations for the FFT is significantly lower (than that of the DFT), resulting in reduced energy dissipation. For example, a radix-2 FFT requires only $(N/2)\log_2 N$ complex multiplications, and for N=128, this translates to a ~ 97% reduction in complex multiplications over the DFT.

There are many different FFT algorithms (e.g. radix-2, radix-4, radix-2^2, radix-8. radix-2^3, split-radix, CORDIC-based FFT and their different combinations) and correspondingly many different architectures [26]–[34], [120], [138], [141]. In this thesis, the established radix-2 FFT algorithm is adopted for its operational regularity and for its hardware simplicity – other higher radix or split-radix FFT algorithms may require fewer operations but at the cost of operational irregularity and increased design effort. The radix-2 FFT algorithm decomposes the N-point DFT successively until each butterfly is a 2-point DFT. There are two types of radix-2 FFT algorithms, namely the decimation-in-time (DIT) and decimation-in-frequency (DIF) [25]. The DIT FFT divides the input samples (time) into two groups, one for odd input samples and the other for even input samples. Conversely, the DIF FFT computes the output samples (frequency) into two groups, one for the odd frequency samples and the other for the even frequency

samples. In terms of computation, both types are the same. In this thesis, DIT is chosen primarily for its simpler address sequencing control of the twiddle factors. Fig. 2.14 depicts the simplest DIT radix-2 butterfly.

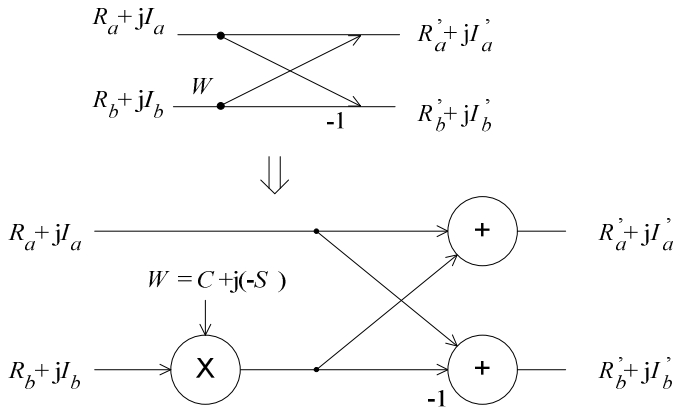

Fig. 2.14 A radix-2 butterfly

$R_a + jI_a$ and $R_b + jI_b$ are the two inputs, and $R'_a + jI'_a$ and $R'_b + jI'_b$ are the corresponding two outputs. The subscripts, *a* and *b*, denote the two different data. *W* is the twiddle factor where *C* is the cosine coefficient and *S* is the sine coefficient. Equations (2.11a) – (2.11d) describe the operation for one radix-2 butterfly.

$$R'_a = R_a + [(C) \cdot R_b - (-S) \cdot I_b] \tag{2.11a}$$

$$R'_b = R_a - [(C) \cdot R_b - (-S) \cdot I_b] \tag{2.11b}$$

$$I'_a = I_a + [(C) \cdot I_b + (-S) \cdot R_b] \tag{2.11c}$$

$$I'_b = I_a - [(C) \cdot I_b + (-S) \cdot R_b] \tag{2.11d}$$

As expressed in (2.8) and (2.9) earlier, the DFT (FFT) algorithm and IDFT (IFFT) algorithms are similar with the exception of the scaling factor 1/*N* and the twiddle factors. This means that the IFFT algorithm can be obtained by using the same FFT algorithm and simply negating the sine coefficients and multiplying the outputs by a scaling factor of 1/*N*.

It is well-known that a radix-2 butterfly potentially causes the output data to grow by two bits resulting in a data overflow. The usual methods to accommodate overflow include input data scaling, output data scaling (unconditional block floating point (BFP) scaling), and conditional BFP scaling [25]. The former two methods simply involve scaling the inputs or outputs at the cost of reduced dynamic range of the signal. The last method, conditional BFP scaling, is more complex but maximizes the dynamic range of the signal. For the FFT/IFFT processors designed in this thesis, the conditional BFP scaling is adopted.

This scaling works as follows. Initially, the inputs are scaled by two bits, known as guard bits, to accommodate the overflow problem in the first stage. For the subsequent stages, the bit growth is monitored in each butterfly operation. When the butterfly operations in each stage are completed, the outputs of the next stage will be scaled (shifted) if there is any bit growth. If there is no bit growth, the outputs remain unscaled for the butterfly operations in next stage.

The bit growth for guard bits can range from 0 to 2 bits (right shift). Fig. 2.15 depicts the flow chart of the conditional BFP scaling for one FFT stage. After each stage, a block exponent (denoted as Exp in Fig. 2.15) will update the total

number of bit growth. As the maximum guard bits are 2 bits, only 3 most significant bits (MSBs) of the output are needed to be checked for the bit growth. For example, if the 3 MSBs of the output are 000 (positive value) or 111 (negative value), no overflow occurs. If the 3 MSBs of the output are 001 or 110, a one-bit overflow occurs. If the 3 MSBs of the output are 01*x* or 10*x*, where *x* is either 0 or 1, a two-bit overflow occurs. Note that the bit growth taken (denoted as BG in Fig. 2.15) is based on the largest bit growth amongst all the outputs monitored.

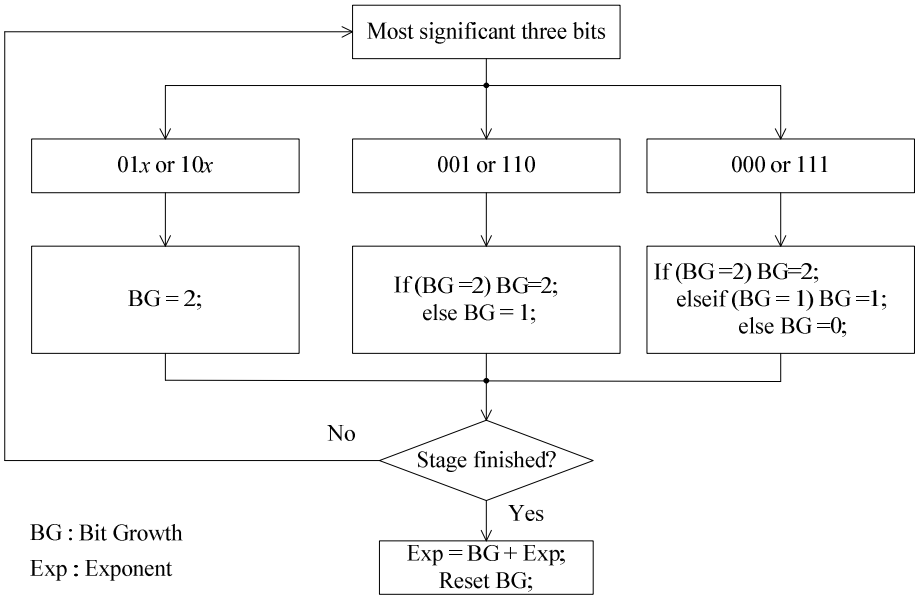

Fig. 2.15 Flow chart of the conditional BFP scaling for one FFT stage

2.4 Summary

This chapter has reviewed and discussed three topics, (i) low power low energy design techniques, (ii) sync and async design approaches, and the (iii) FFT/IFFT algorithm.

Section 2.1 has reviewed the mechanisms of power (energy) dissipation in digital circuits, the low power/energy design methodologies, and the noise margin analysis. The low power/energy attributes of a design is a result of collective design efforts at each level of the design – from the algorithm, architecture, circuit/logic down to the device/process – and taking into consideration the interdependencies amongst these levels. Some of the low power/energy methodologies have been adopted in the proposed circuits in this thesis.

Section 2.2 has reviewed sync and async design approaches. In the latter, the async design review included fundamental concepts of async design, current-art async EDA tools, some basic async microcells (pertinent to the async FFT/IFFT processor), and the potential advantages of the async approach.

Section 2.3 has reviewed the FFT/IFFT algorithm.

Chapter 3

Asynchronous-Logic Microcells

3.1 Introduction

During the course of this PhD program, more than 50 different custom microcell designs (or > 140 microcells including the same design with different driving attributes) were designed and their layouts handcrafted. These microcells serve as part of the NTU async cell library.

For the sake of brevity and coherence with the research theme of the thesis, only the microcells that have significant novelty and pertinence to the realization of the async FFT/IFFT processor (see Chapter 5) will be delineated. The proposed novel microcells include the LA, Latch Accumulator, 2-bit *Type-*γ CCSA, and Broad B latch controller. A large portion of this chapter is extracted from the author's recent publications (or accepted) [59]–[65], including publications in the *IEEE Trans. VLSI Systems* [56] and in the *IEE Proc. Circuits, Devices, and Systems* [58].

As in usual industry practice, all microcells are designed to abide by the library design guidelines [20], [99], [100] and are compatible with the existing standard library cells (with the selected 0.35μm CMOS process). For example as shown in Fig. 3.1, the height of the library cells is fixed – the height of the library cells (such as cells A, B, C and D) is 13μm (including the V_{DD} and *GND* lines), and for

larger cells (e.g. cell E), the height is doubled to 26μm. The width of the power lines (V_{DD} and *GND*) is also fixed at 1.8μm each. The length of the cell can also be varied to accommodate the various complexities of the different cells. As shown in Fig. 3.1, and since all library cells have the same height, the custom library cells can be easily intermixed with the existing standard library cells, resulting in greater flexibility to employ library cells from different libraries. The library cells are also designed in such a way that they can abut seamlessly to one another and/or be flipped (either side) without violating the geometry design rules.

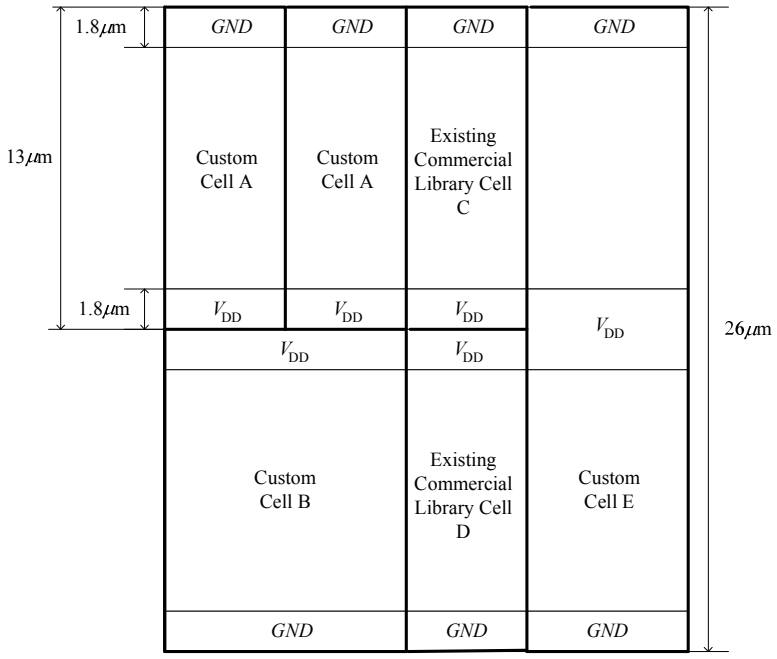

Fig. 3.1 A symbolic template for library cells

Several low energy techniques are adopted for designing the microcells, including careful layout [99], finger technique [70] for large width/length (*W/L*) transistors to reduce the drain/source capacitance, short interconnect, etc. The routability for the microcells is also considered, by defining fixed routing directions for different metal layers. Most of the microcell layouts use only metal-1. Metal-2 and

metal-3 are only considered for more complex microcell designs, with metal-2 in the vertical direction and metal-3, in the horizontal direction. Although the development of these microcells is time-consuming, it is a worthwhile effort as the performance of a circuit greatly depends on the underlying microcells.

The novelty of the proposed microcells includes the incorporation of different functional features (e.g. latch and delay enhancements), resulting in increased versatility, compactness and lower energy dissipation. The proposed LA incorporates a design that integrates an adder and a latch, and the proposed Latch Accumulator incorporates a design that integrates an LA and a latch. The LA and Latch Accumulator will be shown to have lower energy dissipation, smaller IC area, and shorter delay compared to reported designs. The proposed *Type-*γ CCSA includes four added delay balancing transistors, resulting in a better delay robustness than reported CCSAs. The proposed Broad B latch controller features a design that eliminates the possibility of malfunction of the interconnecting successive functional modules.

This chapter is arranged in the following manner. Section 3.2 describes the proposed LA and Latch Accumulator. Section 3.3 first reviews async adder designs and thereafter describes the proposed async 2-bit *Type-*γ CCSA. Section 3.4 describes the proposed Broad B latch controller. Finally, Section 3.5 summarizes this chapter.

3.2 Proposed Latch Adder and Latch Accumulator

3.2.1 Latch Adder

In most of the arithmetic operations (such as multiplication, division, etc.), full adders are the most rudimentary cells. The designs of full adders are generally mature in literature [20], [71], [101]–[107]. A full adder sums three inputs, A, B and carry-in C_{in}, and produces two outputs, sum S and carry-out C_{out}. The prevalent adder designs for low voltage operations that provide rail-to-rail output include the T28 adder [20], T18 adder [102], T16 adder [105], *N-P* dynamic adder (DA) [106] and Domino DA (see Figs. 3.2 (a)–(e)). Other adder designs with a small transistor count (e.g. < 15 transistors per full adder), for example designs in [101], [103], [104], operate unreliably under low voltage conditions ($V_{DD} \approx |V_{Tp}| + V_{Tn}$), and are hence unsuitable for the low voltage applications including the intended hearing aid. Pass-logic adders [101], [103], [104] are sensitive to voltage scaling, and hence also unsuitable for the low voltage applications.

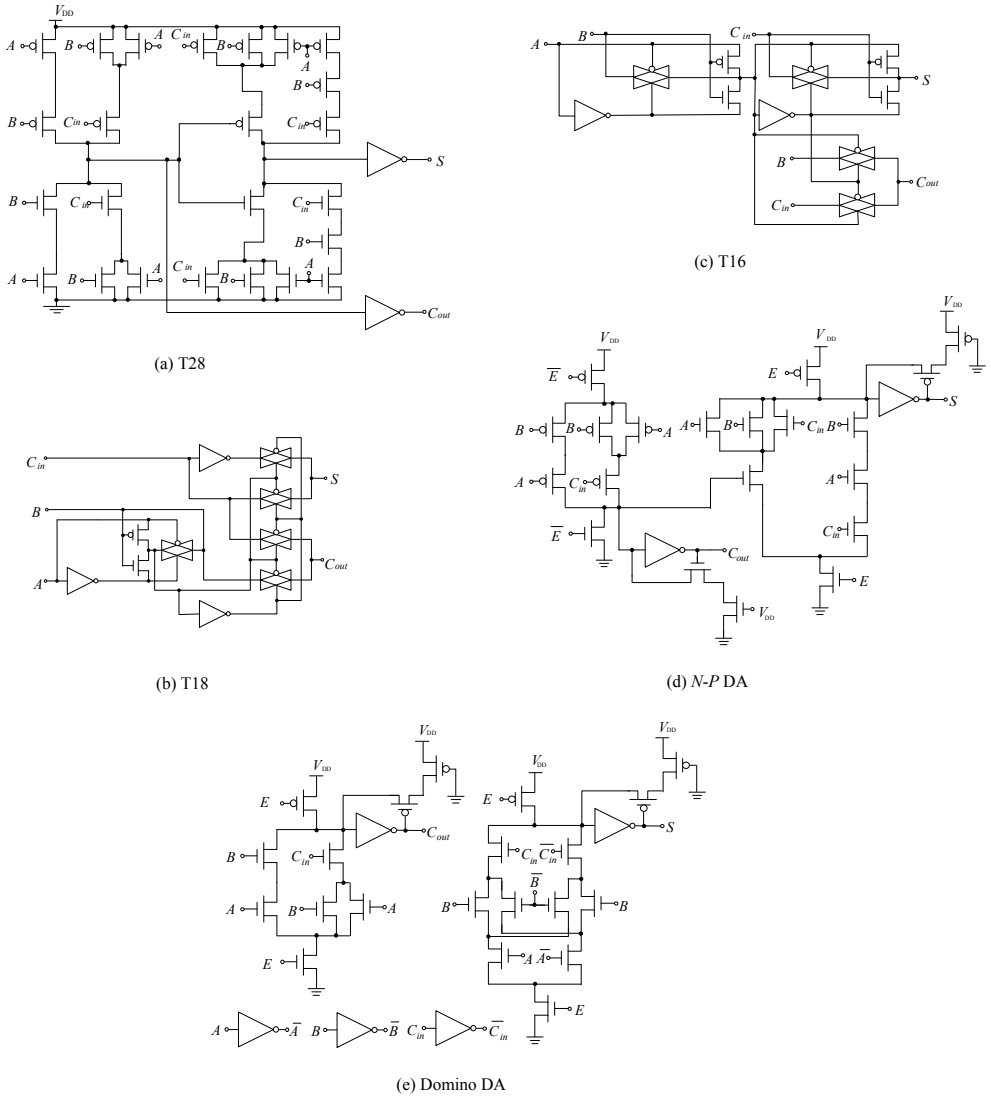

Fig. 3.2 Conventional full adders that provide rail-to-rail signal swing at low voltage operation: (a) T28 adder, (b) T18 adder, (c) T16 adder, (d) *N-P* DA, and (e) Domino DA

The T28 adder is a conventional 28-transistor static CMOS design. The T18 and T16 adders are transmission-gate designs and they comprise 18 transistors and 16 transistors respectively. These full adders (except the DAs) are asserted when their inputs change, and substantial spurious switching results if their inputs are poorly synchronized. The *N-P* DA comprises 24 transistors (or 22 transistors if only one PMOS and one NMOS weak transistors are used) and is a variation of

the NORA (No Race) CMOS serial adder. The operations of the *N-P* DA are controlled by the two additional enable signals, E and $\overline{E}$. The Domino DA comprises 31 transistors (or 29 transistors if only two PMOS weak transistors are used) and its sum output is a 3-way XOR function. The operations of the Domino DA are controlled by one additional enable signal, *E*.

A simple way to reduce spurious switching is to place separate latches in front of the full adder to synchronize the inputs to the adder. However, the cost in terms of IC area (three latches are required per full adder) and the added power dissipated by these latches are high [108]. To circumvent this cost and yet obtain a latch function at the input to the adder to reduce the spurious switching, a latch is integrated into an adder, termed as Latch Adder (LA) [56]. By integrating a latch (that latches all the 3 inputs simultaneously) within an adder instead of being separate and external, the overhead in LA is low (see later). The circuit schematic of the proposed LA is depicted in Fig. 3.3.

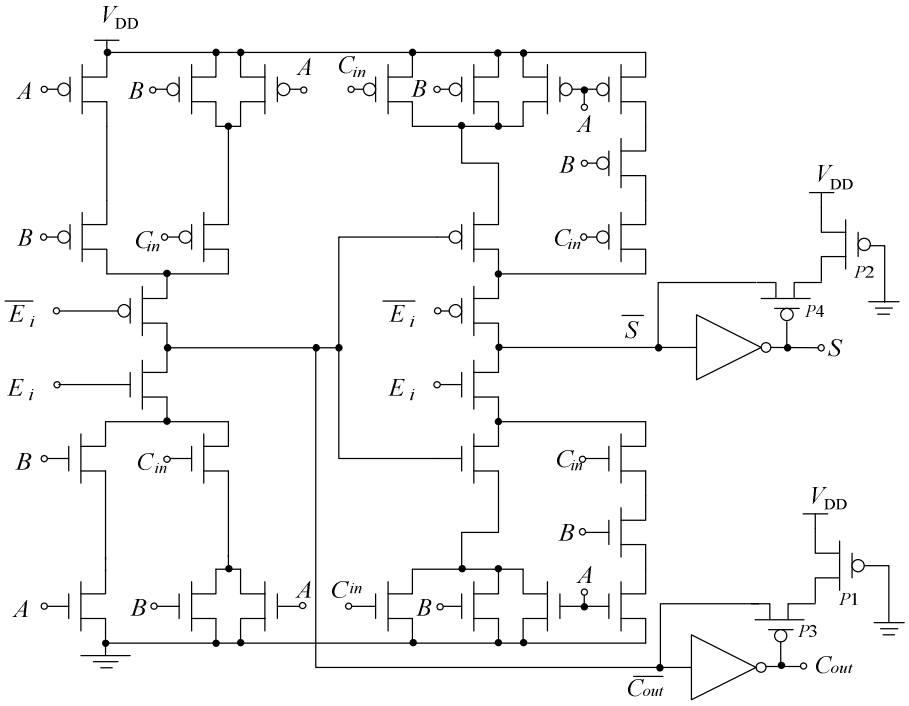

Fig. 3.3 Circuit schematic of the Latch Adder (LA)

The latch function within the LA is controlled by means of the enable signals, E_i and $\overline{E_i}$, and serves to synchronize the 3 input signals (A, B and C_{in}) to the LA. In Fig. 3.3, when E_i = '1' ($\overline{E_i}$ = '0') or asserted, the LA functions as a full-adder. When E_i = '0' ($\overline{E_i}$ = '1') or negated, the LA will hold its output signals (S and C_{out}) by means of weak (small W/L ratio) feedback PMOS transistors ($P1$ to $P4$), and the outputs of the adder are hence latched. Note that although the LA design depicted in Fig. 3.3 is a semi-static CMOS design, its operation is robust. This is shown by considering the following two scenarios. First, assume that S is initially '1'. If E_i = '1', there is no ambiguity. For the same initial condition, if E_i = '0', $\overline{S}$ is at a high impedance state, hence an undesirable condition. This is because charge leaking into $\overline{S}$ may result in an undesirable change in state.

This is, however, not an issue because for E_i = '0', the output is not used. Put simply, for E_i = '0', the output is inconsequential because the LA is in idle state. Second, assume that S is initially '0'. If E_i = '1', there is again no ambiguity. For the same initial condition, if E_i = '0', $\overline{S}$ is a well-defined '1' due to feedback inverter. The same scenarios apply to signal C_{out}. In short, the operation of the proposed LA is robust.

To retain the low power/energy advantage in the proposed design, the weak feedback inverters are designed by cascading $P1$ and $P2$ to $P3$ and $P4$ respectively [56]. The $P3$ and $P4$ transistors are sized to minimum-size (constrained by the design rules), and the $P1$ and $P2$ transistors are sized to have an effective low transistor W/L ratio. In this manner, these transistors present a low capacitive loading at outputs S and C_{out}. This is as opposed to the conventional method where only $P3$ and $P4$ (both with much longer L) are used, thereby presenting a larger capacitive load at outputs S and C_{out}. Also note that the short-circuit current is negligible when the targeted operating voltage is as low as 1.1V (close to $|V_{Tp}| + V_{Tn}$).

It would be of interest to note that if the supply voltage is relatively low ($V_{DD} < |V_{Tp}| + V_{Tn}$), there is no power dissipation due to short-circuit current, and the dynamic CMOS approach can be adopted (without weak feedback PMOS transistors, $P1$ to $P4$). If the supply voltage is relatively high ($V_{DD} > |V_{Tp}| + V_{Tn}$), for the robustness, for reducing the short-circuit current and for enhancing its logic state, weak NMOS transistors can be included (fully static CMOS approach) at the expense of relatively slower speed, larger area and higher switching power

dissipation (compared to dynamic CMOS approach). The semi-static CMOS approach provides a good compromise between the dynamic CMOS and fully static CMOS approaches. Compared to the former approach, the semi-static CMOS approach results in some short-circuit current reduction. Compared to the latter approach, the semi-static CMOS approach results in a lower switching power dissipation and higher speed due to relatively smaller output loads.

The proposed LAs are particularly useful to reduce the spurious switching in multiplier designs, and more illustrations will describe how this low spurious switching can be achieved in Chapter 4.

Two benchmarked comparisons are made on the basis of computer simulations. First, the proposed LA is compared against the reported full adders (see Fig. 3.2). Second, the proposed LA is further compared against the reported full adders with separate latches. The separate latch is based on the low power semi-static non-inverting C^2MOS (with two series weak PMOS transistors implementation). In all the designs, the transistors are sized to have ×1 driving ability (the same as the standard library cells), specifically (2μm/0.3μm) and (1μm/0.3μm) for PMOS and NMOS respectively (except for the weak PMOS and NMOS transistors). Note that all designs are based on the same 0.35μm CMOS dual-poly three-metal fabrication process. The low transistor-count adders and pass-logic adders that do not operate reliably at 1.1V are not considered here.

On the basis of 500 random input signals and on simulations, Table 3.1 summarizes the energy dissipation, delay, EDP, and IC area of the different adders.

From Table 3.1, it is noted that among the conventional adders (T28, T18, T16) and the LA, the LA dissipates the highest energy, is the slowest speed (for *S* signals) and occupies the largest IC area. This is expected due to the added integrated latch function. On the other hand, among all adders compared, the Domino DA is the fastest adder but dissipates the highest energy due to its dynamic operation. Despite featuring 7 less transistors than the Domino DA, the *N-P* DA features high energy dissipation, comparable to the Domino DA. This is because of its dynamic operations, additional control line $\overline{E}$ and the PMOS tree having a larger capacitance. The low mobility of the PMOS transistor (lower than that of NMOS transistor) makes the PMOS tree (and its associated NMOS weak transistor for charge sharing prevention) less attractive in the *N-P* DA. As a result, the *N-P* DA features a relatively poor speed performance compared to the Domino DA.

TABLE 3.1 ENERGY, DELAY, ENERGY-DELAY-PRODUCT (EDP) & IC AREA OF THE PROPOSED LATCH ADDER AGAINST THAT OF OTHER FULL ADDERS @ 1.1V, 1MHZ

Adder	Energy (10^{-8}W/MHz)	Delay (ns)		EDP (10^{-23} J.s)	Core Area (μm^2)
		C_{out}	S		
T28	6.0	3.4	4.8	28.8	389
T18	7.3	4.1	4.3	31.4	374
T16	7.2	5.0	5.9	42.5	387
N-P DA	11.7	7.0	8.3	97.1	486
Domino DA	11.8	1.9	2.7	31.9	660
Latch Adder	10.1	4.4	7.0	70.7	501

The advantage of the LA design becomes more apparent in the second benchmarked comparison as tabulated in Table 3.2. In this table, the circuits now

have the same circuit functionality as the LA described above. When compared against the T28, T18 and T16 adders with separate latches, the proposed LA features the lowest energy dissipation, the least delay (for *S* signal), and occupies the smallest area. As described previously, these desirable parameters are attributed to the integrated latch in the LA.

TABLE 3.2 ENERGY, DELAY, ENERGY-DELAY-PRODUCT (EDP) & IC AREA OF THE PROPOSED LATCH ADDER AGAINST THAT OF OTHER ADDERS WITH LATCHES @ 1.1V, 1MHZ

Adder	Energy (10^{-8}W/MHz)	Delay (ns)		EDP (10^{-23} J.s)	Core Area (μm^2)
		C_{out}	S		
T28 with latches	15.4	5.6	7.3	112.4	1180
T18 with latches	15.8	8.3	8.3	131.1	970
T16 with latches	15.8	9.4	11.0	173.8	1000
Proposed LA	10.1	4.4	7.0	70.7	501

3.2.2 Latch Accumulator

An accumulator accumulates the addition result and uses a master-slave flip-flop (equivalent of using two latches) to latch and subsequently feedback the result for the next addition [64]. The block diagram of a usual 1-bit accumulator is depicted in Fig. 3.4. Note that the adder and flip-flop are separate circuit entities. The inputs to the accumulator are *A* and C_{in} and the outputs, C_{out} and sum, *S*. The accumulated output, *Q*, is an internal signal that is fedback to the accumulator. The accumulator is reset when *Reset* is asserted. The trigger signal, *TR*, synchronizes the accumulator.

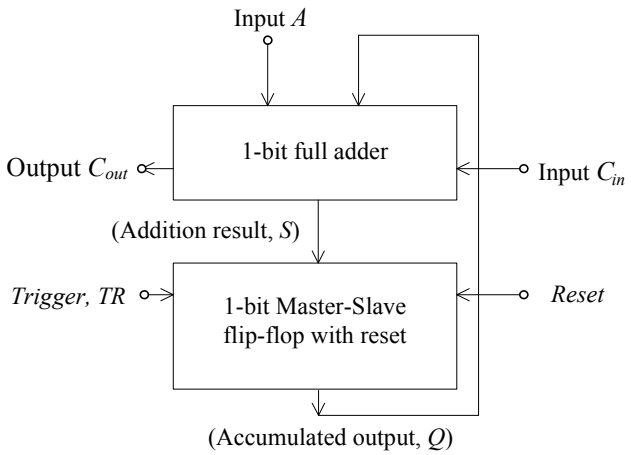

Fig. 3.4 The simplified block diagram of an accumulator realized by usual design (a standard adder and a separate flip-flop)

To obtain a lower power/energy accumulation, the LA is further integrated with an additional latch to be a Latch Accumulator. As delineated previously, instead of having the adder and the flip flop as separate circuit entities, the proposed Latch Accumulator integrates them into the same circuit entity. The merits of this design would be lower energy dissipation, higher speed (hence smaller EDP), and a smaller IC area requirement – see later for the quantifications.

Fig. 3.5 depicts the proposed Latch Accumulator [58] using the proposed LA as the core cell. It is noted that as the LA is able to block the input signals and it holds the output signal, the function of the flip-flop can be realized with the proposed LA. In Fig. 3.5, the proposed 1-bit accumulator integrates the output part of the LA and the input part of the flip-flop (one latch) such that it not only gates the input signals, but also feedbacks the output signal to the adder.

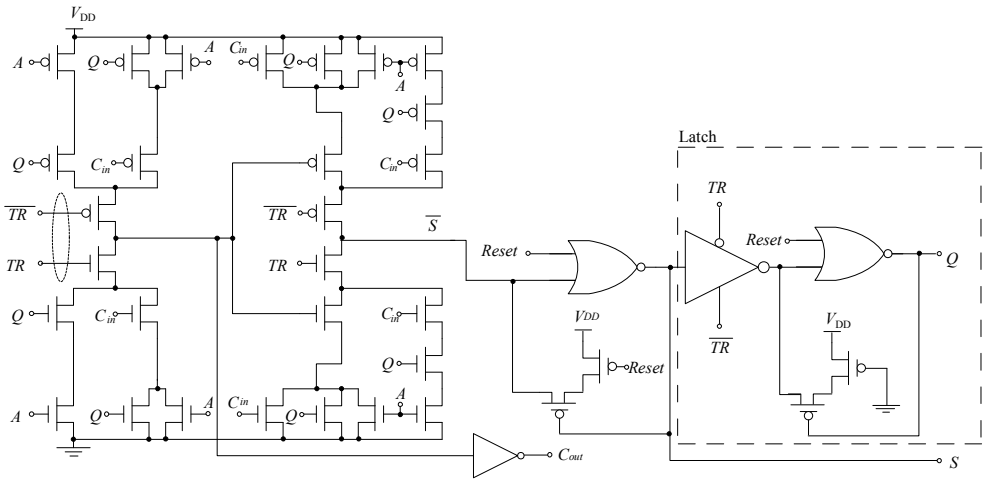

Fig. 3.5 The proposed Latch Accumulator

A prototype 16-bit Latch Accumulator IC has been realized using a 0.35μm dual-poly three-metal CMOS process and the microphotograph of the design is shown in Fig. 3.6. Table 3.3 tabulates a comparison (on the basis of 1,000 random signals) of the proposed 16-bit accumulator and a standard 16-bit accumulator. The simulated parameters and the measured parameters for the proposed design generally agree well. The conventional accumulator is implemented using a T28 adder and a low power flip-flop from a 0.35μm digital cell library. From Table 3.3, on the basis of simulations, the proposed 16-bit accumulator features ~ 22% lower energy, ~ 5% faster, ~ 28% better EDP and ~ 24% smaller when compared the standard design. The proposed design is superior largely because the flip-flop is now an integrated part of the Latch Accumulator.

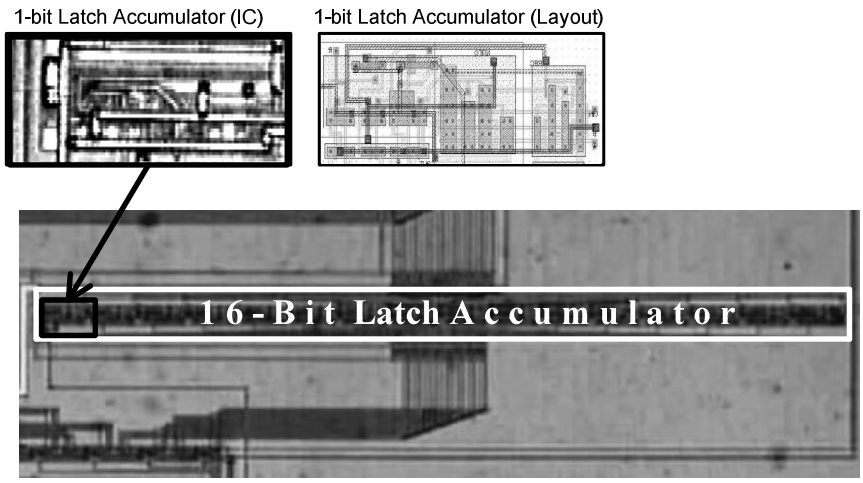

Fig. 3.6 Microphotograph of the proposed 16-bit Latch Accumulator

TABLE 3.3 DELAY, ENERGY, ENERGY-DELAY-PRODUCT (EDP) & IC AREA OF VARIOUS 16-BIT ACCUMULATORS @ 1.1V, 1MHZ BASED ON POST-LAYOUT SIMULATIONS AND MEASUREMENTS ON PROTOTYPE ICS

16-bit Accumulator	Simulations			Measurement			Area (mm^2)
	Delay (ns)	Energy (μW/MHz)	EDP (10^{-21}J.s)	Delay (ns)	Energy (μW/MHz)	EDP (10^{-21}J.s)	
Standard	60	2.7	163.2	—	—	—	0.017
Proposed	57	2.1	116.9	58	2.3	133.4	0.013

3.3 Asynchronous-Logic Adder Cell

3.3.1 Review of Asynchronous-Logic Adder Cells

The async adder is widely used and is one of the basic components in an async data path. It provides its own timing information, and is triggered by a *REQ* signal, and a *COMP* signal is generated upon the completion of the addition. Note that the async adder usually dissipates higher energy dissipation than the

combinational full adders due to the additional circuitry required for implementing the completion detection. However, it has several advantages. First, it provides a completion signal when the addition process is completed. Second, its operation is based on actual computation time – this means that its delay varies for different inputs and the fast average-case delay is usually considered. This is different from the combinational full adders whose worst-case delay is normally used.

Several async adders [4], [54], [55], [109]–[116] have recently been reported based on various structures (carry ripple, carry look-ahead, carry select, and tree structure) for different applications. Of these async adders, the carry ripple adder is arguably more prevalent for its low energy dissipation – in view of this, the carry ripple adders will only be considered here. A long wordlength carry ripple adder (e.g. 16-bit) is usually implemented by cascading a number of shorter wordlength (e.g. 1-bit or 2-bit) adders. Of these rudimentary 1-bit or 2-bit adders, the CCSAs [4], [55], [63], [109], [116] are arguably most prevalent in async carry ripple adders. In the CCSA, only the carry-out signals are implemented using dual-rail logic whereas the sum signals are implemented using single-rail logic. One basic assumption for CCSAs is that all sum outputs are assumed to be ready by the time the carry-out signals become valid. However, most of the reported CCSAs [4], [54], [55], [109], [114], [116] do not fully satisfy this timing assumption, resulting in potential timing violations (PTVs) – *COMP* may instead be generated before the outputs are ready. To safeguard against this timing violation, additional hardware, usually delay lines, are used to delay the carry-out signals appropriately, or the detection completion circuit may otherwise embody

additional hardware to appropriately delay the same; another possibility is careful design of appropriate *W*/*L* transistor ratios. This safeguard may unnecessarily complicate the system-level design – requiring timing analysis and testing due to the extra delay assumption. Depending on the specific design, this extra delay safeguard inevitably results in additional energy dissipation due to the added hardware. For example, the robust 1-bit reported CCSA design [4] features long delay and dissipates relatively high energy.

Figs. 3.7 (a) – (d) depict the reported async CCSAs, namely the dynamic CCSA [4], pass-logic CCSA [114], *Type-*α CCSA [55], and *Type-*β CCSA [55]. The dynamic CCSA, *Type-*α CCSA and *Type-*β CCSA are implemented using domino logic. The pass-logic CCSA, as its name suggests, is implemented using pass transistor logic. The dynamic CCSA and pass-logic CCSA are 1-bit wordlength while the *Type-*α CCSA and *Type-*β CCSA are 2-bit wordlength. The other sum signal, S_0, for the *Type-*α CCSA and *Type-*β CCSA, is the same as the Sum Block in the dynamic CCSA. The advantage of implementing 2-bit adder (as a basic cell) instead of 1-bit adder is the reduced redundant signals (by sharing the common transistors in its Sum and Carry blocks), resulting in lower power/energy dissipation and improved speed [109]. Note that it may appear that using an even longer wordlength *M*-bit CCSA ($M > 2$) as the basic cell may further reduce the redundancy. However, it is not usually applied because of the increased complexity [109] in the resultant design.

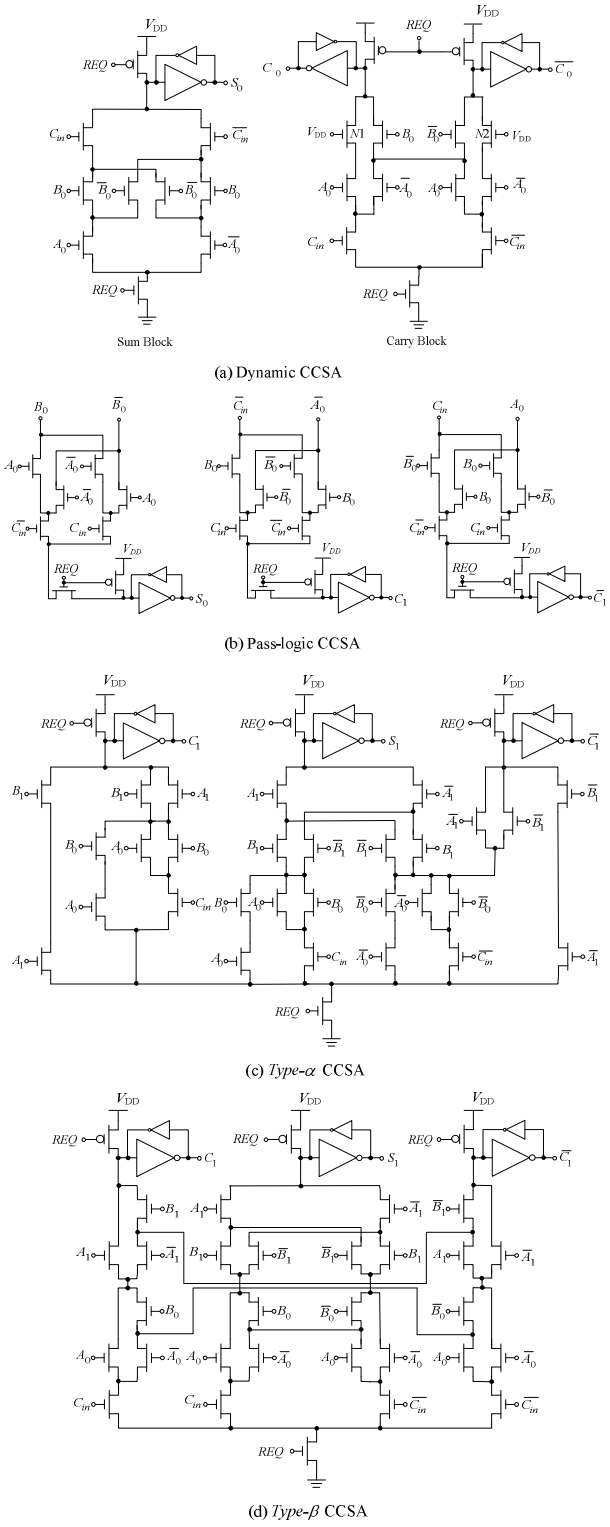

Fig. 3.7 Conventional carry completion sensing adders (CCSAs): (a) 1-bit dynamic CCSA, (b) 1-bit pass-logic CCSA, (c) 2-bit *Type-*αCCSA, and (d) 2-bit *Type-*βCCSA

It is remarked that by using complementary carry-out signals, C_1 and $\overline{C_1}$, with a completion sensing circuit [38], [55], [115], the *COMP* signal for addition can be generated. The validity of the *COMP* signal can be analyzed in two situations:

1: The delay of the sum signals is always shorter than the delay of the carry-out signals.
2: The delay of the sum signals is always shorter than the total delay from the carry-out signals and the completion sensing circuit.

It is obvious that for a robust basic adder design, situation 1 has to be satisfied. If situation 1 is not satisfied, careful attention (e.g. transistor sizing or delay adjustment) is required to satisfy situation 2. Put simply, for an async CCSA, a truly robust design needs to satisfy situations 1 and 2, a partially robust design satisfies situation 2 but not necessarily situation 1 and a weak design does not satisfy situations 1 and 2. To the best of the author's knowledge, most reported CCSAs are only partially robust – although the robustness can be achieved by means described earlier. This partial robustness may compromise the flexibility and ease of implementation of async circuits at a higher-level design.

For example, assuming the same *W*/*L* ratios for the NMOS transistors, *Type-*α and *Type-*β adders are partially robust. This is because the sum S_1 signal of these adders goes through a series of 5 or 6 NMOS transistors while the carry-out signals only go through a series of 3 – 6 transistors, resulting in a potentially slower generation of the sum S_1 (than the carry-out signals). Although the pass-logic CCSA features a similar structure for its outputs, this type of pass transistor

logic is sensitive to voltage scaling at lower V_{DD}, resulting in PTVs as well. The dynamic CCSA is likely to be a robust design as the outputs of the dynamic CCSA always go through a series of 4 transistors.

Note that the premise for the abovementioned is that the same aspect ratio (W/L) is used for all NMOS transistors. Although the robustness of all reported designs can be made truly robust by adjusting the delays of pertinent paths appropriately by resizing transistors and/or adding delay circuits, this adjustment of delay is not trivial and often requires many simulations and design iterations.

3.3.2 Proposed 2-bit Asynchronous-Logic *Type-*γ Adder

The robust 2-bit *Type-*γ CCSA is proposed and it satisfies the situations 1 and 2 without the need for careful transistor sizing and/or delay adjustment. Fig. 3.8 depicts the circuit schematic of the *Type-*γ CCSA for the sum S_1 and carry-out signals, C_1 and $\overline{C_1}$. The circuit schematic of sum S_0 signal is the same as the Sum Block depicted in Fig. 3.7(a). The *Type-*γ CCSA also employs the transistor sharing technique to share the common transistors in its Carry and Sum blocks [55], [109], resulting in reduced redundant signals. Although this design may appear similar to the *Type-*α and *Type-*β CCSAs, there are differences as discussed next.

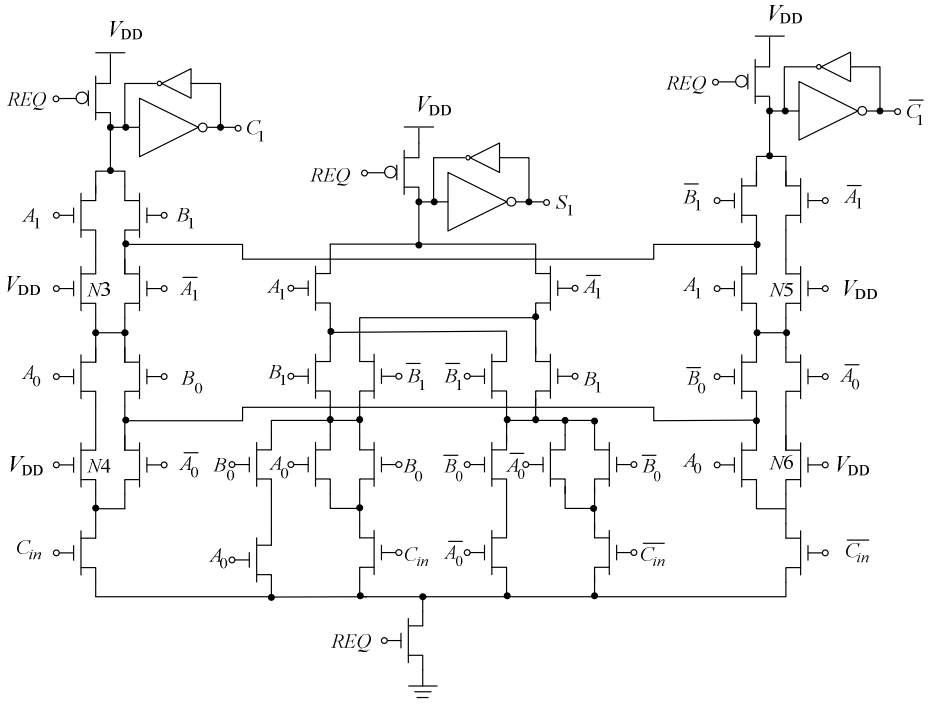

Fig. 3.8 Circuit schematic of the proposed CCSA, *Type-*γ

Assume that the NMOS transistors are to be of equal *W/L*. For the proposed design, the four delay balancing transistors, *N*3 – *N*6, are first deliberately included to delay the carry-out signals (C_1 or $\overline{C_1}$). It becomes apparent that either one of the carry-out signals always goes through a series of 6 transistors in the NMOS tree. Second, the sum S_1 signal is designed such that it always goes through a series of 5 transistors. With this circuit implementation, the S_1 signals will be generated faster than the carry-out signals and there is no ambiguity that the S_0 signal is always faster than the carry-out signals. Consequently, the *Type-*γ CCSA is truly robust (no PTVs).

Fig. 3.9 depicts an example of a 16-bit async adder embodying the proposed 2-bit *Type-*γ CCSAs; any wordlength adder can be designed. The OR gate serves as the completion sensing circuit to generate the *COMP* signal. It is noted that any

wordlength adder embodying the proposed CCSAs will be robust. For a higher average-case delay async adder design, the half of the 2-bit *Type-*γadders in LSBs can be replaced with *Type-*α 2-bit adders without affecting its robustness [63]. This is because the carry propagation in the proposed 2-bit *Type-*γ CCSA is always longer than that of the *Type-*αCCSA.

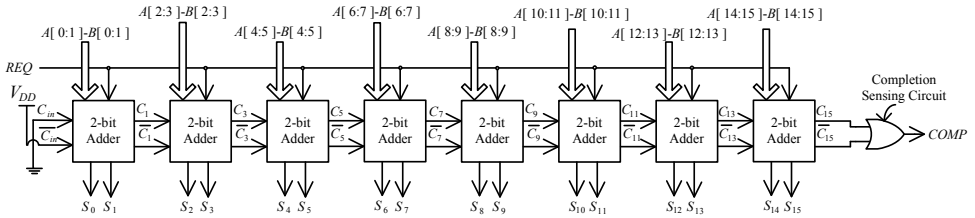

Fig. 3.9 Block diagram of a 16-bit async adder embodying the proposed *Type-*γ CCSAs

The proposed design and reported async CCSA designs are implemented based on the same fabrication process (0.35μm dual-poly four-metal CMOS process). To fairly compare these adders, the dynamic CCSAs and pass-logic CCSAs are constructed as 2-bit adders (the same as other adders) and all the adders are sized with the *W*/*L* ratios for NMOS trees to be 2.0μm/0.35μm. All input combinations (32 input patterns) are simulated for these 2-bit adders @ 1.1V, 1MHz. The total energy consumed includes the energy due to the input capacitance of the input drivers (inverters) and to the 2-bit adders. The delay is defined by the time *REQ* is asserted until the output signal becomes '1' for different input signals changed.

Table 3.4 depicts the best-case (t_b), average-case (t_a), and worst-case (t_w) delays for all outputs, average energy (*E*), the number of PTVs, and IC area for different 2-bit CCSAs. PTV refers to a failure or violation of situation 1 for a given input

pattern. From Table 3.4, it is apparent that the pass-logic, *Type-*α, and *Type-*β CCSAs have 8 – 22 PTVs. The *Type-*α CCSA is the fastest (t_a for the carry-out signals) and the *Type-*β 2-bit adder features the lowest energy among the adders compared. By scrutinizing all the input pattern combinations, it is verified that the proposed design and dynamic CCSA are robust (0 PTV) despite the worst-case delay of S_1 being longer than the best-case delay of the carry-out signals. Furthermore, the proposed design, on average, features ~ 16% faster (t_a for its carry-out signals), ~ 5% lower energy dissipation but ~ 9% larger IC area compared to the dynamic CCSA. In summary, the proposed *Type-*γ CCSA is robust and yet dissipates low energy.

TABLE 3.4 DELAY (BEST-CASE, AVERAGE-CASE, AND WORST-CASE), ENERGY (E), NUMBER OF PTVS & IC AREA OF VARIOUS ADDERS @ 1.1V, 1MHZ

CCSAs	Delay S_0 (ns)			Delay S_1 (ns)			Delay C_1 (ns)			Delay $\overline{C_1}$ (ns)			E (fJ)	PTV	Area (μm^2)
	t_b	t_a	t_w	t_b	t_a	t_w	t_b	t_a	t_w	t_b	t_a	t_w			
Dynamic	2.8	2.9	2.9	4.4	4.9	5.2	5.0	5.7	6.3	5.1	5.7	6.3	335	0	1019
Pass-logic	1.8	1.9	2.1	4.7	5.0	5.3	4.2	4.7	5.1	4.5	4.9	5.5	322	12	1037
Type-α	2.6	2.6	2.7	2.9	3.8	4.5	2.1	2.7	3.3	2.0	2.9	4.2	302	22	1074
Type-β	2.6	2.6	2.7	3.1	3.9	4.3	2.7	3.7	4.4	2.8	3.8	4.9	295	8	1092
Type-γ	2.6	2.6	2.7	2.8	3.3	3.7	3.6	4.6	5.3	3.9	4.9	5.7	317	0	1110

3.4 Proposed Broad B Latch Controller

As discussed in Chapter 2, the async latch controllers should coordinate the handshake signals in a pre-defined sequence so that the interconnecting latches and controllers can transfer data successfully.

From the STGs of the conventional latch controllers, as earlier depicted in Figs. 2.12 (a) and (b), it is noted that R_{out} may be asserted during the interval when the latches become transparent (*En* is '1'). This means that the interconnecting functional modules (if any) commence their operations even when the data are transferring into the latches. Consequently, this leads to two potential drawbacks to the functional modules. First, the functional modules may have increased spurious switching (or redundant operations) because the data may still be changing (until they become stable), resulting in increased energy dissipation. Second, the functional modules may even malfunction, in particular when dynamic circuits are used; dynamic circuits, for some situations, can only be evaluated correctly after the inputs are ready. Although these two undesirable situations can be resolved by (i) adjusting for sufficient delay of the R_{out} signal appropriately by adding delay circuits, and/or (ii) ensuring the latches have a short transfer time (and the data do not change thereafter), these adjustments are not trivial and often requires many simulations and design iterations. To prevent these situations, another latch controller, Broad B latch controller, is proposed [62] and is depicted in Fig. 3.10.

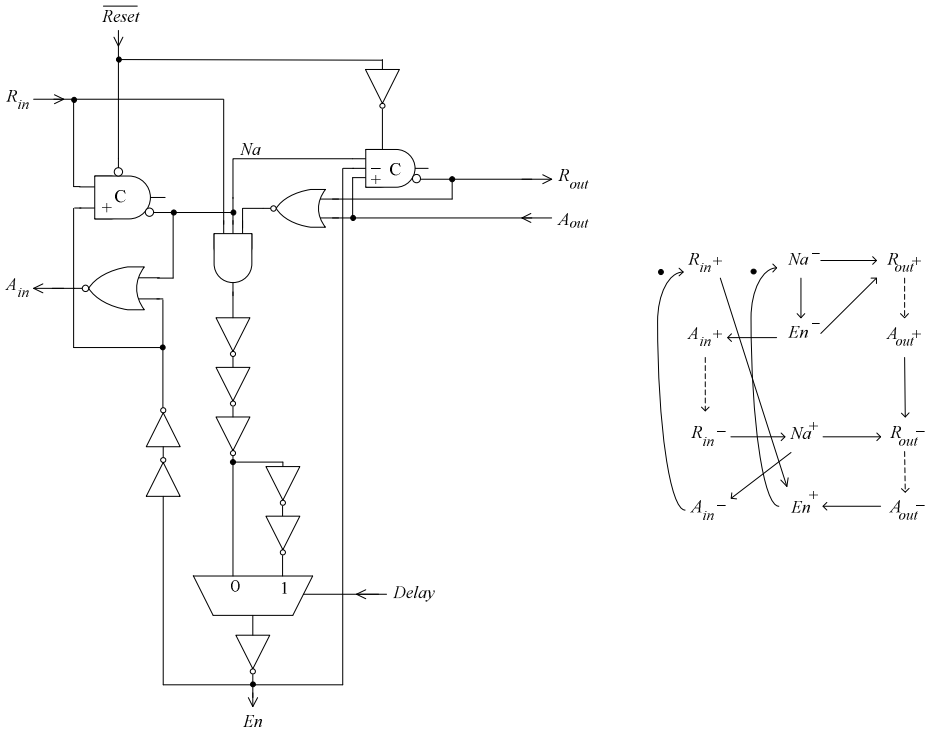

Fig. 3.10 The proposed latch controller: Broad B latch controller

As illustrated in the STG in Fig. 3.10, R_{out} in the Broad B latch controller will only be asserted if *En* is de-asserted. This means that by the time R_{out} is asserted for operation in functional modules (if any), the new data will have been latched (in steady state). This is done by using the three-input asymmetric *C*-Muller to generate R_{out} when only *En* is de-asserted. This realization of the proposed Broad B latch controller has the advantages to reduce spurious switching within the successive functional modules and more importantly to prevent malfunction of the successive functional modules. The drawback of the Broad B latch controller is the longer required delay for R_{out}.

The assumption made in the operation of the proposed Broad B latch controller (and other latch controllers, see Chapter 2) is that the predetermined delay (the

pulse from *En*– to *En*+ to *En*–) has to be sufficiently long enough to allow the complete data transfer to the latches. This delay is controlled by the inverter chain therein.

3.5 Summary

More than 50 microcells (or more than 140 microcells, including the same design with different driving attributes) have been developed and handcrafted during this research exercise. Only the novel microcells have been described herein and the emphases are low energy dissipation and robustness (compatible with the existing standard library cells and that can be supported with commercial EDA tools).

The proposed novel designs included the LA, Latch Accumulator, async 2-bit *Type-*γ adder, and the Broad B latch controller. The LA used its integrated latch to synchronize its input signals (with different arrival times), thereby resulting in possible reduced redundant operations (spurious switching). Compared to adders with latches where the adders and latches were separate entities, the proposed LA featured the least delay, the lowest energy dissipation, the best EDP, and the smallest IC area. The Latch Accumulator integrated an LA and an additional latch. By means of this integration, the proposed Latch Accumulator featured shorter delay, lower energy dissipation, better EDP, and smaller IC area compared to the conventional accumulator. The async 2-bit *Type-*γ adder eliminated the potential timing violation problems by means of the insertion of four delay balancing transistors. By using the hybrid *Type-*α and *Type-*γ adders in a carry

ripple structure for a longer wordlength adder, the delay of the long wordlength adder has been shortened (compared to the long wordlength adder using only *Type-*γ adders). The async Broad B latch controller removed the possibility of triggering the interconnecting successive functional modules (if any) while the input data were being transferred into the latches. It hence prevented the malfunction of the functional modules (in particular where dynamic circuits were used).

Chapter 4

Macrocells: Multiplier and Memory Designs

4.1 Introduction

This chapter describes three proposed designs, a 16×16-bit Booth array-based multiplier core, an async Control-Multiplier specifically for the FFT algorithm, and a 128×16-bit memory macrocell. A large portion of this chapter is extracted from the author's recent publications (or accepted) [59]–[62] including publications in the *IEEE Trans. VLSI Systems* [56], the *IEEE J. Solid-State Circuits* [57], the Singapore patent (granted) [59], and the USA patent (granted) [60]. As a matter of interest, there are other macrocell designs but are of synchronous-logic in nature (hence incoherent to be included in this thesis) proposed by the author, including a filter bank and a Leapfrog multiplier that are published elsewhere [10], [117], [118].

As mentioned in Chapter 1 earlier, the bulk of the operations for the FFT/IFFT algorithm are multiplications and memory accesses, and these operations are relatively high power (energy) dissipation operations (compared to simpler operations such as additions). In a typical FFT/IFFT processor, the multiplications and memory accesses each generally dissipate > 30% of the total power dissipation, hence a total of > 60%. It is hence imperative for the multiplier and memory macrocell designs to be low power/energy dissipation in

order to significantly reduce the overall power/energy dissipation of the FFT/IFFT processor.

The proposed 16×16-bit Booth array-based multiplier core is a multiplier embodying a novel technique to reduce spurious switching; the high spurious switching is a classical problem (see later) in multiplier designs [56], [108]. The basic idea behind this reduced spurious switching technique is the replacement of most of the adders in the adder array of the multiplier by the proposed LAs described in Chapter 3 earlier. With the latch function (in the proposed LA) and by means of simple delay circuits, the inputs to the adders (in the adder array of the multiplier) are synchronized in a predetermined chronological sequence, thereby substantially reducing the spurious switching. The switching activity in the proposed 16×16-bit and 32×32-bit multiplier designs is analyzed and the analysis will show that the number of switchings (spurious and that for computation) is respectively reduced from ~ 5.6 and ~ 10 per adder (in the adder array of the conventional 16×16-bit and 32×32-bit designs) to ~ 2 per adder (in the proposed designs). The 16×16-bit multiplier core is verified by computer simulations and on basis of experimental measurements on prototype ICs. It will be shown that the proposed design dissipates ~ 32% less energy, is ~ 20% slower but has ~ 20% better EDP than conventional designs. In the case of a 32×32-bit multiplier, the proposed design dissipates ~ 53% less energy, is ~ 29% slower but has ~ 39% better EDP than the conventional general array multiplier. The prototype IC and the conventional array multipliers are realized using a 0.35μm dual-poly three-metal CMOS process.

The proposed async Control-Multiplier macrocell is an async multiplier especially suited for the FFT/IFFT algorithm. It embodies the abovementioned proposed Booth array-based multiplier core and added control circuits. The proposed Booth array-based multiplier core serves to compute non-trivial multiplications, and the added control circuits are for trivial multiplications (i.e. $\times 1, \times 0$, and $\times -1$) and for async handshaking. It will be shown that a further 22% energy reduction (compared to the stand-alone proposed multiplier core) is obtained based on simulations where 40% (similar % in the 128-point FFT algorithm) of the multiplications are trivial. The async Control-Multiplier is realized using a 0.35μm dual-poly four-metal CMOS process and is embodied in the FFT/IFFT prototype IC.

The 128×16-bit async memory macrocell is an async SRAM that employs several well-established low energy design techniques, including the partition of a large memory block into several sub-memory blocks [20], [21]. The memory macrocell further includes proposed Word Line Controllers to reduce unnecessary sub-memory block assertions. Its attributes are low energy dissipation and small IC area. The async memory macrocell is realized using a 0.35μm dual-poly four-metal CMOS process and is embodied in the FFT/IFFT prototype IC.

This chapter is arranged in the following manner. Section 4.2 first reviews multiplier designs and thereafter describes the proposed 16×16-bit Booth array-based multiplier core and the async Control-Multiplier. Section 4.3 describes the proposed 128×16-bit memory macrocell. Finally, Section 4.4 summarizes this chapter.

4.2 Multiplier Macrocell

4.2.1 Review of Multiplier Designs

Multipliers can be broadly classified into two types: serial [119] and parallel [120]. The serial multiplier receives its inputs bit by bit (i.e. serial processing) and then performs the additions and accumulations iteratively by means of full adders and storage elements. The key advantages of the serial multiplier are small hardware circuitry and small IC area; the small IC area used to be one of the most important criteria in about a decade ago and earlier due to the low IC area density. The drawbacks of the serial multiplier are slow processing speed (due to its iterative processing) and high energy dissipation per multiplication (due to the high energy dissipated in the storage elements and adders).

The parallel multiplier, on the other hand, receives its inputs simultaneously (i.e. parallel processing) and then performs all the additions therein concurrently (once the inputs to the adders are ready). The advantages of the parallel multiplier are improved speed and relatively lower energy dissipation (than the serial multiplier). The drawback of the parallel multiplier, however, is the larger IC area requirement. Due to the prevalence of the parallel multiplier (as opposed to the serial multiplier) for current-art DSPs, the parallel multiplier is only reviewed and is the type of interest in this thesis.

Fig. 4.1 depicts a parallel multiplier comprising two function blocks, namely the partial product generator (PPG) and the Adder Block.

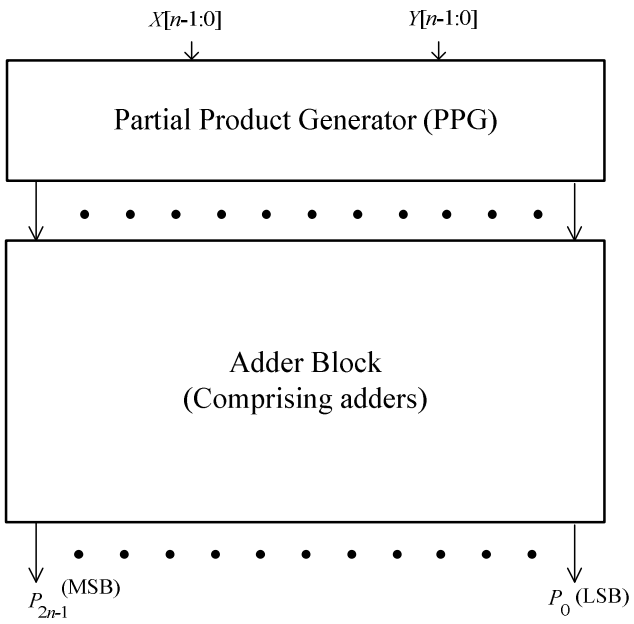

Fig. 4.1 A simplified block diagram of a parallel multiplier

From an n-bit multiplicand X input and an n-bit multiplier Y input, the PPG generates the partial products which are subsequently added in the Adder Block to obtain a $2n$-bit multiplication product, P. Of the different parallel multiplier designs, the tree-based (Wallace tree, Wallace & Dadda's scheme, algorithmic approach) [120]–[126] and array-based structures [56], [118], [120], [126]–[128] are ubiquitous for their respective high speed and regular layout (small area) attributes. In terms of power/energy dissipation, the tree-based structure is potentially lower power/energy than the array-based structure if the former is properly designed [126]. This is because the tree-based multiplier embodies a smaller number of rows of adders in the Adder Block compared to the array-based multiplier, hence the consequent lower spurious switching but at the cost of the

increased layout difficulty and possibly increased parasitic. In other words, the power/energy dissipation (and speed and IC area) of a tree-based multiplier would, to a large degree, depend on layout and fabrication technology specifics. In view of the intended hearing aid application where small IC area and ease of implementation in layout (many power/energy critical circuit blocks are custom designed) are imperative, and where speed is secondary (< 5MHz), only the parallel array-based multipliers will be considered for comparisons with the proposed design (see later). It would be worthwhile to note that the proposed method to reduce spurious switching applies to both array-based and tree-based multipliers.

It is well established that the primary power/energy dissipation mechanism of the multiplier is dynamic switching power/energy. In typical designs, there is substantial spurious (redundant) switching in the Adder Block in a multiplier, resulting in wasted power/energy dissipation. The spurious switching is largely attributed to the different arrival times of the input signals to the adders in the Adder Block. The spurious switching propagates from the adders in the first row of the Adder Block to the adders in the latter rows where the amount of spurious switching increases. The wasted power/energy dissipation due to spurious switching is substantial, typically in the order 40% of the total power/energy dissipation of 16×16-bit multipliers, and is large for longer wordlength multipliers, for example ~ 60% for 32×32-bit multipliers [56].

To reduce spurious switching in the Adder Block in a multiplier, reported designs include adders with output C^2MOS latches [129] (for a pipelined multiplier-

accumulator), ECDL-based (enabled/disable CMOS differential logic) adders [128], dynamic adders (DAs) with delayed-evaluation [130], a different array-based structure design such as the Leapfrog multiplier [127], delay matching for the intermediate adder signals [131], delay balancing for the partial products [132], dynamic range determination units with flip-flops placed in front of the multiplier [133] and more direct designs that employ latches placed in front of the adders [108]. Although the output C^2MOS latch does reduce the spurious switching by latching the output signal (thereby preventing the spurious switching from propagating to the following stage), substantial spurious switching within the adders remains, in particular where the arrival times of the input signals into the adders are poorly synchronized. The ECDL-based adders/DAs, on the other hand, require reset/pre-charge and enable/evaluate operations to appropriately time the turning on/off of the ECDL adders/DAs and these additional switchings defeat the advantages gained. Furthermore, the ECDL-based adders require complementary signals which double the switching activity, and require a larger IC area.

The Leapfrog array-based structure in a multiplier can reduce the switching to a certain degree. However, substantial spurious switching remains especially if the input signals to the adders are poorly synchronized. The delay matching for the intermediate adder signals by means of delay circuits (delay lines) to synchronize the input to the adders is somewhat impractical for a large ($\geq$ 16×16-bit) multiplier design because of the high cost (area and power) in implementing numerous delay circuits. The delay balancing technique for the partial products reduces the power/energy dissipation in the Adder Block but the power/energy dissipation advantage may then be offset by the higher power/energy dissipation in the PPG

and larger IC area. The dynamic determination units with flip-flops placed in front of the multiplier are advantageous in reducing the spurious switching in the PPG and to some extent, in the Adder Block. However, the degree of spurious switching reduction in the latter strongly depends on the input data and the dynamic range determination unit, and the flip-flops incur expensive power/energy overhead. Similarly, the advantage gained from the simplistic design of placing latches in front of the adder is defeated by the cost in hardware and the added power/energy dissipated by the added latches.

Another way to reduce the power/energy dissipation in the parallel multiplier is by 'brute-force' truncation at the cost of higher quantization errors (compared to the standard multiplier without truncation) and this multiplier is called as truncated multiplier [65], [134], [135]. This truncation involves removing a number of partial products in the LSBs, and this in turn reduces the number of additions in the Adder Block. In many applications, the resulting quantization is too serious and is inappropriate for general purpose applications.

4.2.2 Proposed 16×16-bit Booth Array-Based Multiplier Core

The majority of multipliers employ the modified Booth's algorithm [120], [136] for the PPG. This is because the use of the Booth algorithm results in a small number of partial products and this in turn reduces the number of rows (and hence the number of adders) in the Adder Block. In the case of the 2's complement format, a correction term is usually added so that the hardware required for the sign bit extensions (that would otherwise incur substantial hardware overhead) is

small [137]. Fig. 4.2 depicts the partial products of a 16×16-bit 2's complement Booth radix-4 multiplier generated by the Booth PPG. Rows *a* to *h* contain the partial products and the LSB and the MSB are respectively at the extreme right and at the extreme left sides. The partial products in row *r* correspond to modified Booth encoded carry-in signals.

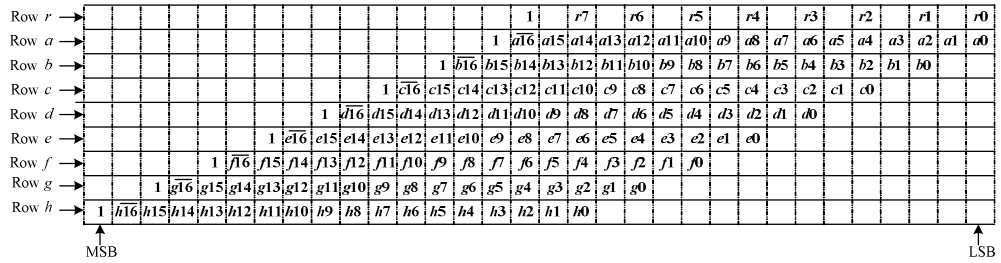

Fig. 4.2 The partial products in a 2's complement 16×16-bit Booth radix-4 multiplier

In a parallel array multiplier, the multiplication process simply involves the summation (by means of full adders) of the partial products row by row in a regular fashion in the Adder Block. Although the regularity of the array structure is changed slightly to accommodate the sign bit in 2's complement format, the majority of the full adders are still placed in a regular fashion. The array structure has the advantage (over a tree-based structure) of simple interconnects between the full adders.

Fig. 4.3 depicts the block diagram of the proposed 16×16-bit array multiplier core [56] embodying the Booth PPG, delay circuits (D_1-D_7) and Adder Block. The Adder Block comprises 7 rows (Row 1 – Row 7) of LAs and a final row of carry ripple adders (CRAs) that serves a carry propagation adder.

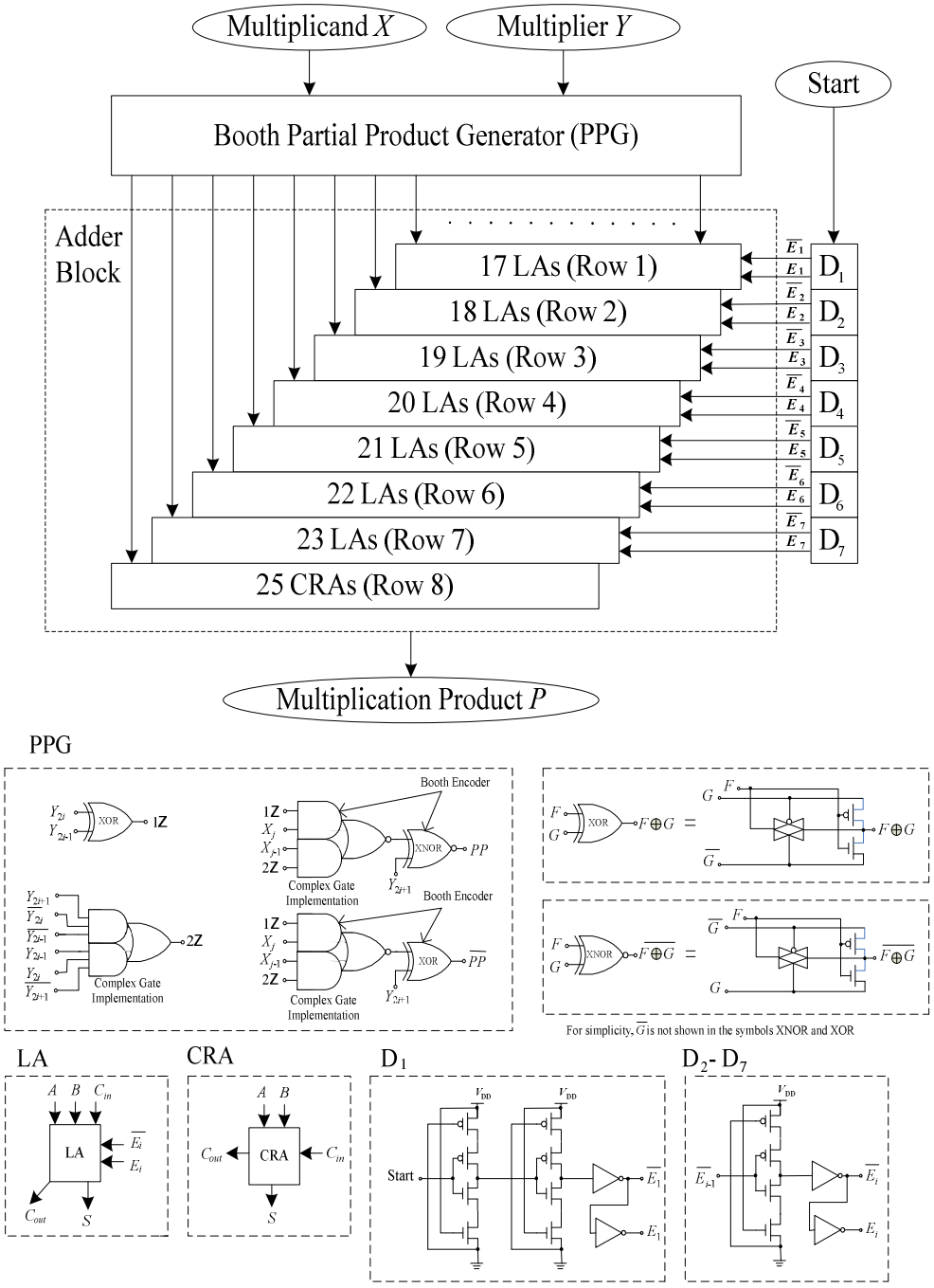

Fig. 4.3 Block diagram of the proposed 16×16-bit Booth array multiplier, SB-Proposed16

The operation of the multiplier is as follows. When a multiplication is initiated with the multiplicand *X* and multiplier *Y* inputs, the Start signal asserts D_1. After a predetermined delay in D_1 that is at least equal to the duration required of the

PPG to generate the partial products, E_1 and $\overline{E_1}$ will assert the latches in the LAs of Row 1 and assert D_2. As the inputs to the adders in the LAs in Row 1 are synchronized, there is little, if any, spurious switching in this row. In the remaining rows of the Adder Block, there is no switching as the LAs there remain negated. Subsequently, after a predetermined delay in D_2 that is at least equal to the duration of the delay in the adders in Row 1, E_2 and $\overline{E_2}$ will assert the latches in the LAs in Row 2 and assert D_3. Similarly, as the inputs to the adders in the LAs in Row 2 are synchronized, there is little, if any, spurious switching in this row. Similarly, there is no switching in the remaining rows as the LAs there remain negated. The process repeats until Row 7, and the spurious switching in Rows 1–7 of the Adder Block is virtually eliminated. The last row of CRAs computes the final multiplication product *P*.

The delay circuit is usually implemented using an inverter chain with appropriate *W/L* transistor sizing. The normal approach is to increase *L* (thereby decreasing the *W/L* ratio) as a means to increase the delay, however this would compromise the load capacitance. Instead, the inverter depicted in Fig. 4.4 is designed where transistors *P*5 and *N*5 are used to adjust the equivalent *W/L* ratio of the P ($(W/L)_{eq,p}$) and N ($(W/L)_{eq,n}$) -type transistors of the inverter, hence the amount of delay. This approach is the same as the double series PMOS weak inverter as described in Chapter 3 earlier. With this implementation, the input gate capacitance is determined by transistors, *P*6 and *N*6, both having a small diffusion area. For example, based on post-layout simulations, the total node capacitance at the input of the new delay circuit and at the conventional delay circuit (simple inverter) is 11.5fF and 24.9fF respectively, for the same delay of 5ns. The energy required

for the preceding input driver (a standard inverter) to drive the new delay circuit and the conventional delay circuit is 12.5fJ and 30.8fJ respectively @ 1.1V. This translates to ~ 59% less energy dissipation if the new delay circuit is adopted. The output of the delay circuit is buffered by two standard inverters to drive the control signals E_i and $\overline{E_i}$, depicted in Fig. 4.3.

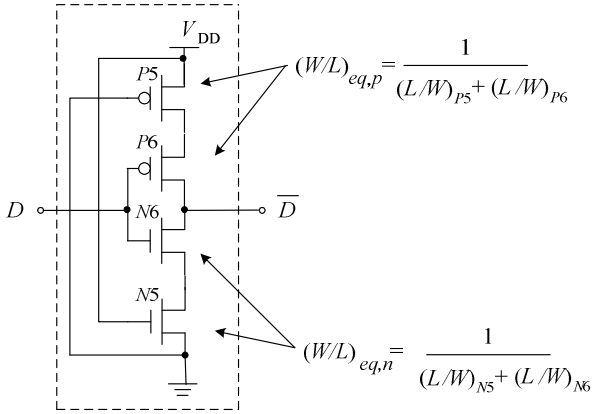

Fig. 4.4 Delay circuit

To delineate the efficacy of the proposed approach to reduce spurious switching, Fig. 4.5 depicts, from simulations with 10,000 random inputs, the detailed breakdown of the power dissipation of the different blocks of different 16×16-bit and 32×32-bit array multipliers operating at 1MHz @ 1.1V; their energy dissipation can be found in μW/MHz accordingly. The different blocks include the Booth PPG, Adder Block and Input Drivers (buffers for inputs). For the SB-Dynamic [130] and the proposed multipliers, the 'Overhead' block refers to the power dissipated by the delay circuits to control the Domino DAs and LAs respectively. The T28 adders (see Fig. 3.2 (a) earlier) are used in the Leapfrog [127] and General [120] designs, and are used as the CRA in the proposed design. Note that the total power dissipation of the multipliers is based on the post-layout

simulations but the power breakdown distribution is estimated from pre-layout simulations.

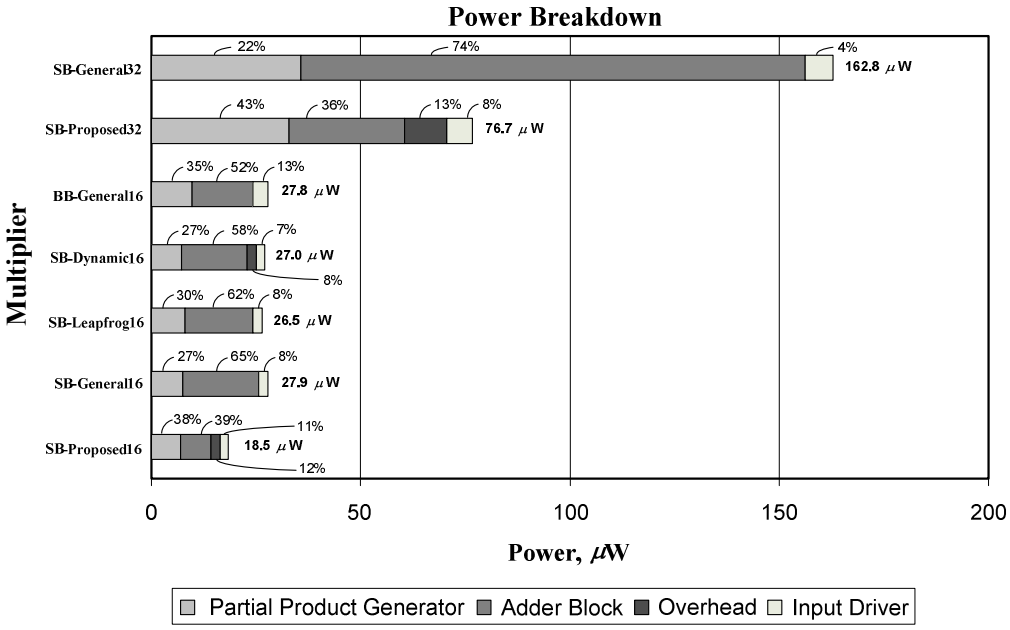

Fig. 4.5 Power breakdown (in % and μW) of different array multipliers based on simulations @ 1.1V, 1MHz

In Fig. 4.5, two different PPGs are considered in the General array multipliers [120]. The first PPG (embodying 12-transitor Booth encoders, see PPG in Fig. 4.3) is the standard Booth PPG and this standard Booth PPG is denoted as 'SB' PGG. The second PPG (embodying 18-transistor Booth encoders [132]) is used to balance (equalize) the arrival times of partial products to the Adder Block, resulting in potentially reduced spurious switching in the Adder Block and this Balanced Booth PPG is denoted as the 'BB' PPG.

From Fig. 4.5, it is noted that although the BB-General16 embodying the BB PPG reduces the power dissipation in the Adder Block (compared to the SB-General16

employing the SB PPG from 65% (18.1μW) to 52% (14.4μW)), the overall power dissipation of the two multipliers is approximately the same. This is because the reduced power in the Adder Block embodied in the BB-General16 is offset by the higher power dissipation in the BB PPG and the input drivers. Furthermore, the BB PPG is more sensitive to *W/L* ratio sizing, parasitic effect, floorplan in layout, and voltage variations. Consequently, design considerations to minimize these undesirable susceptible effects in the BB PPG are critical — the BB PPG may otherwise generate partial products with very different times, much less synchronous than expected. These different arrival times of the inputs consequently result in larger amount of spurious switching in the Adder Block, hence higher power dissipation in the multiplier. Furthermore, the IC area required for the BB PPG is larger (compared to the SB PPG). In view of the potentially higher power dissipation, undesirable added design considerations and larger IC area required of the BB PPG, the BB PPG is not considered and instead only the SB PPG is considered in reported multiplier designs (except BB-General16).

From Fig. 4.5, it is also noted that for the reported SB-Dynamic [130], SB-Leapfrog [127], SB-General [120], and BB-General [120], [132] 16×16-bit multipliers, an average of 59% (16.1μW) of the total power is dissipated in the Adder Block. Put simply, the Adder Block dissipates the largest power in typical 16×16-bit multipliers. By means of the proposed approach to significantly reduce the spurious switching and with little overhead, the Adder Block now dissipates 39% (7.2μW) of the total power. In other words, the proposed design on average dissipates ~ 32% less power than the reported designs compared herein, and

~ 30% less power than the lowest power reported design, the SB-Leapfrog multiplier.

To quantify the degree of reduced spurious switching, Fig. 4.6 depicts the number of switchings per adder in the different rows of the Adder Block based on information obtained from pre-layout simulations by the transistor-level Nanosim simulations. The total number of switching (i.e. changed voltage levels from V_{DD} to ground or vice versa) is first recorded for all the adders in the specific row, and is then averaged by the total number of adders in that row. The average number of switchings per adder for the SB-Leapfrog16, SB-General16, and BB-General16 multipliers is 5.3, 6, and 5.3 respectively, and the amount of switching increases as the row number increases. This is not unexpected as the inputs to the adders become increasingly less synchronous in the latter rows. In the case of the multiplier with Domino DAs (termed SB-Dynamic16), the number of switchings per adder remains largely unchanged, an average of 5.6 switchings per adder. This is because the Domino DAs always first pre-charge and then evaluate when the inputs are ready. The high number switching in the Domino DAs is due to its dynamic operation (constant precharge and evaluation). By comparison, the number of switchings per adder in the proposed design is substantially low at 2 switchings per adder, leading to the much desired lower power attribute. The average 3 switchings per adder (instead of 2) in the last row of the SB-Proposed16 and SB-Proposed32 multipliers is largely due to the carry ripple effect of the multiplication product in the CRAs in the last row.

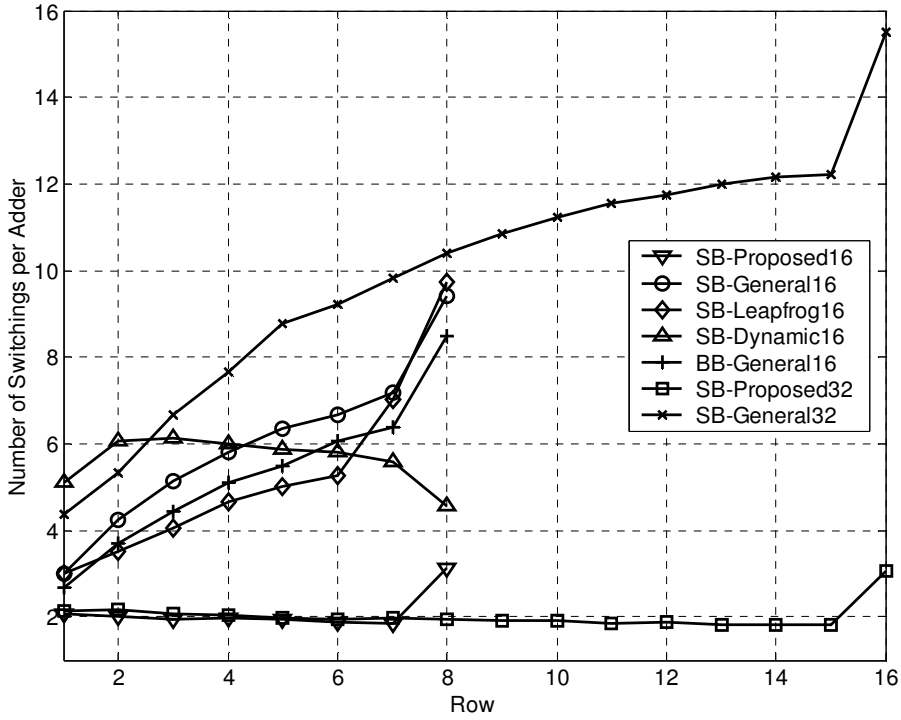

Fig. 4.6 Switching analysis in the Adder Block of various multipliers

The efficacy of the proposed approach is more pronounced in longer wordlength multipliers, for example a 32×32-bit multiplier. This is because there is a larger number of rows and because the input signals to the adders in the latter rows are even less synchronous. Fig. 4.6 also depicts the number of switchings per adder of a General 32×32-bit array multiplier. On average, there are ~ 10 switchings per adder @ 1.1V, 1MHz. By means of the proposed approach, the number of switchings per adder remains approximately the same as the proposed 16×16-bit design – 2.1 switchings per adder. In power terms @ 1.1V, 1MHz from Fig. 4.5, the Adder Block dissipates 74% (120.5μW) and 36% (27.6μW) of the total power of the SB-General multiplier and the proposed multiplier design respectively. In other words, the proposed design dissipates ~ 53% less power than the

SB-General multiplier.

Two 16×16-bit multiplier designs, the proposed SB-Proposed16 and the conventional SB-General16, have been realized in monolithic form using a 0.35μm dual-poly three-metal CMOS process. The microphotographs of these designs are depicted in Fig. 4.7. Fig. 4.8 depicts the test jig for the multiplier ICs. On the basis of measurements on prototype ICs and on simulations, Table 4.1 summarizes the parameters of the different multipliers. The measurement results agree well with the simulations.

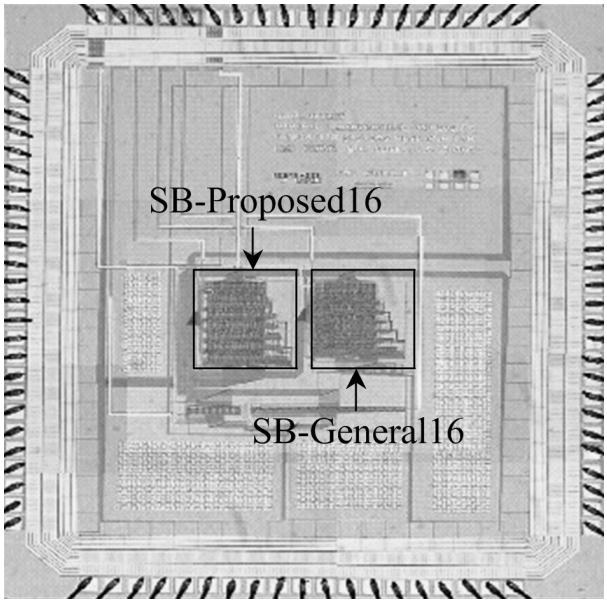

Fig. 4.7 Microphotograph of the proposed multiplier (SB-Proposed16) and the conventional multiplier (SB-General16)

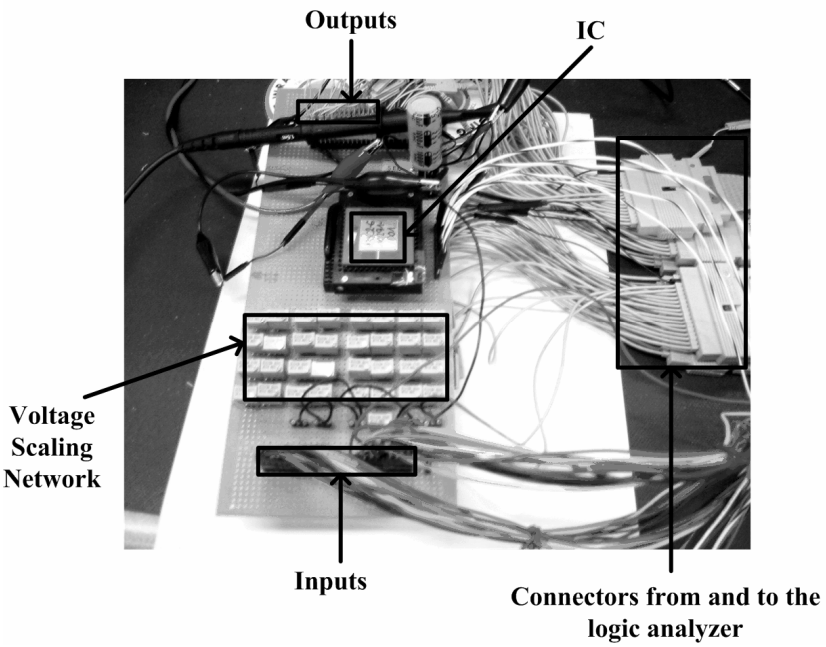

Fig. 4.8 The test jig for the multiplier IC

TABLE 4.1 AVERAGE NUMBER OF SWITCHINGS, ENERGY, DELAY, AREA & ENERGY-DELAY-PRODUCT (EDP) OF VARIOUS ARRAY MULTIPLIERS @ 1.1V, 1MHZ BASED ON POST-LAYOUT SIMULATIONS AND MEASUREMENTS ON PROTOTYPE ICS

Multiplier	Simulations based on Post-Layouts					Measurements based on Prototype ICs		
	Average Switching	Energy (μW/ MHz)	Delay (ns)	Area (mm^2)	EDP (a J.s)	Energy (μW/ MHz)	Delay (ns)	EDP (a J.s)
SB-General16	1412	27.9	104	0.18	2.9	28.3	122	3.5
SB-Leapfrog16	1320	26.5	103	0.17	2.7	—	—	—
SB-Dynamic16	1375	27.0	134	0.24	3.6	—	—	—
BB-General16	1291	27.8	96	0.20	2.7	—	—	—
SB-Proposed16	657	18.5	131	0.20	2.4	18.8	145	2.7
SB-General32	8350	162.8	201	0.45	32.7	—	—	—
SB-Proposed32	2752	76.7	260	0.51	19.9	—	—	—

Of the 16×16-bit designs in Table 4.1, the proposed design features the lowest energy dissipation, one of the slowest designs but the lowest EDP. In some

applications, such as that for the intended hearing aid application where the embodied DSP does not require high speed operations, the delay is inconsequential. The speed of the proposed design can be increased by adopting a more aggressive timing for the delay controlled by the delay elements $D_1 – D_7$ in Fig. 4.3. However, this may compromise the degree of spurious switching reduction, and hence the energy dissipation. The speed can also be increased by replacing the CRAs in the final row of the adder to a faster adder such as a carry look-ahead adder but at the slight cost of increased energy dissipation. For even higher speed operation, V_{DD} would need to be increased, for example from 1.1V to 3.3V – in this case, for the same degree of spurious switching reduction, the delay of our design reduces from 131ns (7.6MHz) to 26ns (38MHz) and the power dissipation remains low at ~ 273μW/MHz. Note that at 3.3V operation, the short-circuit current is no longer negligible, dissipating ~ 30% of the total power. For the other multipliers, the short-circuit current is similarly not negligible. To reduce the short-circuit power dissipation, the adders and delay lines would need to be resized.

For the 32×32-bit multipliers, the comparison had been discussed earlier in Fig. 4.5 and the comments had also been made therein.

In terms of IC area, both the proposed 16×16-bit and 32×32-bit designs are slightly larger than the 16×16-bit SB-General and SB-Leapfrog multipliers, and the 32×32-bit SB-General multiplier. This is due to the overhead of the LAs and the delay circuits. However, the proposed design is smaller than the 16×16-bit SB-Dynamic multiplier and is comparable to the 16×16-bit BB-General multiplier.

In summary, the proposed multiplier core features very low energy dissipation owing to the substantially reduced spurious switching, and is suitable for energy-critical applications including hearing aids.

4.2.3 Proposed Asynchronous-Logic 16×16-bit Control-Multiplier

In the computation of the FFT/IFFT algorithm, a large number of multiplications are trivial due to the coefficients of ×1, × –1, and ×0 [25]. Table 4.2 tabulates the number of trivial multiplications and its % (with respect to the total multiplications, including trivial and non-trivial) for various *N*-point radix-2 FFT algorithms. From Table 4.2, it is noted that for a 128-point FFT, about 42% of the multiplications are trivial. The computation of trivial multiplications can be simply obtained by passing the inputs directly to be the outputs (for × 1), by setting the outputs to be zeros (for × 0), or by negating the inputs (for × –1). In these instances, the effective energy dissipation due to the multiplier can be reduced by having the appropriate control circuits output these trivial multiplication products accordingly. Fig. 4.9 depicts the proposed async 16×16-bit multiplier with the added control circuits (comprising a control logic, a converting circuit, and multiplexers) and the multiplier is termed as Control-Multiplier [57]. The multiplier core design, is based on the earlier multiplier design where the spurious switching/glitch therein is largely eliminated by means of LAs timed by the delay lines, D_1 to D_7.

TABLE 4.2 NUMBER OF TRIVIAL MULTIPLICATIONS AND ITS % IN 8- TO 512-POINT RADIX-2 FFT ALGORITHMS

N	Stage									Total	%
	1	2	3	4	5	6	7	8	9		
8	16	16	8	–	–	–	–	–	–	40	83%
16	32	32	16	8	–	–	–	–	–	88	69%
32	64	64	32	16	8	–	–	–	–	184	58%
64	128	128	64	32	16	8	–	–	–	376	49%
128	256	256	128	64	32	16	8	–	–	760	42%
256	512	512	256	128	64	32	16	8	–	1528	37%
512	1024	1024	512	256	128	64	32	16	8	3064	33%

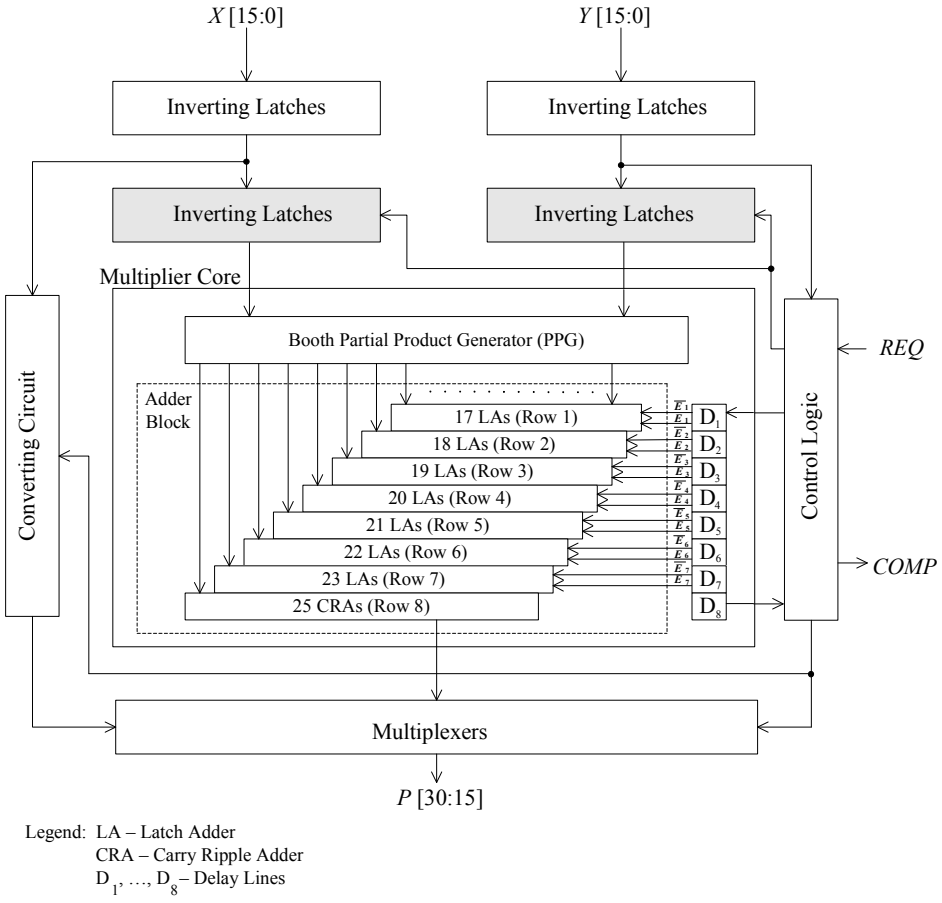

Fig. 4.9 Block diagram of the Control-Multiplier

The inputs are the multiplication *X* (input signals) and the multiplier *Y* (coefficient of the twiddle factor). In Fig. 4.9, the shaded inverting latches block the inputs to the multiplier core in cases of trivial multiplications. For non-trivial multiplications, the shaded inverting latches will be updated with the new data to the multiplier core for multiplications. The control circuits determine if the multiplications are trivial or otherwise, initiate multiplications when *REQ* is asserted and generate *COMP* upon the completion of a multiplication. The *COMP* signal is matched (by means of delay lines) to the worst-case delay for either one of two situations – either for the trivial multiplications or the non-trivial multiplications.

Fig. 4.10 depicts the microphotograph of the Control-Multiplier (part of the FFT/IFFT prototype IC) using a 0.35μm CMOS dual-poly four-metal CMOS process. Table 4.3 tabulates the simulated energy dissipation and IC area for the Control-Multiplier and the proposed prototype multiplier core (see Fig. 4.7). For a fair comparison, the same proposed multiplier core (in Table 4.3) is re-implemented using the same 0.35μm CMOS dual-poly four-metal CMOS process (instead of the previous 0.35μm CMOS dual-poly three-metal CMOS process). Compared to the previous prototype multiplier core, the re-implemented proposed multiplier core now features lower energy dissipation and smaller area (due to the lower power adder cells employed, the availability of the additional metal layer and better floor planning). Based on simulations with 40% (estimated % of the actual multiplications) of the multiplications being trivial, the Control-Multiplier features ~ 22% lower energy dissipation. The Control-Multiplier, however, requires about 7% larger IC area than the multiplier core.

For completeness, the delay of the Control-Multiplier is ~ 60ns @ 1.1V for trivial multiplications and ~ 160ns @ 1.1V for non-trivial multiplications respectively.

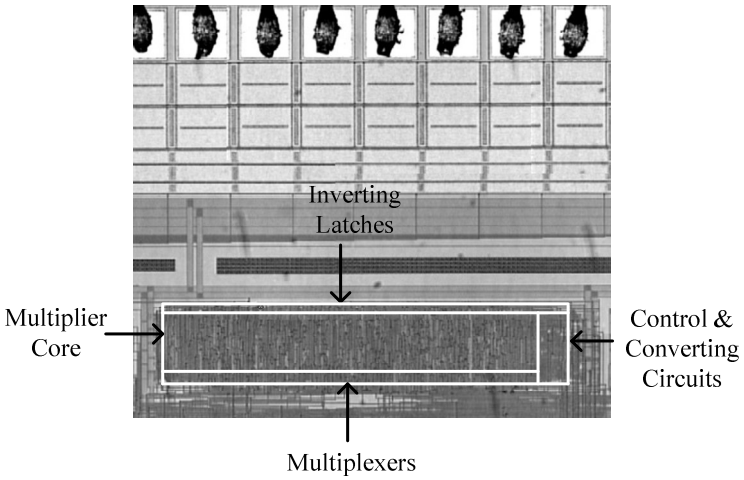

Fig. 4.10 Microphotograph of the Control-Multiplier (part of the FFT/IFFT prototype IC)

TABLE 4.3 THE ENERGY AND IC AREA FOR THE CONTROL-MULTIPLIER AND THE MULTIPLIER CORES @ 1.1V, 1 MHZ

	Energy (μW/MHz)	Area (mm^2)
Control-Multiplier$^+$	14.1	0.134
Multiplier core$^+$ (redesigned)	18.0	0.125
Multiplier core* (Fig. 4.7)	18.5	0.200

+ dual-poly four-metal CMOS; * dual-poly three-metal CMOS

In short, for FFT/IFFT algorithms where the N is small (e.g. < 512-point), the realization of the Control-Multiplier is worthwhile due to its low energy dissipation attribute.

4.3 Proposed Asynchronous-Logic 128×16-bit Memory

Fig. 4.11 depicts the block diagram of the proposed 128×16-bit single-port async memory macrocell [57].

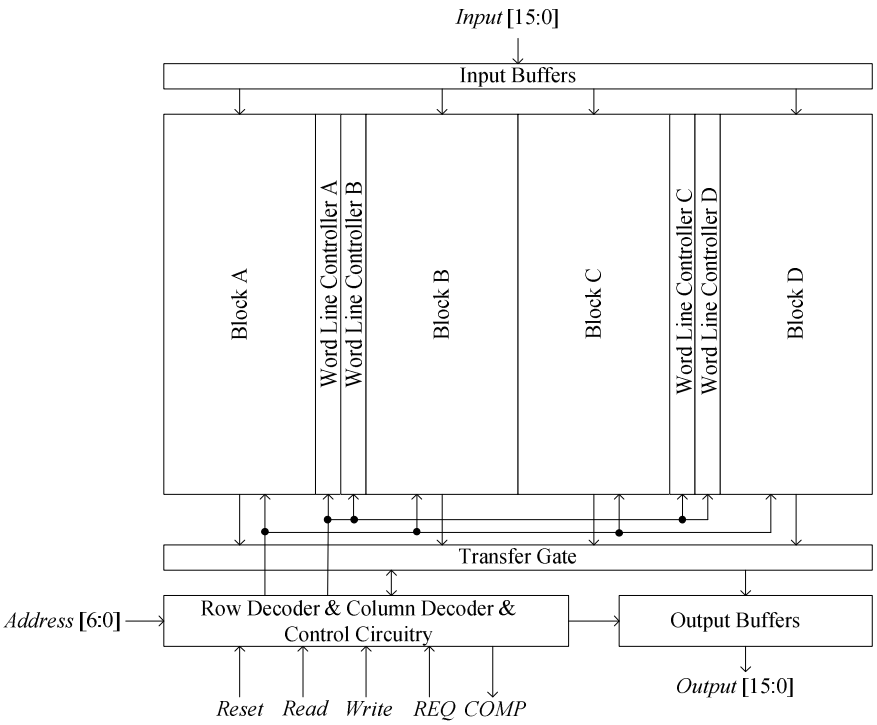

Fig. 4.11 Block diagram of the 128×16-bit single-port async memory

For low energy dissipation, the partitioning approach (banking) [20], [21] is adopted to reduce the capacitance. In this approach, the memory is sub-divided into four 32×16-bit sub-memory blocks and these sub-memory blocks are controlled by respective proposed Word Line Controllers. The Row Decoder, Column Decoder, and Control Circuitry control the memory for the write and read accesses. For the read access, the stored data are retrieved (via the Transfer Gate from one of the four sub-memory blocks) to the output buffers, according to the address bus, A[6:0]. For the write access, the new input is written (from the input

buffers) into one of the four sub-memory blocks, according to the address bus, *A*[6:0]. The *REQ* and *COMP* signals respectively indicate the request and completion signals of the memory.

For completeness, Figs. 4.12 to 4.16 depict the schematic diagrams of the different blocks of the async memory macrocell, including the 32×16-bit sub-memory block, Row Decoder and Word Line Controller, Column Decoder, Transfer Gate, and Input Buffers. Except the Word Line Controller, the design and operations of the abovementioned blocks are well-established [20], [21].

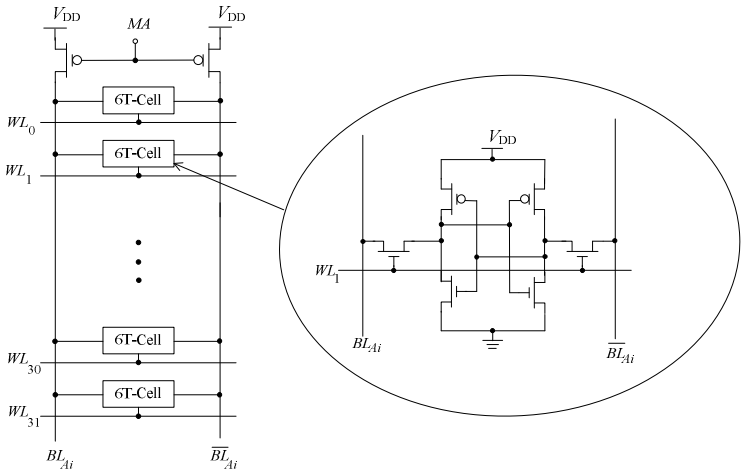

Fig. 4.12 Circuit schematic of the storage cells for one column of a 32×16-bit sub-memory block; 16 columns required for one sub-memory block

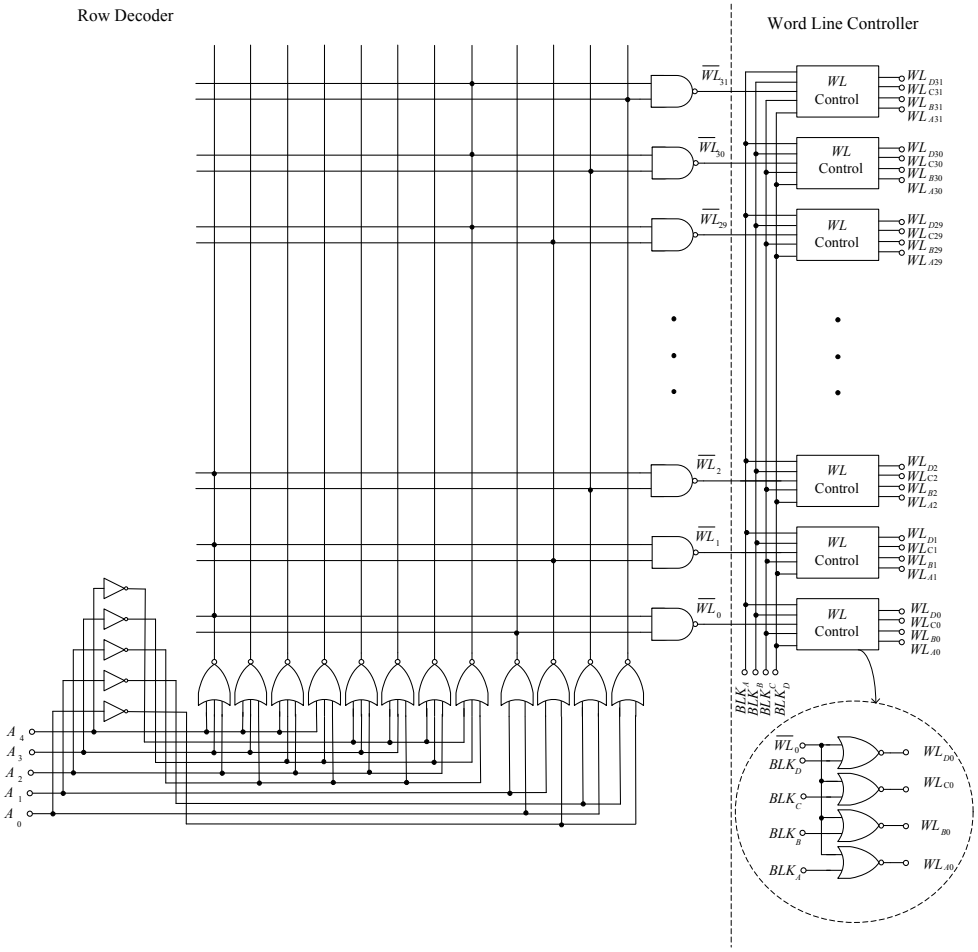

Fig. 4.13 Circuit schematics of the Row Decoder and the Word Line Controller

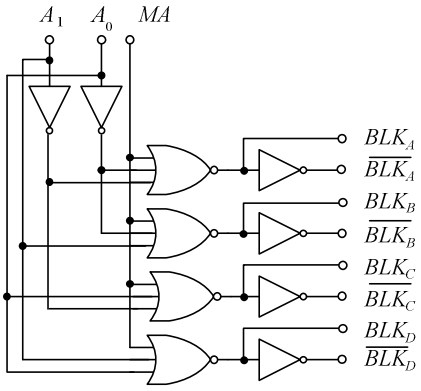

Fig. 4.14 Circuit schematic of the Column Decoder

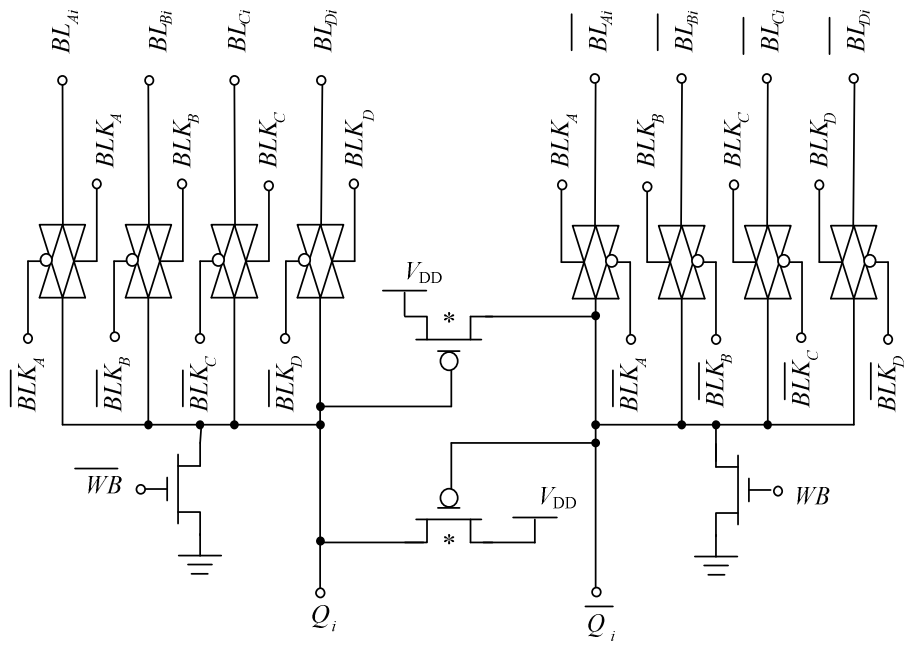

Fig. 4.15 Circuit schematic of the Transfer Gate

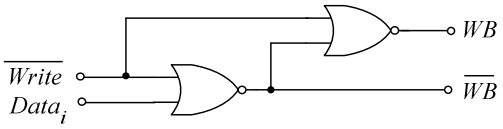

Fig. 4.16 Circuit schematic of the Input Buffer

It is interesting to note that the Word Line Controller (see Fig. 4.13) is proposed to activate the particular sub-memory block of interest and to prevent multiple assertions of unselected sub-memory blocks, resulting in reduced power/energy dissipation. The Word Line Control is simple but novel, and it is simply a gating design technique. Nevertheless, the adoption of the Word Line Control prevents the redundant assertions of the word lines, in particular due to the multiple changes (if any) of the address lines, A_6 to A_0, from one address to the desired address (during transients). The control signals, BLK_A to BLK_D, are determined by the two addresses, A_1 and A_0, from the Column Decoder. One of the control signals (amongst BLK_A to BLK_D) will only be enabled when the Memory Access

(*MA*) signal is '1' (indicating all the address lines are ready). In the case of no memory operation (*MA* = '0'), all the control signals, BLK_A to BLK_D, are '0', hence the word lines are not asserted, including the instances when the address lines change.

Fig. 4.17 depicts the completion detection circuit (control circuitry) in the async 128×16-bit memory macrocell. The bit lines are used for completion detection as they are always initialized to the pre-charged state for every new operation; this completion detection is similar to the dual-rail encoding described in Chapter 2 earlier. However, for low energy consideration, only the most significant bit (instead of all bits) of each sub-memory block is used. This is based on the assumption that the delay of the write operation (or the read operation) is similar for other bits. The inverter delay chain is used for additional delay (safety) margin and to match the delay from the bit lines to the output latches (for the read access).

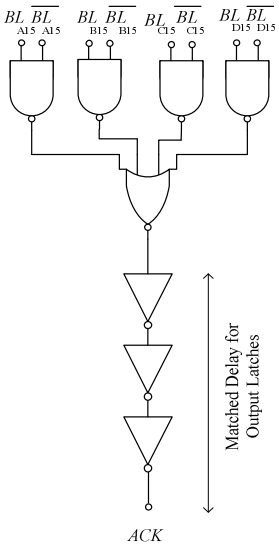

Fig. 4.17 Circuit schematic of the completion detection circuit for the memory macrocell

Fig. 4.18 depicts the async 128×16-bit memory macrocells embodied in the async FFT/IFFT processor IC. Pearl scripts are developed to floorplan the four sub-memory blocks and the Word Line Controllers are inserted next to their respective sub-memory block for short interconnections. The entire async memory macrocell occupies the area of 410μm × 346.4 μm (or 0.142mm^2) @ dual-poly four-metal CMOS process.

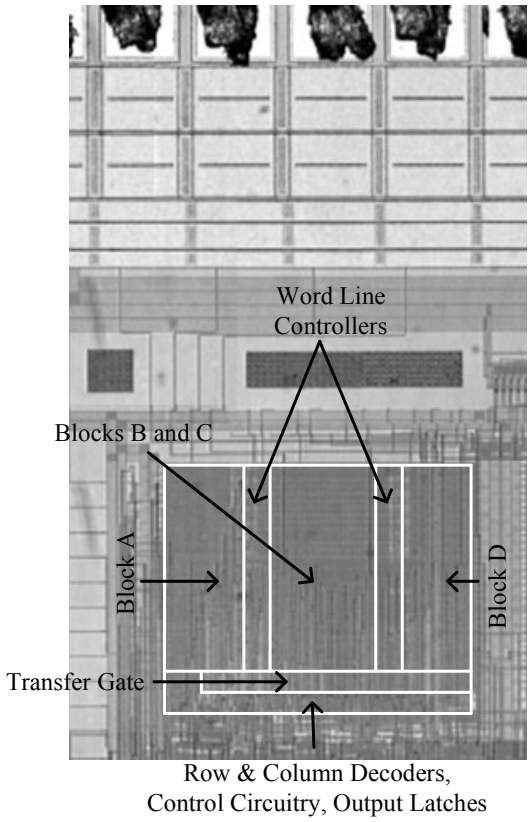

Fig. 4.18 The async 128×16-bit memory macrocell (part of the FFT/IFFT prototype IC)

Table 4.4 tabulates the characteristics of the async 128×16-bit memory macrocell based on the post-layout computer simulations @ 1.1V, 1MHz. The read or write access time is defined as the time from the *REQ* signal being asserted until the

ACK signal is asserted. The read and write access times of the memory macrocell are 60ns and 40ns respectively, and the energy dissipation of the memory macrocell is 6μW/MHz.

TABLE 4.4 THE CHARACTERISTICS OF THE ASYNC 128×16-BIT MEMORY MACROCELL @ 1.1V, 1MHZ

	Characteristics
Supply Voltage	1.1V
Energy	6μW/MHz
Read Access Time	60ns
Write Access Time	40ns
Area	0.142mm^2

4.4 Summary

Three proposed macrocells, the 16×16-bit Booth array-based multiplier core, the async Control-Multiplier specifically for the FFT/IFFT algorithm, and the 128×16-bit memory macrocell, have been described.

The proposed 16×16-bit Booth array-based multiplier core featured substantially reduced spurious switching, and hence lower energy dissipation. The low spurious switching attribute was obtained by replacing the usual full adders with the proposed LAs in the Adder Block and by synchronizing their inputs by means of delay line circuits. It has been shown that the resulting switching activity in the Adder Block was reduced to only ~ 38% and ~ 21% of reported 16×16-bit and 32×32-bit designs respectively (translating to ~ 32% and ~ 53% energy

reduction). Although the minor penalty was an increased (but inconsequential) delay, the proposed design still featured the lowest EDP. The prototype 16×16-bit multiplier IC dissipated 18.8μW/MHz @ 1.1V, featured a delay of 122ns @ 1.1V and the IC area was 0.2mm^2 @ 0.35μm dual-poly three-metal CMOS process.

The proposed Control-Multiplier embodied the proposed 16×16-bit Booth array-based multiplier core and added control circuits, and featured further reduced energy dissipation (compared to the multiplier core). This was obtained by using the proposed 16×16-bit Booth array-based multiplier core for non-trivial multiplications and the (lower energy) control circuits for trivial multiplications, and because ~ 42% of the multiplications for an FFT/IFFT computation were trivial. The Control-Multiplier dissipated 14.1μW/MHz (~ 22% lower energy compared to the proposed multiplier core), the delay was ~ 60ns for trivial multiplications and ~ 160ns for non-trivial multiplications, and the IC area was 0.134mm^2 @ 0.35μm dual-poly four-metal CMOS process.

The async 128×16-bit memory macrocell featured low energy dissipation, obtained by the adoption of well-established memory design techniques and by the proposed Word Line Controllers that activated only the selected sub-memory block (the unselected sub-memory blocks remained inactivated). The memory macrocell dissipated 6μW/MHz @ 1.1V, featured delays of 40ns and 60ns for the write access and read access respectively, and the IC area was 0.142mm^2 @ 0.35μm dual-poly four-metal CMOS process.

Chapter 5

Synchronous-Logic and Asynchronous-Logic FFT/IFFT Processors

5.1 Introduction

This chapter describes a sync 128-point 16-bit DIT FFT/IFFT processor and a proposed async 128-point 16-bit DIT FFT/IFFT processor for energy-critical applications, including hearing aids. The sync design serves as a benchmark as the sync approach is the current prevalent approach. A portion of this chapter is extracted from the author's paper that has been accepted as a regular paper in the *IEEE J. Solid-State Circuits* [57].

The FFT/IFFT algorithm is one of the most prolific algorithms in digital signal processing. A review of the FFT/IFFT algorithm was given in Chapter 2. In this chapter, the architecture of the proposed async FFT/IFFT processor is somewhat similar to that of the sync FFT/IFFT processor but there are some differences in an attempt to exploit specific advantages of the async approach.

To demonstrate the efficiency of the async approach over the sync approach, the async FFT/IFFT design is benchmarked against its sync counterpart in terms of energy, delay, and IC area. Both designs are realized with the same functionality and similar architecture, and fabricated using the same 0.35μm CMOS process technology. Two approaches for the sync design are considered. In the first

approach, the clock is allowed to run continuously. It will be shown that the async design dissipates as much as ~ 50% lower energy than this first sync design approach. In the second approach, the well established clock gating approach is adopted (idle state) to reduce energy dissipation when computation is not required. It will be shown that the async design dissipates ~ 37% lower energy than this second sync design approach. However, the minor drawbacks of the async design are the required larger IC area, by 10%, and an inconsequential 1.4 times average worse delay (from 1.1V to 1.4V) – if the matched delay elements therein are better tuned, the delay is shorter, 0.6 times average worse delay and without much power/energy penalty; the delay is inconsequential because of the low speed (< 5MHz) hearing aid application. The lower energy attribute is achieved largely by means of the async approach embodying established and proposed async circuit designs, including: (i) reduced redundant (spurious) operations, (ii) energy-efficient async data path circuits (e.g. the proposed Control-Multiplier and memory macrocells described in Chapter 4), and (iii) the use of simple latches (as opposed to flip-flops in sync designs). This design is yet another demonstration of how the async approach may be a good alternative to the prevalent sync approach.

This chapter is arranged in the following manner. Section 5.2 describes the sync FFT/IFFT processor and Section 5.3 describes the proposed async FFT/IFFT processor. Section 5.4 provides measurements and comparisons of the sync and async FFT/IFFT processors. Finally, Section 5.5 summarizes this chapter.

5.2 Synchronous-Logic FFT/IFFT Processor

Fig. 5.1 depicts the block diagram of the sync FFT/IFFT processor [30], and the main functional blocks are a 128×32-bit SRAM memory, two shifters, two multipliers, three adders/subtractors, write back registers, a ROM, and controllers.

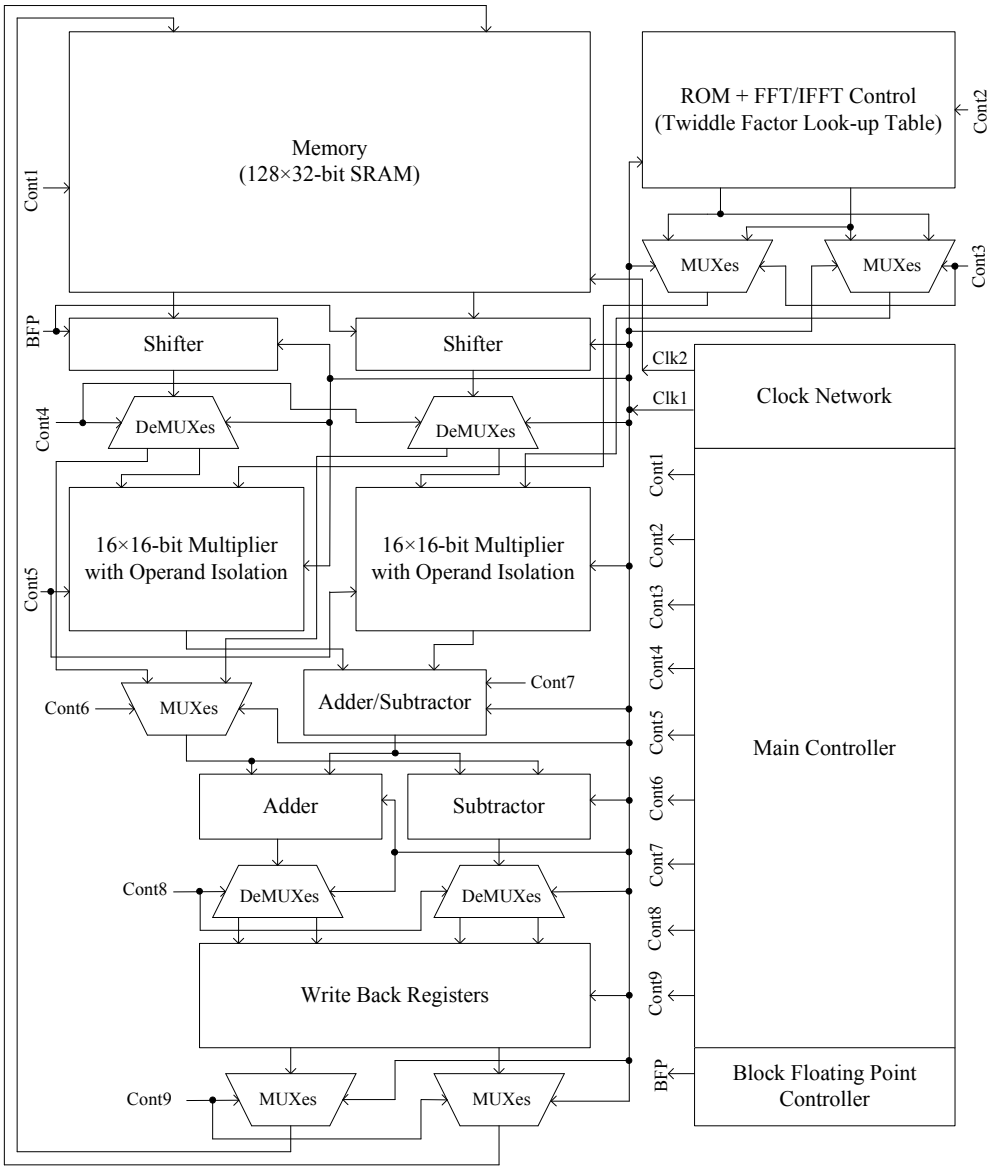

Fig. 5.1 Block diagram of the sync FFT/IFFT processor

There are two clock signals, Clk1 and Clk2, (a multiple-clock approach, see Section 2.2.1) in the sync FFT/IFFT processor. The frequency of Clk2 is 2× Clk1, and is used to trigger the memory to enable the availability of two memory data for every Clk1 cycle. Clk1 is used as the system clock for the remaining modules.

The FFT/IFFT processor requires two data flows to compute each butterfly operation as depicted in Figs. 5.2 (a) and (b). Data flows in Figs. 5.2 (a) and (b) respectively compute the real and imaginary outputs. Each data flow is performed within five operation sequences as shown in Fig. 5.3. In Memory Read, the inputs are read and in Scaling the inputs are scaled (where necessary) to avoid overflow. In Multiplications, two multiplications are performed and in Additions/Subtractions, three additions/subtractions are performed. Finally in Write Back, the outputs are stored.

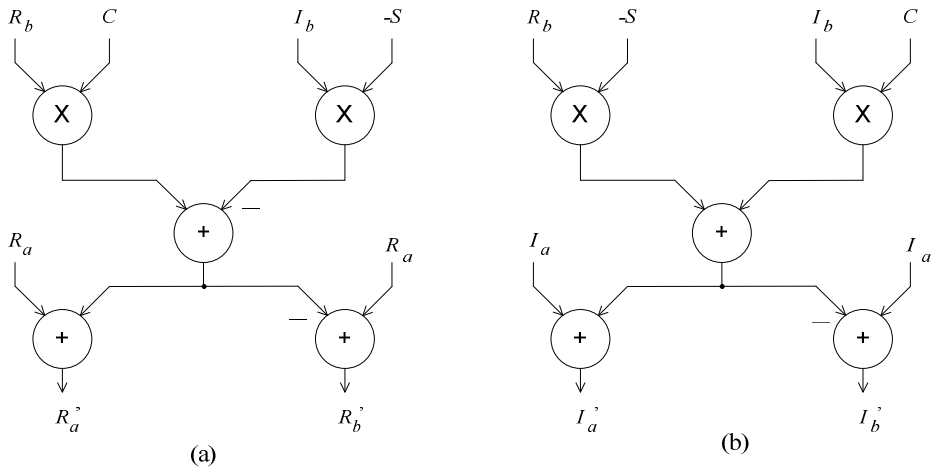

Fig. 5.2 Data flows for each butterfly operation: (a) to compute the real outputs, and (b) to compute the imaginary outputs

Memory Read	Scaling	Multiplications	Additions/ Subtractions	Write Back

Fig. 5.3 Pipeline sequences for each data flow

To reduce the power/energy dissipation in the sync FFT/IFFT processor, three approaches are applied. First, the sync FFT/IFFT processor is controlled in an *ad hoc* manner (including with the multiple-clock approach) and the flip-flops (for unused modules) are disabled for the reduced redundant operations. Second, the power-hungry multipliers are disabled (by means of operand isolation [72]) when trivial multiplications occur ($\times 1$, $\times -1$ and $\times 0$). This approach is worthwhile because about 42% of multiplications are trivial in a 128-point FFT algorithm. Third, low power digital cells are employed where most of the digital cells used are the lowest power circuits among the available cells in the standard (commercial) cell library.

In addition, the clock gating approach is also adopted. From a system perspective, in the case of async designs, 'gating' is innate and inherently controlled at 'fine-grain' gating. In the clock-gated sync design, on the other hand, 'clock gating' is 'course-grain' and can be made 'fine-grain' but with deliberate design effort. In this design, the sync FFT/IFFT design utilizes 2 clocks, Clk1 and Clk2 where the frequency of Clk2 is 2 times Clk1. Clk1 serves as the clock for data path modules (multipliers, adders/subtractors and shifters) and Clk2 for the memory. One complete FFT/IFFT computation (including initialization and synchronization) requires 959 Clk1 and 2430 Clk2 cycles and the coarse-grain clock gating blocks the remaining clock cycles. For example, the time duration for 64 samples @ 16KHz sampling rate is 4ms. For one FFT/IFFT computation

in a typical hearing aid with a 1MHz Clk1 clock frequency (Clk2 is 2MHz), by means of coarse-grain clock gating, 3041 (of 4000) Clk1 and 5570 (of 8000) Clk2 cycles are blocked. In short, the coarse-grain clock gating blocks ~ 76% and ~ 70% of Clk1 and Clk2 respectively. If the clock frequency increases, more unused clock cycles will be gated.

If fine-grain clock gating is instead employed, some small % of Clk1 and Clk2 may be further gated and this is during the initialization and synchronization phases. Specifically, 63 (6.6%) out of 959 Clk1 cycles and 126 (5.2%) out of 2430 Clk2 cycles may be further gated and collectively this translates to a low 6% improvement in energy dissipation. Put simply, should fine-grain clock gating be employed, the degree of reduction in the number of clocks (and energy) is small and the added costs incurred would largely defeat any advantages gained. The costs include a design of higher complexity including consideration for synchronization issues, skew problems and race conditions. Further, fine-grain clock gating by gating the input of all flip-flops may not necessarily save power/energy dissipation [140]. Note that the data path modules in the sync circuits are in fact highly pipelined, thereby resulting in useful computation in most of the active clock cycles.

For completeness, Section 5.4 will delineate the measurement results with and without the clock gating approach for the sync FFT/IFFT processor.

5.3 Proposed Asynchronous-Logic FFT/IFFT Processor

Fig. 5.4 depicts the block diagram of the async FFT/IFFT processor. There are 7 modules, 5 data path modules and 2 control modules, in the async FFT/IFFT processor. They are the Memory Data Path, Shifter Data Path, Multiplier Data Path, Adder Data Path, Write Back Data Path, Async FFT/IFFT Controller and Pulse Circuit. The 5 data path modules embody their respective async HCCs for asynchronous operations, and these data paths execute the butterfly operations within the FFT/IFFT processor. The Async FFT/IFFT Controller pipelines the sequence of operations within the processor and the Pulse Circuit generates a local 'clock' to synchronize the events according to the handshake and control signals.

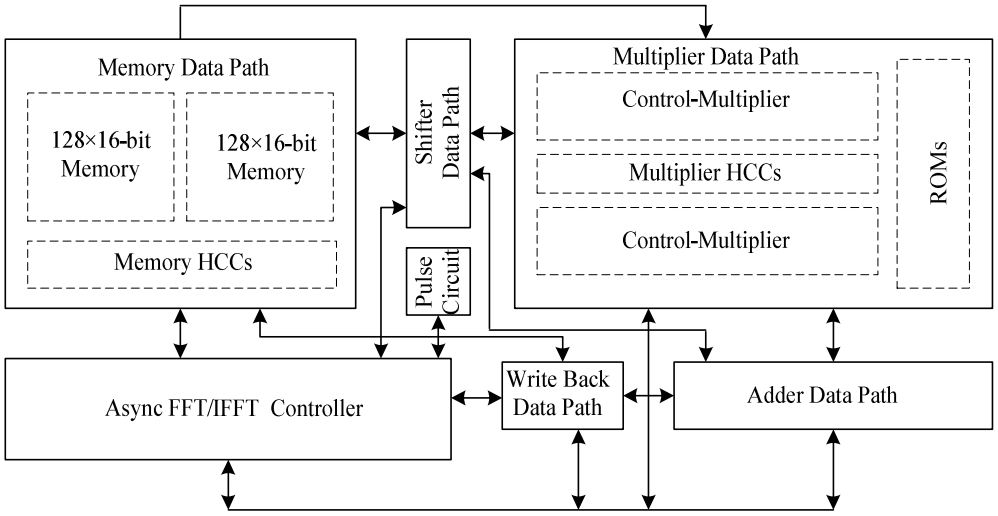

Fig. 5.4 The block diagram of the async FFT/IFFT processor

Fig. 5.5 depicts the interface signals for the async FFT/IFFT processor, particularly for the two main portions, the control portion (Async FFT/IFFT Controller and Pulse Circuit) and the data path portion (five data paths). These

signals include the I/O signals to interface to the external environment (refer to Table 5.1 later), and the interface signals between the control and data path portions. In Fig. 5.5, the signal *Ackin_Adder* is shared between Shifter and Multiplier Data Paths, and the signal *MuxData* is shared between the Multiplier and Adder Data Paths.

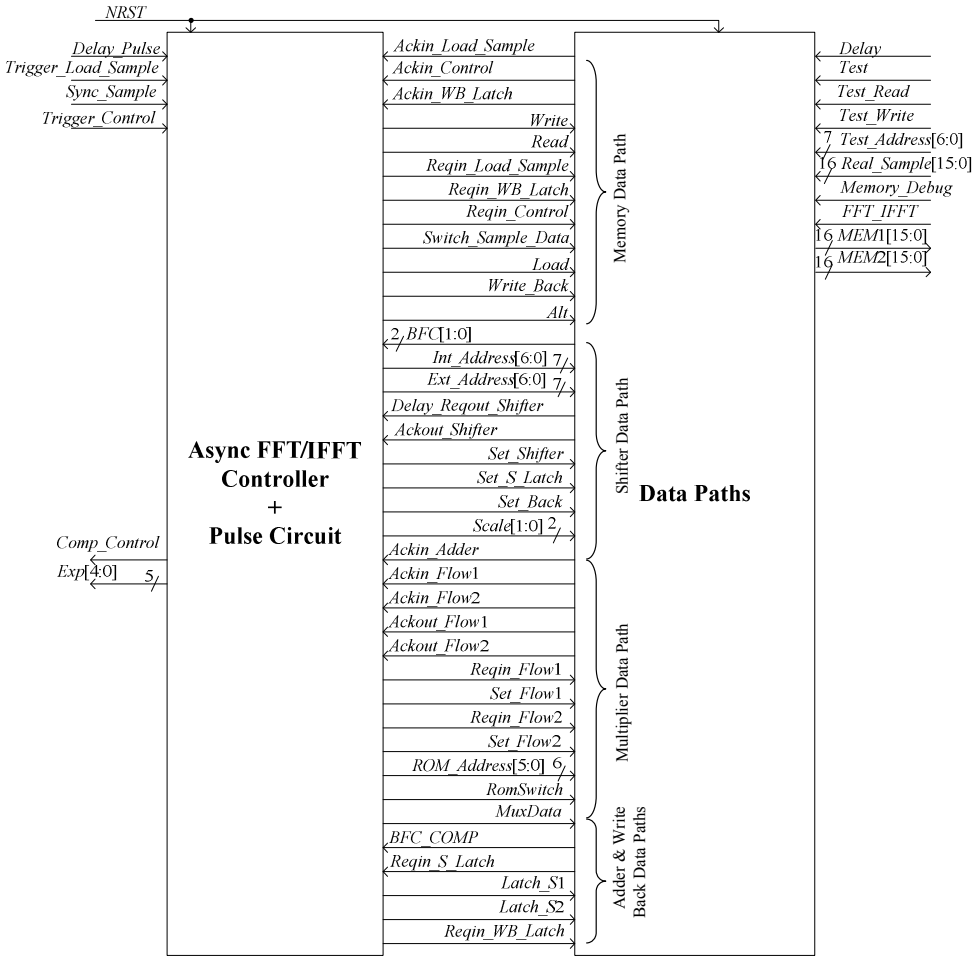

Fig. 5.5 The interface signals in the async FFT/IFFT processor

TABLE 5.1 THE TOP-LEVEL SIGNALS AND FUNCTIONS OF THE ASYNC FFT/IFFT PROCESSOR

Signal	Function
NRST	To reset the system
Delay_Pulse	To increase the delay of the pulse
Trigger_Load_Sample	To load the new samples
Sync_Sample	To synchronize the new samples
Trigger_Control	To start the control portion
Delay	To increase the delay of the data paths
Test	To test the circuits
Test_Read	To test the read access function in the memory
Test_Write	To test the write access function in the memory
Test_Address[6:0]	To generate the address for testing memory
Real_Sample[15:0]	New input sample
Memory_Debug	To control the memory for debugging
FFT_IFFT	To indicate for either FFT or IFFT operation
Comp_Control	To indicate the completion of the controller
Exp[4:0]	The exponent value after the block floating point scaling
*Mem*1[15:0]	The real output
*Mem*2[15:0]	The imaginary output

An overview of the operation of the async FFT/IFFT processor is now described. The new input samples, *Real_Sample*[15:0], are first written into the memory by triggering the *Trigger_Load_Sample* signal. The *Sync_Sample* signal is used to synchronize the new samples (when needed) into the memory according to the data input rate. After loading the 128 samples, the async FFT/IFFT processor is ready for the butterfly operations. When the *Trigger_Control* signal is asserted, the async FFT/IFFT processor performs the butterfly operations, depending on the *FFT_IFFT* signal (‘0’ for FFT process; ‘1’ for IFFT process). Once the 128-point FFT or IFFT process is computed, the *Comp_Control* signal will be

activated. The *Test*, *Memory_Debug*, *Test_Address*, *Test_Write*, and *Test_Read* signals are used to test the data stored in memory via output signals *Mem*1[15:0] and *Mem*2[15:0]. The *Exp*[4:0] signals indicate how many shifts have been performed in the FFT or IFFT process for overflow prevention. The *Delay* and *Delay_Pulse* signals are used to increase the internal delay (optional) for extra delay safety margin. The *NRST* signal is used to reset the async FFT/IFFT processor. For simplicity, the *NRST* (if any) will not be shown in the subsequent figures.

Fig. 5.6 depicts the architecture of the FFT/IFFT control portion. The Main Sequence Controller, Sample Loading Controller, Data Loading Controller, Butterfly Unit Operation Controller, Write Back Controller and Block Floating Point Controller are collectively grouped as the Async FFT/IFFT Controller. This Async FFT/IFFT Controller controls the respective data path circuits according to the control and handshake signals. The Main Sequence Controller pipelines the FFT/IFFT sequences and addresses for the other modules. The Sample Loading Controller pipelines the new input data into the memory. The Data Loading Controller pipelines the stored data that are retrieved from the memory. The Butterfly Unit Operation Controller coordinates the butterfly operations within the FFT/IFFT processor. The Write Back Controller pipelines the outputs that are written back into the memory. The Block Floating Point Controller executes the necessary scaling for each stage.

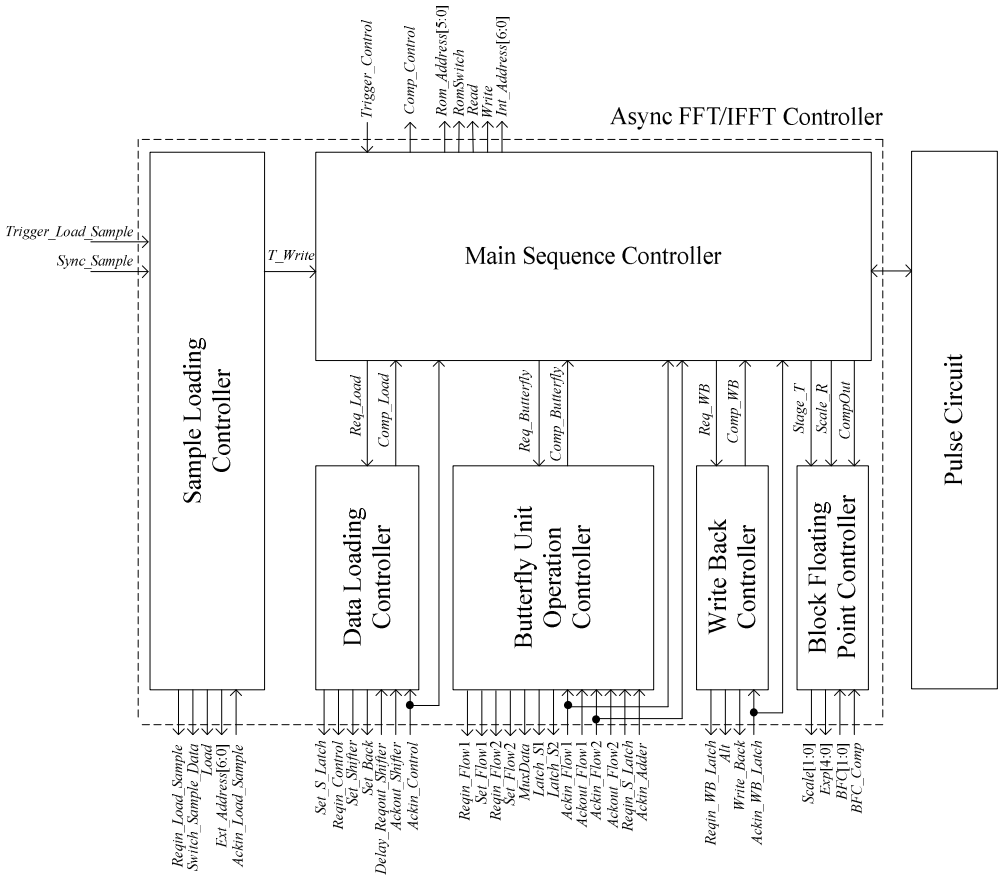

Fig. 5.6 Architecture of the control portion in the FFT/IFFT processor

Fig. 5.7 depicts the architecture of the data paths in the async FFT/IFFT processor. The Memory Data Path reads and writes data during the FFT/IFFT operations. The Shifter Data Path is for conditional block floating point scaling to retain the dynamic range of the signals, avoiding overflow – the signals therein are scaled (in the next stage) when overflow is detected in the Write Back Data Path (in the present stage). The Multiplier Data Path is used to multiply the coefficients (from the ROM) with the signals from the Shifter Data Path. The Adder Data Path performs the additions and subtractions, and finally the Write Back Data Path stores and writes the outputs back to the Memory Data Path.

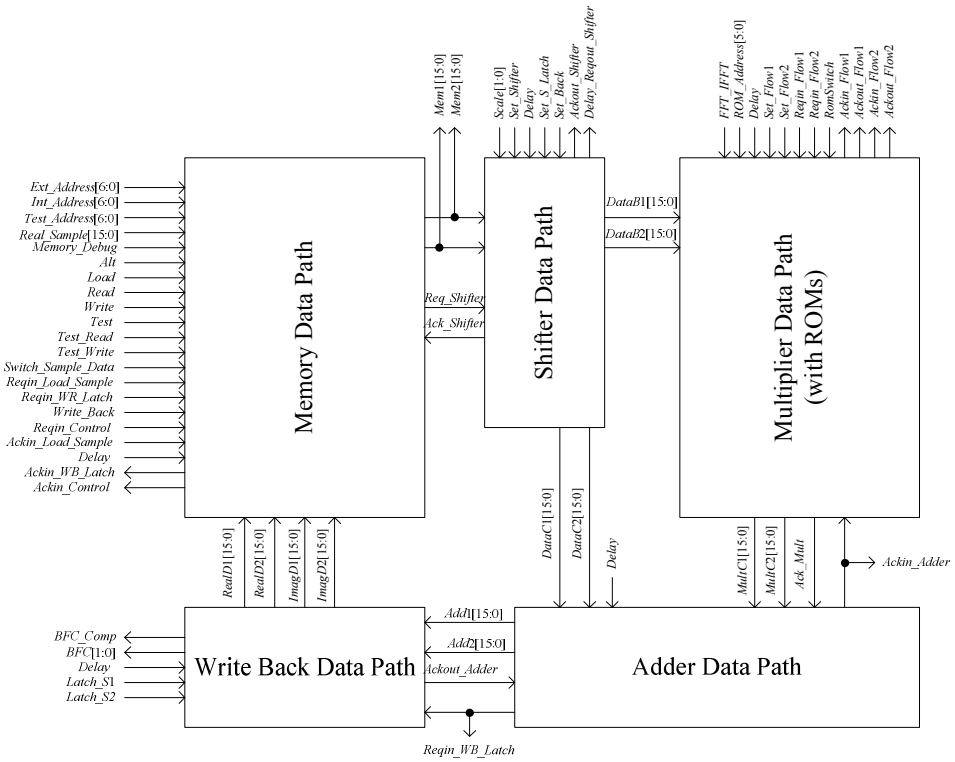

Fig. 5.7 Architecture of the data paths in the async FFT/IFFT processor

To depict the operations of the async FFT/IFFT processor more simply, consider its simplified STGs in Figs. 5.8 to 5.11. The solid lines indicate the direct relationship between the signals and the dashed lines indicate indirect relationship between the signals (e.g. via some circuits that depend on other signals). The dots indicate the initial condition for operations.

Fig. 5.8 depicts the signal flows for loading new samples into the memory, initiated by triggering the *Trigger_Load_Sample* signal. The *Switch_Sample_Data* signal is used to perform the bit reversal process [25] for new data. The pulse-like signal, *Sync_Sample*, controls the rate of input data. This loading process stops when 128 new samples are written into the memory in bit reverse format.

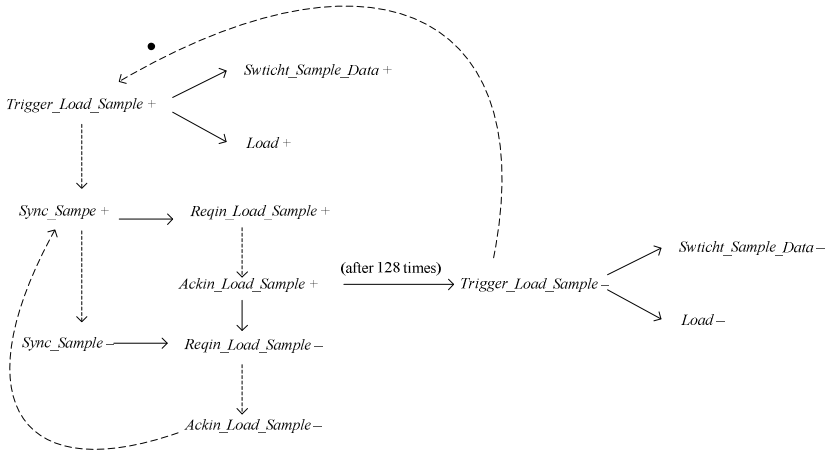

Fig. 5.8 The simplified signal transition graph for loading new samples

Fig. 5.9 depicts the signal flows for loading the data from the memory for a butterfly operation, initiated by triggering the *Trigger_Control* signal. The *Set_Shifter* signal enables the signal flows in the Shifter Data Path and two set of data are thereafter read – one is stored in the temporary latches (by enabling the *Set_S_Latch* signal) and the other at the shifters. For the second set of data, the *Set_Back* signal provides a feedback path to the Memory Data Path. After the two data are retrieved for each butterfly operation, the *Comp_Load* signal is triggered.

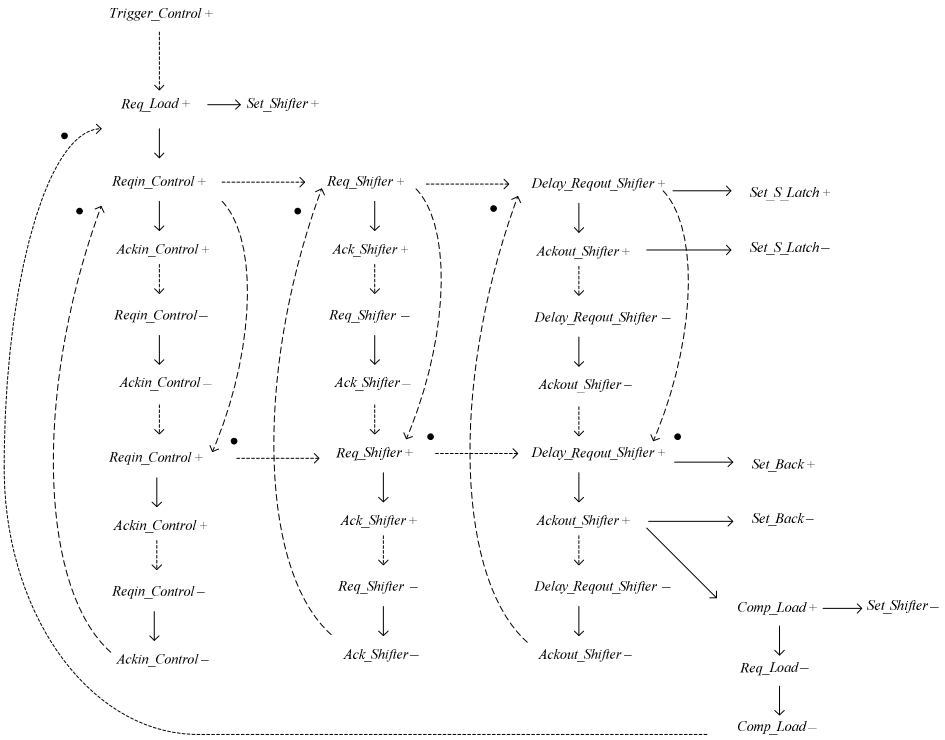

Fig. 5.9 The simplified signal transition graph for loading data for butterfly operations

Fig. 5.10 depicts the signal flows for one butterfly operation, initiated when the *Comp_Load* signal is received. Two data flows (*xxx_Flow*1 and *xxx_Flow*2 where *xxx* indicates the prefix labels for various signals) compute the multiplications and additions/subtractions. The *MuxData* signal swaps the sine and cosine coefficients alternately in these two data flows. The *Latch_S*1 and *Latch_S*2 signals enable the temporary latches for storing the outputs. The *BFC_Comp* signal monitors the overflow signal (where necessary in each computation). After a butterfly operation is computed, the *Comp_Butterfly* signal will be triggered.

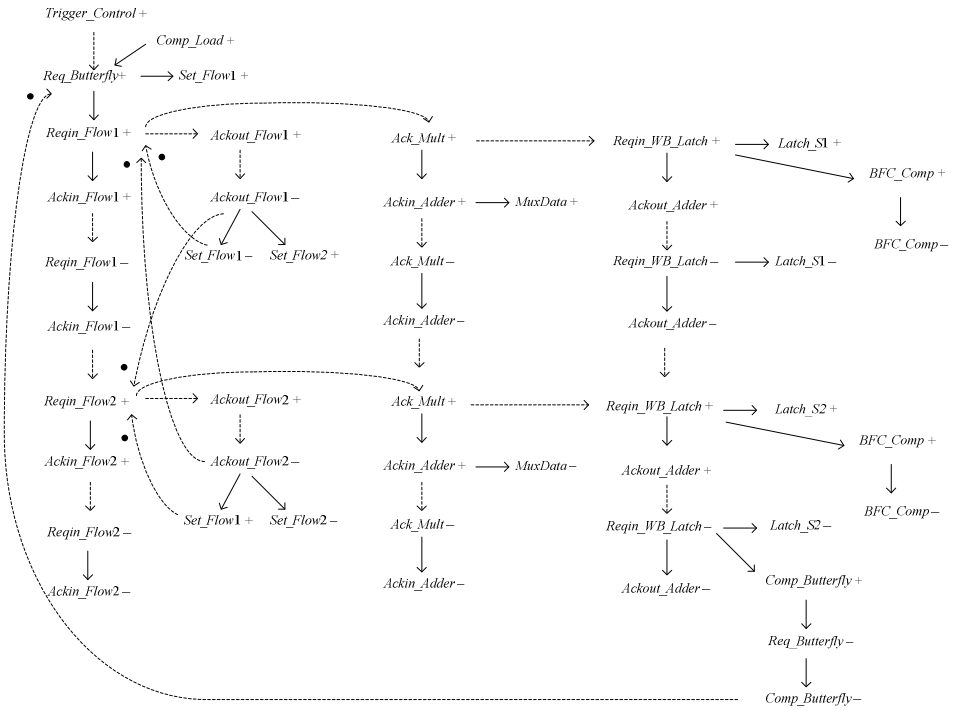

Fig. 5.10 The simplified signal transition graph for executing a butterfly operation

Fig. 5.11 describes the signal flows for writing the temporary outputs (from latches) into the memory, initiated by the *Comp_Butterfly* signal. When all the butterfly operations have been computed, the *Comp_Control* signal will be triggered, and it will in turn negate the *Trigger_Control* signal. At this time, one computation of FFT or IFFT process is complete.

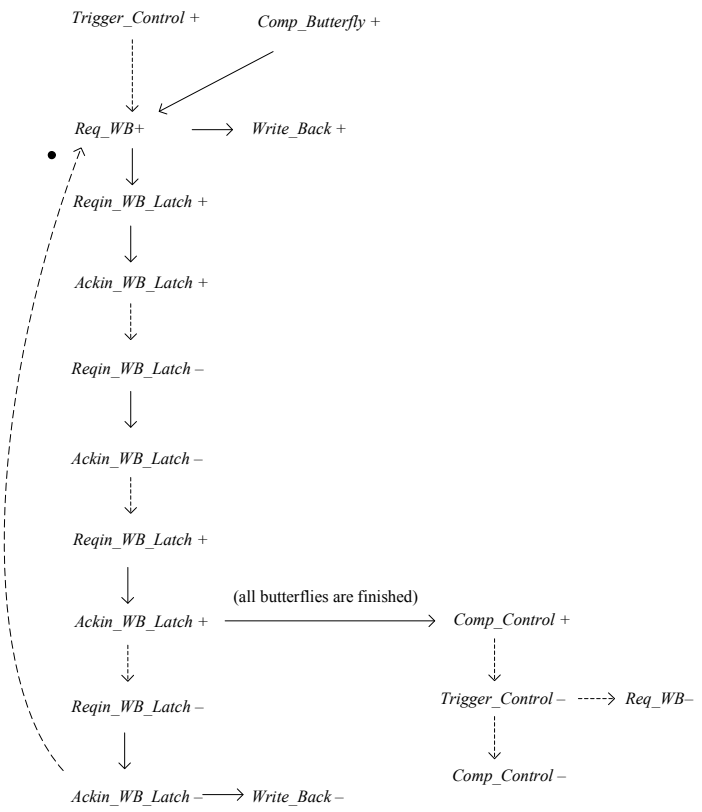

Fig. 5.11 The simplified signal transition graph for writing outputs into the memory

5.3.1 Control Portion

Fig. 5.6 earlier depicted the architecture of the control portion of the async FFT/IFFT processor, comprising the Async FFT/IFFT Controller and Pulse Circuit. The Async FFT/IFFT Controller is modeled using Verilog and its interfaces are modified for async operations. The simple linear pipeline for the Async FFT Controller is adopted as it does not require high speed operation in view of the intended hearing aid application. In other words, although the processing within one butterfly is operated in different pipeline stages and at different rates, the butterfly operations are in fact realized sequentially. This approach yields two advantages. First, the data flow of the handshake and

control signals can be greatly simplified, resulting in simple coding and hence, a simple hardware implementation. Second, the timing conflicts among different modules can also be easily identified and then corrected, resulting in simple verification. The drawback of this approach is the increased operation delay. However, the delay is inconsequential due to the intended low frequency application. Fig. 5.12 depicts a simplified modeling timing diagram for the Async FFT/IFFT Controller.

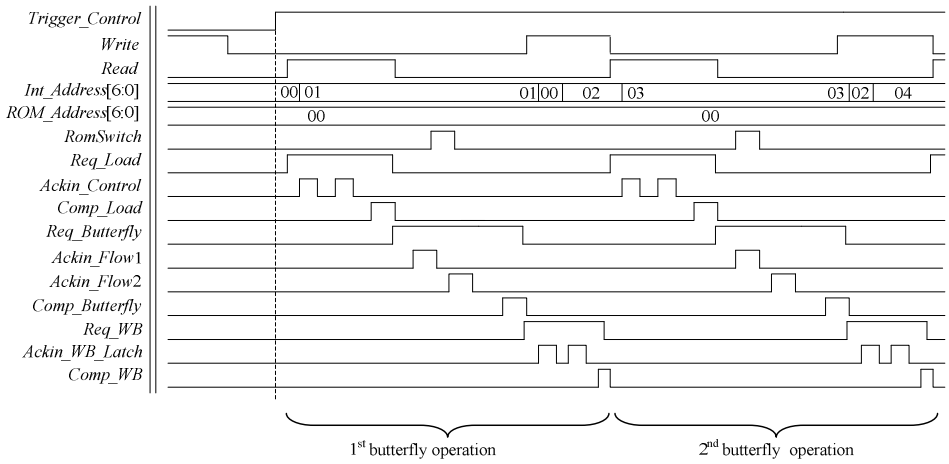

Fig. 5.12 A modeling timing diagram for the Async FFT/IFFT Controller. Note that the pertinent signals are only shown and the delays herein do not reflect the actual circuit delays

The gray-code method [70] is adopted for the Async FFT/IFFT Controller to reduce the possibility of static hazards – to avoid the immediate transitions that may possibly trigger the handshake and control signals. Most of the handshake and control signals are also gated (by means of AND gates) and they would only be triggered at appropriate time. Fig. 5.13 depicts an example of the control-state signal transition in the Write Back Controller. As depicted, only one bit transition changes (from '0' to '1' or vice versa) in the control-state signals

(enclosed in circles) from the current state to the next state per operation. The handshake and control signals, *Reqin_WB_latch* and *Alt*, are associated within the control-state (near the circle). The control-state signal flow in the right hand side is for writing the first set of outputs (R'_a and I'_a, see Chapter 2) in the butterfly operations, and in the left hand side for writing the second set of outputs (R'_b and I'_b, see Chapter 2). The recursive operations (first –> second –> first –> second ···) repeat until all butterfly operations are performed.

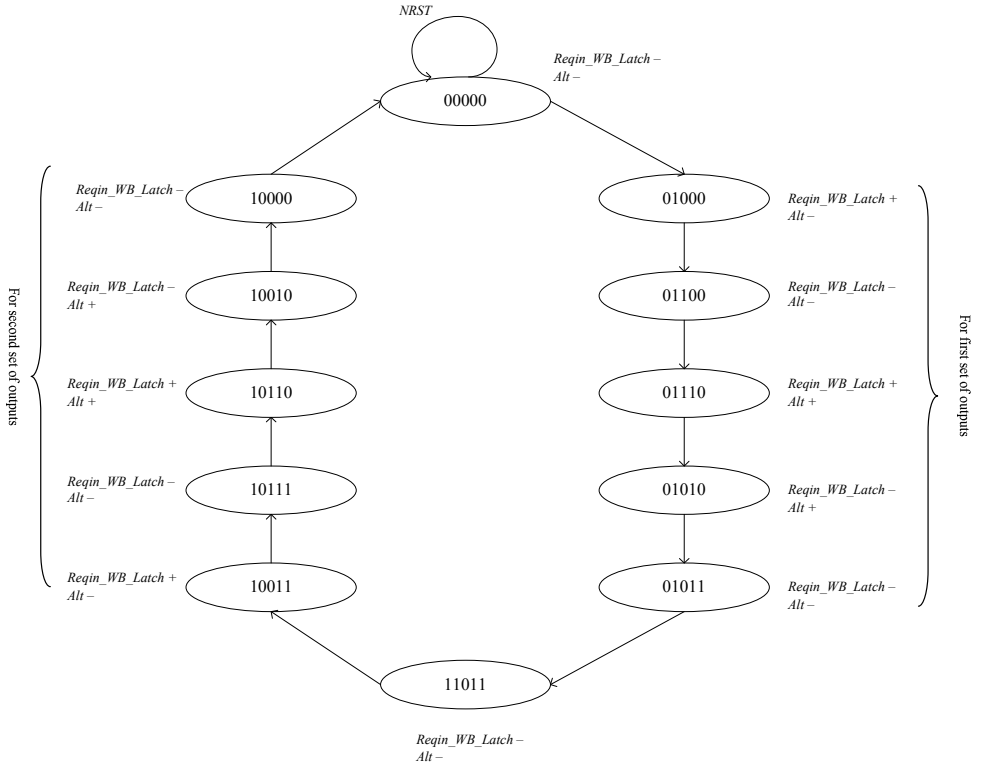

Fig. 5.13 The control-state signal transition flow (from the current state to the next state) in the Write Back Controller

The Pulse Circuit, embodying the pulse generator (see Chapter 2) and buffer circuits, coordinates the FFT/IFFT operations according to the handshake and control signals from the Async FFT/IFFT Controller.

5.3.2 Data Paths

The data path portion comprises the Memory Data Path, Shifter Data Path, Multiplier Data Path, Adder Data Path and Write Back Data Path and these are depicted in Figs. 5.14 to 5.18 respectively. For illustration purpose, only the pertinent handshake and control signals are shown.

5.3.2.1 Memory Data Path

The Memory Data Path depicted in Fig. 5.14 comprises two types of latch controllers (Broadish and Broad A latch controllers), multiplexers, latches, two async 128×16-bit memory macrocells, some control circuits and interfacing circuits. In the 4-phase HCCs therein, there are two Broadish latch controllers and two Broad A latch controllers. One of the Broadish latch controllers is used for a read operation and the other for a write operation. Similarly, one of the Broad A latch controllers is used for a read operation and the other for a write operation.

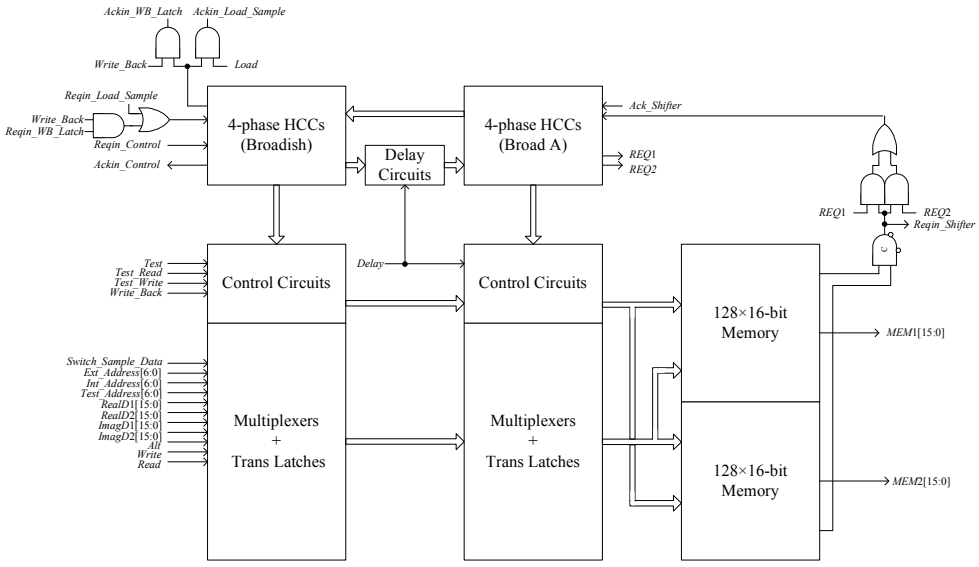

Fig. 5.14 Block diagram of the Memory Data Path

The Broadish latch controller first enables the latches to store input, addresses, handshake and control signals. The multiplexers multiplex the signals either for loading new samples (in a bit-reversal format) or for normal butterfly operations. After a predefined delay (by means of the delay circuits), the Broad A latch controller re-synchronizes the inputs to the memory macrocells. This synchronization is necessary as it ensures only the correct signals are sent to the memory macrocells. Another purpose for this synchronization is to synchronize the arrival times of all inputs to the memory to be approximately the same, resulting in reduced spurious switching in the memory macrocells. The memory macrocells finally compute the operations, for either the read or write access.

5.3.2.2 Shifter Data Path

The Shifter Data Path depicted in Fig. 5.15 comprises the Broadish latch controllers, latches, shifters, some control and handshake signals. As discussed in

Chapter 2 earlier, in a radix-2 FFT algorithm, the maximum overflows that can occur per butterfly operation are 2 bits. Hence, the shifters need only to be implemented using simple 3-to-1 multiplexers [20].

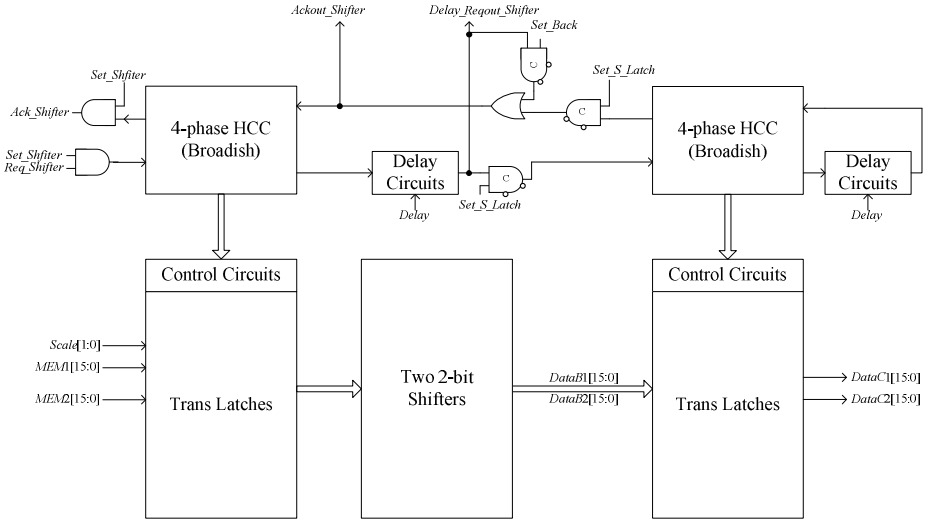

Fig. 5.15 Block diagram of the Shifter Data Path

The Broadish latch controller first enables the latches for storing the data from the memory, and the shifters then shift the data (where necessary) according to the *Scale*[1:0] signal. After a predefined delay (by means of the delay circuits), another Broadish latch controller enables temporary latches to store the first set of output data (*DataC*1[15:0] and *DataC*2[15:0] stored in the latches); *Set_S_Latch* is set to '1' at this time. The second set of output data (*DataB*1[15:0] and *DataB*2[15:0] stored in the shifters) is generated similarly by triggering the first Broadish latch controller. However, at this time, the second Broadish controller is not triggered; *Set_S_Latch* remains at '0'.

5.3.2.3 Multiplier Data Path

The Multiplier Data Path depicted in Fig. 5.16 comprises two types of latch controllers (Broadish and Broad B latch controllers), multiplexers, latches, ROMs, two Control-Multipliers, some control circuits and interfacing circuits.

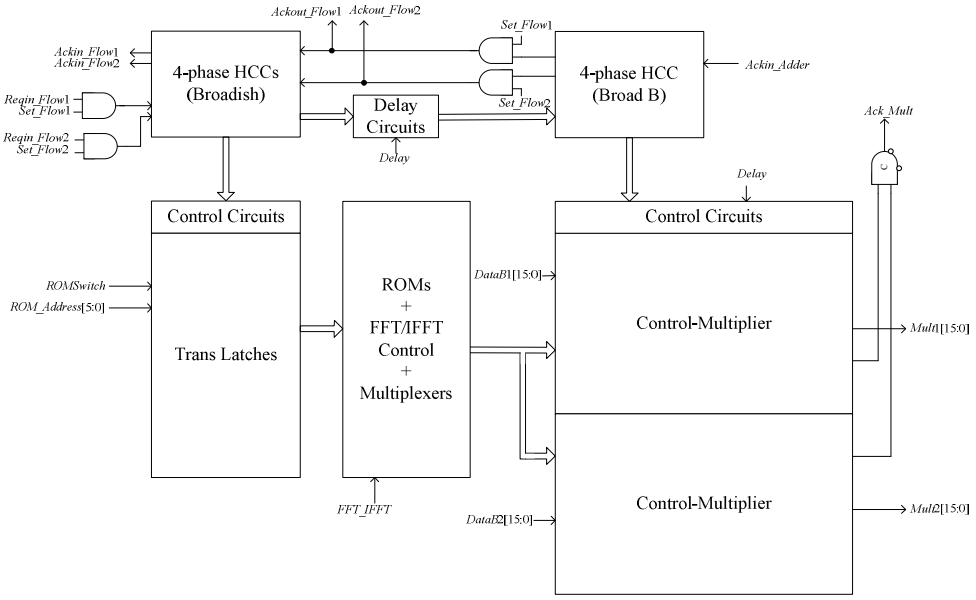

Fig. 5.16 Block diagram of the Multiplier Data Path

The Broadish latch controller first enables the latches to store ROM address that is used to retrieve the coefficients in the ROMs. The coefficients retrieved depend on the *FFT_IFFT* signal and the multiplexers. For IFFT operations, the sine coefficients that are stored in the ROMs will be negated; there is no change in cosine coefficients. For FFT operations, there is no change in the sine and cosine coefficients. After a predefined delay (by means of the delay circuits), the Broad B latch controller triggers the multiplications in the Control-Multiplier. The

Control-Multipliers finally compute the outputs (*Mult*1[15:0] and *Mult*2[15:0]), and the acknowledge signal, *Ack_Mult*, is generated.

The Broad B latch controller is used to ensure that the pertinent signals are stored in the latches before the interconnecting successive Control-Multipliers are asserted. It eliminates the possible spurious switchings that would otherwise propagate in the control circuits in the Control-Multipliers.

5.3.2.4 Adder Data Path

The Adder Data Path depicted in Fig. 5.17 comprises a Broad B latch controller, multiplexers, latches, three 16-bit async CCSAs, some control circuits and interfacing circuits. The 16-bit CCSAs are in hybrid '*Type-*α and *Type-*γ' CCSA configuration where the 4 adders in the LSBs are 2-bit *Type-*α CCSAs and the 4 adders in the MSBs are 2-bit *Type-*γ CCSAs (refer to Chapter 3).

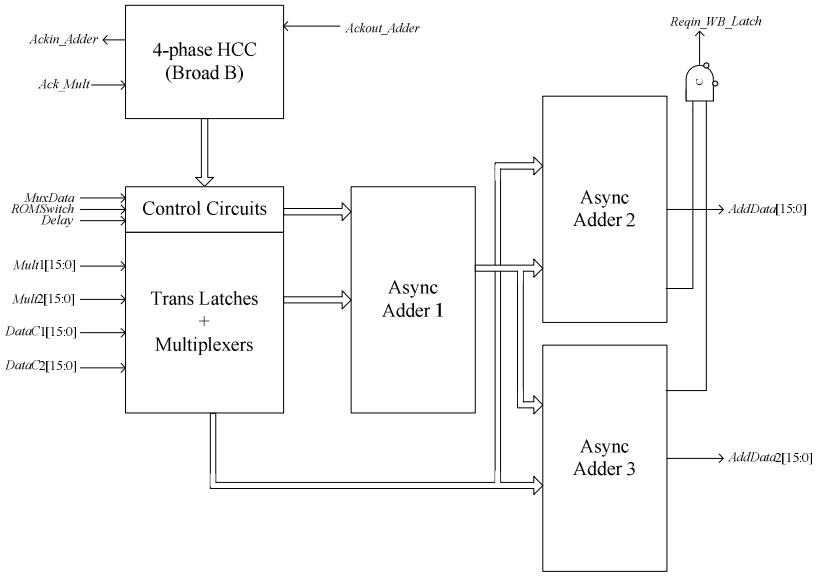

Fig. 5.17 Block diagram of the Adder Data Path

The Broad B latch controller first enables the latches to store the outputs from the Shifter Data Path (i.e. *DataC*1[15:0] and *DataC*2[15:0]) and from the Multiplier Data Path (i.e. *Mult*1[15:0] and *Mult*2[15:0]). Thereafter, depending on the *RomSwitch* signal, the async Adder 1 performs the subtraction or addition. The async Adder 1 subsequently activates the async Adder 2 and Adder 3. The async Adder 2 and Adder 3 finally generate the outputs, *AddData*1[15:0] and *AddData*2[15:0], and the acknowledge signal, *Reqin_WB_Latch*, is generated.

The Broad B latch controller is used to ensure all pertinent signals have been stored in the latches before activating CCSAs. This is to prevent the possibility of malfunction of the async CCSAs.

5.3.2.5 Write Back Data Path

The Write Back Data Path depicted in Fig. 5.18 comprises a Broadish latch controller, latches, block floating point circuits, some control circuits and interfacing circuits.

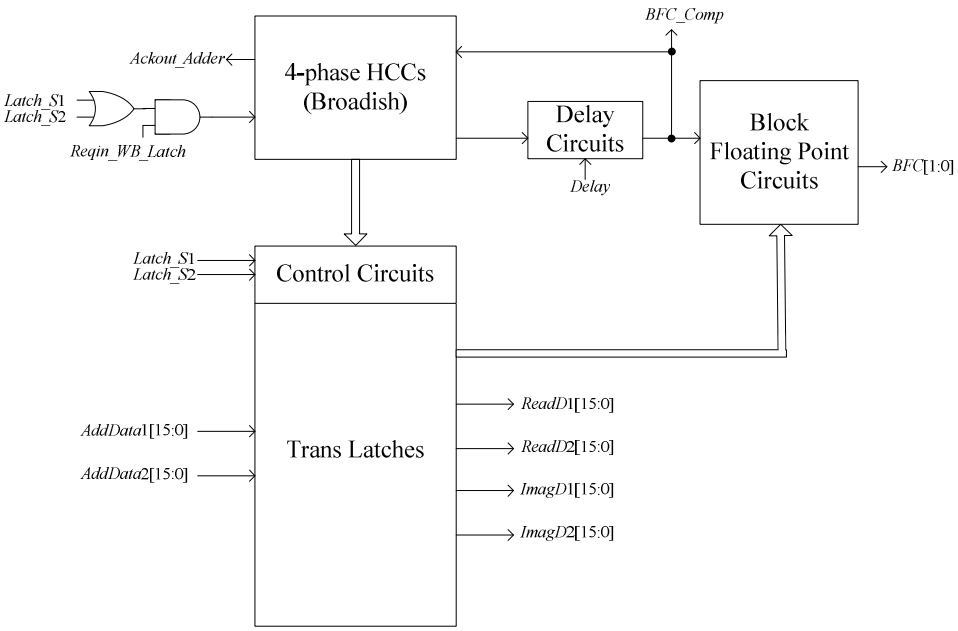

Fig. 5.18 Block diagram of the Write Back Data Path

The Broadish latch controller enables the latches to store the outputs from the Adder Data Path. After the outputs for each butterfly are stored temporarily, the *Ackout_Adder* signal is fedback to the Async FFT/IFFT Controller, and the Memory Data Path is triggered to finally re-write the temporary outputs into the memory. The block floating point circuits monitor the output and decide how many bits of scaling are required for the butterfly operations in next stage, avoiding overflow.

5.4 Measurements and Comparisons

Figs. 5.19 and 5.20 depict the microphotographs of the sync FFT/IFFT processor and the proposed async FFT/IFFT processor respectively. The sync FFT/IFFT design occupies an IC area of ~ 1.44mm^2 and the async FFT/IFFT design occupies

an IC area of ~1.6mm^2. Both designs are realized using the same 0.35μm dual-poly four-metal CMOS process.

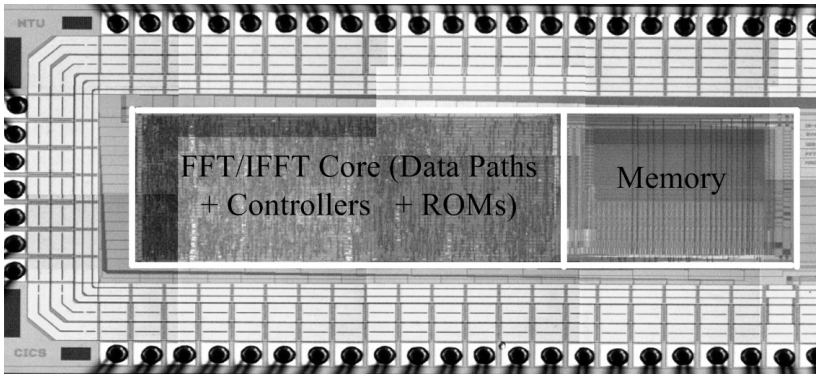

Fig. 5.19 Microphotograph of the sync FFT/IFFT processor

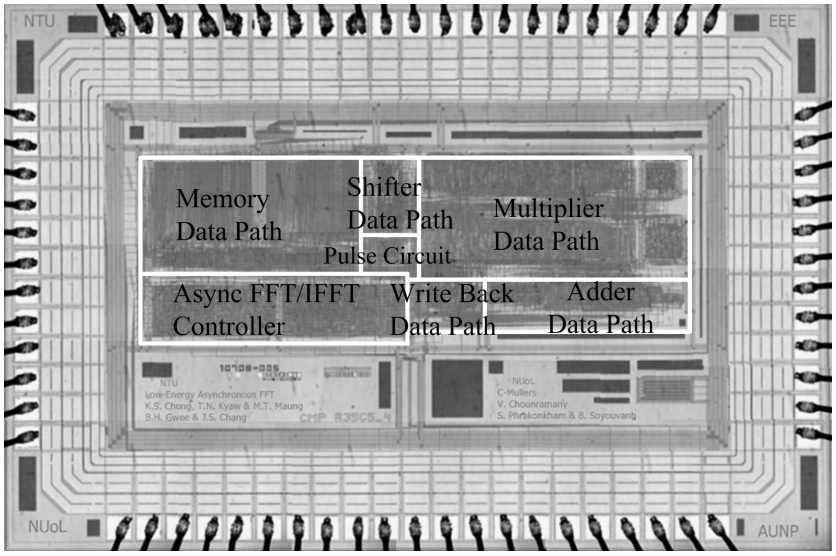

Fig. 5.20 Microphotograph of the async FFT/IFFT processor

The sync FFT/IFFT processor takes 256 clock cycles to write new samples into and read the outputs from the memory. The butterfly operations require 959 clock cycles. Hence, one complete FFT (or IFFT) computation requires 1215 clock cycles.

Similar to the sync FFT/IFFT processor, three delays are included for the async FFT/IFFT processor. There are (1) the delay for loading the 128 input data into the memory, (2) the delay for performing butterfly operations, and (3) the delay to output the 128 output data from the memory. However, in this case, the delays are determined based on its request and acknowledge signals.

Fig. 5.21 depicts the constructed test jig for the async FFT/IFFT processor. Based on the given random inputs, all measured outputs (from the prototype ICs) are compared with the expected outputs obtained from the simulations. Fig. 5.22 depicts the real outputs obtained from the logic analyzer and the prototype ICs are functionally correct.

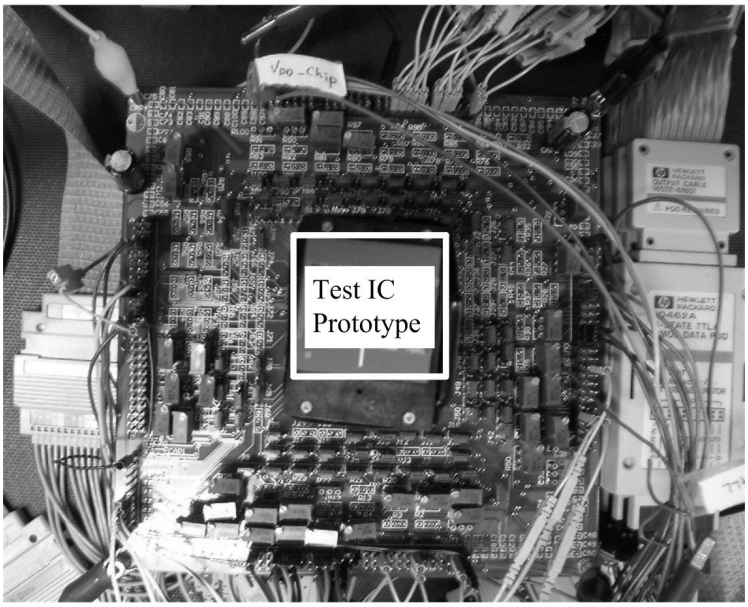

Fig. 5.21 The test jig for the async FFT/IFFT processor

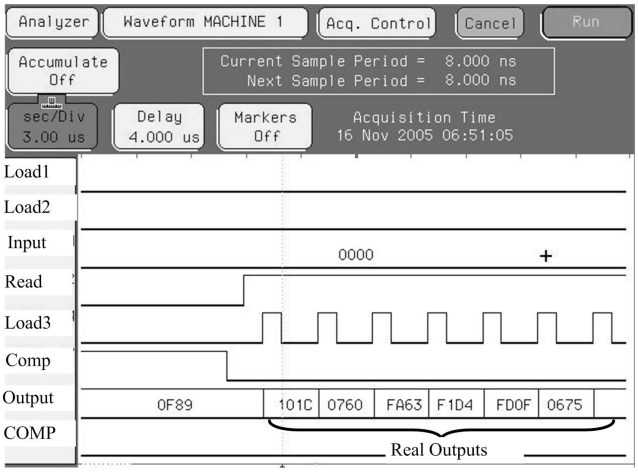

Fig. 5.22 The real outputs obtained using the logic analyzer

To best evaluate the async FFT/IFFT processor, its sync counterpart is used as a benchmark. Table 5.2 re-tabulates some characteristics of the two IC chips. In general, the algorithm, process used, and computational complexity of these two designs are the same, and their architectures are largely similar. The sync design is based on standard digital cells and a 128×32-bit memory obtained from the foundry. The async design is based on several fully and partially handcrafted async cells (including two block of async 128×16-bit memory macrocells, two Control-Multipliers, etc) and where pertinent, on the same cells used in the sync design. The philosophy behind the evaluation process is an attempt to make the comparison as fair as possible.

TABLE 5.2 THE CHARACTERISTICS OF THE SYNC AND ASYNC FFT/IFFT PROCESSORS

	Sync IC	Async IC
Algorithm	Radix-2 decimation-in-time	
Number of point	128	
Computational effort	7×64 butterfly operations	
Process	0.35μm (dual-poly four-metals) CMOS	
No. of multiplier used	Two	
No. of adders used	Three	
No. of memory used	One (128×32-bit)	Two (128×16-bit each)
Cells	Standard library	Standard library + custom cells

Table 5.3 tabulates the operating conditions for the sync and async FFT/IFFT processors. The time taken for simulations and measurements is 4ms (equivalent to the time of 64 samples @ 16KHz sampling frequency).

Two scenarios are considered when measuring the energy dissipation of the sync FFT/IFFT design. In the first scenario, the clock signal is allowed to enter into the sync design and this scenario is termed as 'No Gating'. The unused clock cycles in the 'No Gating' scenario would result in wasted power/energy dissipation (primarily in the clock buffers and in the flip-flops). In the second scenario, only 1215 clock cycles (the number of clock cycles required for one complete FFT or IFFT computation) are sent to the sync design within 4ms, and the clock signal is thereafter disabled. This scenario is termed as 'Gating'.

TABLE 5.3 OPERATING CONDITIONS FOR THE SYNC AND ASYNC FFT/IFFT PROCESSORS

	Sync IC		Async IC
	Gating	No Gating	
Voltages	1.1V to 1.4V	1.1V to 1.4V	1.1V to 1.4V
Timing	4ms	4ms	4ms
Gating clock	Yes	No	—
Operations	Loading 128 set of inputs, performing 7 × 64 butterflies, and generating 128 set of outputs		

Fig. 5.23 depicts the energy dissipation of the proposed async design and of the sync design in the 'Gating' and 'No Gating' scenarios @ 1.0V – 1.4V. Note that the system clock (Clk1) of the sync design is 1MHz (where Clk2 is 2MHz). The async prototype IC can work at 1.0V but the sync prototype IC fails to work at 1.0V. The reason for this will be discussed later.

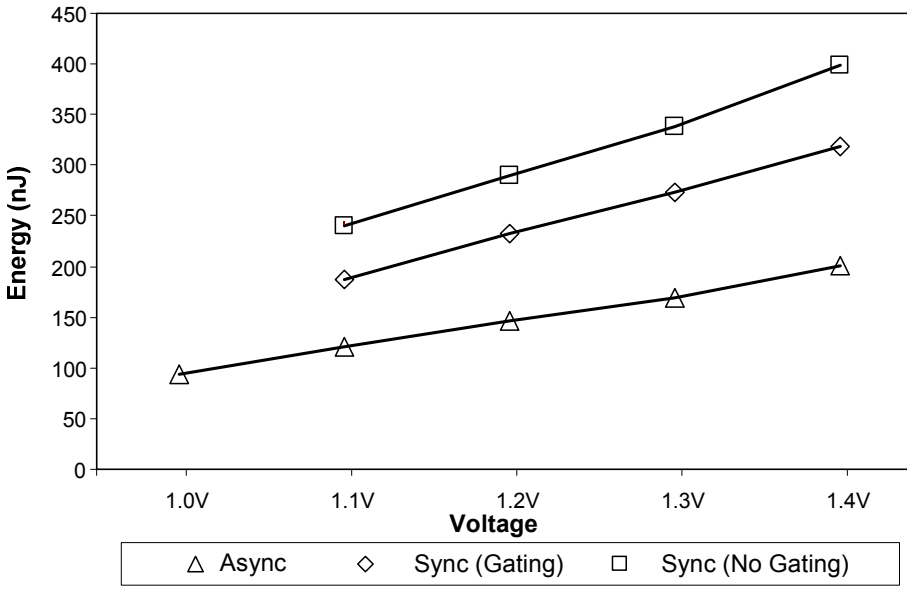

Fig. 5.23 Energy dissipation for the async and sync (@ 1MHz) FFT/IFFT processors at 1.0V to 1.4V

From Fig. 5.23, the proposed async design features lower energy dissipation than the sync design for both scenarios. Based on the measurements on prototype ICs, the async design, on average from 1.1V to 1.4V, is ~ 37% and ~ 50% lower energy compared to the sync design with the 'Gating' and 'No Gating' scenarios respectively. The voltage range of 1.1V to 1.4V is the typical voltage range for hearing aids.

Apart from the novelties from the previous microcell and macrocell design, in particular when compared to the sync design with 'Gating', the low energy feature of the proposed async design is attributed to three reasons.

First, in the sync design, despite the coarse-grain clock gating described earlier, a small number of data path circuits (e.g. during the initialization phase where signals are not ready for the particular pipeline stage) remain asserted by the clock signal(s), resulting in ~ 6% redundant operations in some pipeline stages. To design the sync circuits to eliminate the 6% redundant operations by means of fine-grain clock gating would require a more complex sync controller with multi-phasic clocks and complex logic circuits, hence largely defeating the 6% saving and this would likely incur further costs due to the associated overheads. This redundant switching does not occur in the async design. Further, the 96 flip-flops that serve as delay registers to pipeline the data sequence to the data path circuits in the sync design are unnecessary in the async design. Overall, this first reason accounts for an overall reduced energy of ~ 9%.

Second, the spurious switchings in the async data path circuits are reduced by synchronizing the arrival time of the inputs to the custom-designed data path circuits (e.g. memories, multipliers, adders, etc.) with better control by means of the latches and delay lines, by gating the redundant hazards and transitions with async handshake circuits, and by disabling the unused blocks (particularly for latches whose data are not updated for computations). In the sync design, the inputs of the signals can also be synchronized at added cost and increased design complexity, and this is applicable only to the circuits where input flip-flops are present. Overall, this second reason resulting in reduced switchings accounts for ~ 18% energy saving.

Third, a latch is typically dissipated 50% lower energy than a flip-flop and by comparison, there are a total of 783 flip-flops in the sync design against the 122 flip-flops and 459 latches in the async design. The sync design can also be designed based on a less prevalent latch-based approach, a somewhat unconventional approach that is more complex including the need to consider the signal synchronization more carefully. Overall, this third reason accounts for ~ 10% energy saving.

It is also worthwhile to mention that the clocking energy dissipation in the sync FFT/IFFT processor is estimated to be small, ~ 5% of the total energy dissipation. This can be attributed to the low clock rate, the FFT/IFFT architecture, and the simplicity of a simple buffer network for the clock infrastructure. In other words, in this specific low clock frequency example, the absence of a global clock infrastructure in the async realization does not contribute significantly to the

energy savings. The energy dissipation due to the clock infrastructure is likely to be more significant if the clock rate is high and/or if the IC area is large.

Fig. 5.24 depicts the delays of the async and sync designs. The delay here is defined as the minimum time for one complete FFT computation. Note that the minimum time of the sync design is the same for the 'Gating' and 'No-Gating' scenarios. The async design has ~ 1.4 times average worse delay (1.1V to 1.4V) than the sync design and if the matched delay elements in the former are better tuned, the delay is estimated to be reduced to ~ 0.6 times worse. Nevertheless, the async design still meets the speed requirement, < 2ms, for the intended hearing aid application. Note that the async design can also be designed to be faster by means of high speed early latch controllers [38], [87] and this may not necessarily incur higher energy dissipation. However in view of the intended low speed hearing aid application, this added speed is unnecessary.

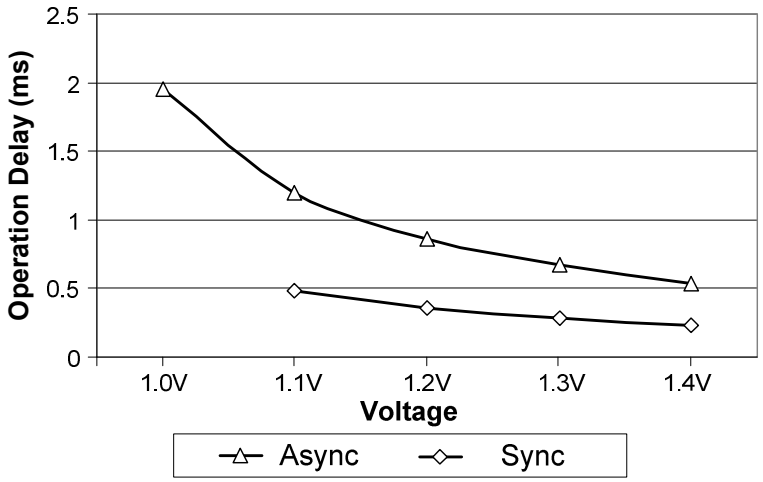

Fig. 5.24 Delays for the async and sync FFT/IFFT processors

As mentioned earlier, the async design can operate at a lower functional operating voltage (e.g. 1V) albeit the longer delay. It fails to operate when $V_{DD} < 1V$ largely due to the delay mismatch in some matched delay components. A dual-rail delay-insensitive approach [38] may overcome the delay mismatch problem, thereby theoretically allowing the async design to operate as long as $V_{DD} > V_T$, but at the expense of higher energy dissipation and area overhead. In the case of the sync design, it fails to operate when $V_{DD} < 1.1V$ largely due to clock skews. Theoretically, the sync design can also operate for $V_{DD} > V_T$ if clock skews can be accommodated – assuming the critical path therein is shorter than the clock delay. However, the design of a low skew clock distribution network in a sync design is a challenging task, and the clock network may dissipate significant power/energy.

Fig. 5.25 depicts the energy-delay curves for the async and sync FFT/IFFT processors, by varying both the voltages (within 1V to 1.4V) and delay. The delay here is the time required for one complete FFT (or IFFT) computation and this may not necessarily be the maximum speed. These energy-delay curves provide an insight to how energy can be traded for speed (the converse is true) by changing the operating conditions. The region X and region Y respectively indicate the energy-delay space for the sync design in the 'Gating' and 'No-Gating' scenarios. They are spaces (regions) in the sync design because both the parameters, voltage and delay, can be adjusted independently. The bold solid line indicates the energy-delay curve for the async design. It is only a curve for the async design because its delay depends only on the supply voltage.

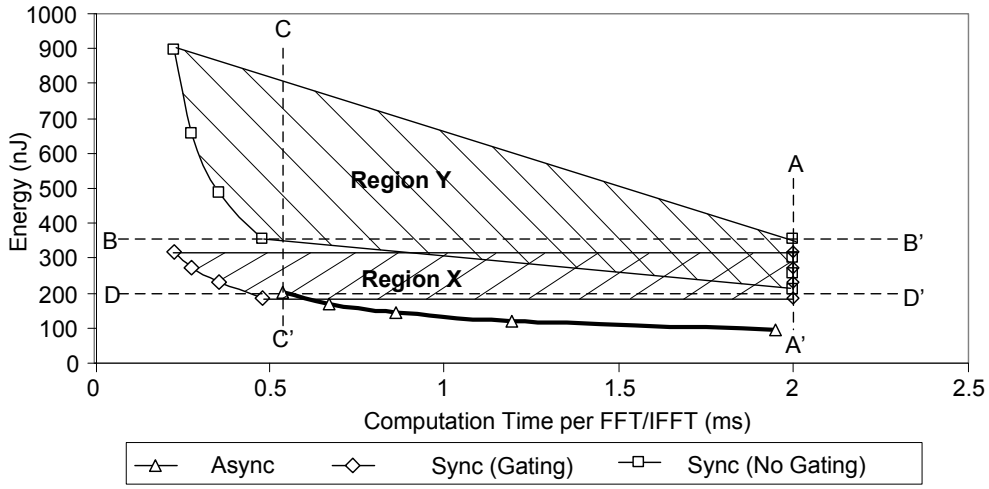

Fig. 5.25 Energy-delay curves of the async and sync FFT/IFFT processors

Five interesting remarks can be made from Fig. 5.25.

(1) The sync design has to be clocked at > 0.6075MHz so that the delay is < 2ms (the pertinent requirement for the intended hearing aid application, see Table 1.1). The delay curves of the sync and async design that lie to the left hand side of dotted A to A' line will meet the delay requirement.

(2) The region below the dotted B to B' line indicates the energy budget required for the FFT/IFFT processor. Both the proposed async design and the sync design in 'Gating' scenario meet this energy requirement. The sync design in the 'No Gating' scenario only conditionally meets the energy requirement (e.g. at low voltage and low speed conditions).

(3) If the sync design lies at the left hand side of the dotted C to C' line, it definitely runs faster than the async design. Otherwise, the delays of the sync and async designs are comparable, depending on the voltage and frequency used (required in the sync design).

(4) It is possible that the sync design in the 'Gating' scenario conditionally dissipates lower energy than the async design (region slightly below the

dotted D to D' line). This occurs when the async and sync FFT/IFFT processor operate at different voltages, and the sync design operates at much lower voltage condition than the async design, e.g. the sync design @ 1.1V and the async design @ 1.4V. This is, however, not a meaningful comparison.

(5) To summarize above, for non-critical speed applications including hearing aids, the proposed async design is highly energy efficient (more energy efficient than the sync design) for various voltage conditions.

To further delineate the design of the async FFT/IFFT processor, Fig. 5.26 shows its distribution of number of transistors in various modules. The Memory Data Path makes up the largest portion, ~ 43% total of the transistors, followed by the Multiplier Data Path, ~ 30% total of the transistors, and the Async FFT/IFFT Controller and Pulse Circuit, ~ 19%. The remaining modules only take up ~ 8% of the transistors. If it is desired to reduce the IC area of the async FFT/IFFT processor, the IC area of the Memory and Multiplier Data Paths could be the primary candidate.

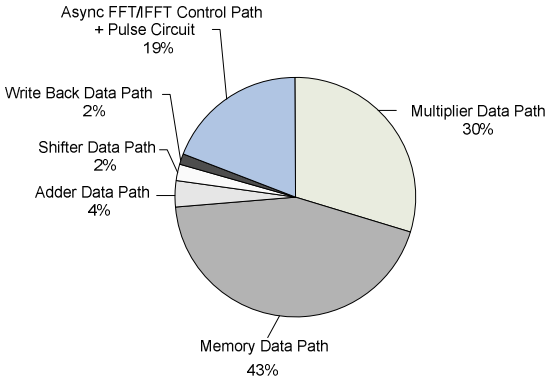

Fig. 5.26 The distribution of the number of transistors in various modules in the async FFT/IFFT processor; the total number of transistors is ~ 74K

Fig. 5.27 depicts the energy breakdown (in %) of the async FFT/IFFT processor at 1.1V to 1.4V (based on the post-layout computer simulations). Both the Multiplier Data Path and the Memory Data Path make up more than 33% of the total energy dissipation each. The Shifter Data Path, Adder Data Path, and Write Back Data Path collectively dissipate ~ 16% of the total energy dissipation. The Async FFT/IFFT Controller and Pulse Circuit dissipate only 14% of the total energy.

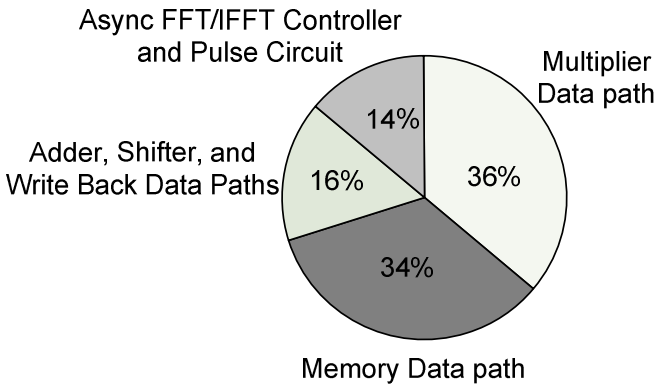

Fig. 5.27 The energy breakdown (in %) of the async FFT/IFFT processor at 1.1V to 1.4V

Based on simulations, the switching activity of the async FFT/IFFT processor (for one complete computation) is ~ 4.5 million. Unfortunately, the detailed % distribution (transistor count and energy) and the switching activity of the sync FFT/IFFT processor are unavailable because the transistor-netlist of the 128×32-bit memory IP core (proprietary core) is unavailable.

Table 5.4 tabulates a comparison of several low energy FFT designs [26], [31]–[34], [138] including the benchmarked sync FFT/IFFT and the proposed async FFT/IFFT designs therein. The entries in the last column, normalized FFTs

(million, M) per energy, have been scaled to a 0.35μm CMOS, 1.1V, 16-bit wordlength and 128-point FFT algorithm based on (5.1) [33].

$$\text{Normalized FFTs/Energy} = \frac{(\frac{L_{eff}}{0.35})\times(\frac{V_{DD}}{1.1})^2\times[\frac{2}{3}\cdot\frac{W_{bit}}{16}+(1-\alpha)\cdot\frac{1}{3}\cdot(\frac{W_{bit}}{16})^2]\times(\frac{(N/2)\log_2 N}{448})}{\text{Energy Dissipation}} \quad (5.1)$$

where L_{eff} is the effective transistor length of the CMOS process, W_{bit} is the wordlength, φ is the truncated portion for the multiplier therein, and N is the number of points of the FFT algorithm.

TABLE 5.4 CHARACTERISTICS OF RECENT FFT PROCESSORS AND THE PROPOSED ASYNC AND BENCHMARKED SYNC FFT/IFFT PROCESSORS

	V_{DD} (V)	L_{eff} (μm)	N	W_{bit} (bit)	φ	Algorithm	Energy (μJ)	Normalized FFTs (M) per Energy
Bass, 1999 [33]	1.10	0.6	1024	20	0.27	Radix-2	3.14	7.6
Wang, 2005 [34]	0.35	0.18	1024	16	0	Radix-2	0.16	3.7
Stevens, 1998 [138]	3.30	N.A	1024	32	0	Mixed	18.00	N.A
Ding, 1999 [31]	3.30	0.70	64	24	0	Radix-4	1.95	3.5
Yeh, 2003 [26]	3.30	0.35	64	12	0	Split-radix	0.33	8.0
Jia, 1998 [32]	3.30	0.60	128	10	0	Radix2/4/8	1.20	7.0
Sync (Fig. 5.19)	1.10	0.35	128	16	0	Radix-2	0.19	5.3
Async (Fig. 5.20)	1.10	0.35	128	16	0	Radix-2	0.12	8.3

Although a comparison between the various designs is contentious due to large variations of the designs and parameters therein, it is nonetheless worthwhile to note that the benchmarked sync design and the proposed async designed described in this thesis are indeed energy-efficient and the latter being the more efficient.

In summary, a low voltage low energy async FFT/IFFT processor that satisfies all specifications in Table 1.1 has been successfully designed.

5.5 Summary

In this chapter, the design of a low voltage low energy sync FFT/IFFT processor and the proposed async FFT/IFFT processor has been described. The sync design served as a benchmark as the sync approach is the current prevalent approach. The async approach has been shown to be higher energy efficient by means of established and proposed circuit designs, including (i) reduced redundant operations, (ii) energy-efficient data path circuits, and (iii) the use of simple latches. The proposed async FFT/IFFT processor has been shown to dissipate ~ 37% and ~ 50% lower energy than its sync counterpart with and without the clock gating approach respectively. The drawbacks of the async design have been shown to be a 10% IC area penalty and an inconsequential (for the intended application; < 5MHz) 1.4 times worse delay. The proposed async design has also been shown to be competitive in terms of energy efficiency when benchmarked against other reported FFT designs.

In summary, a low voltage low energy async FFT/IFFT processor that satisfied all specifications in Table 1.1 has been successfully designed.

Chapter 6

Conclusions and Recommendations

6.1 Conclusions

This thesis has reported the adoption and exploitation of the async approach for the realization of the low energy attribute 128-point radix-2 decimation-in-time FFT/IFFT processor for energy-critical audio applications, including hearing aids. The proposed async FFT/IFFT processor IC dissipated ~ 93nJ to ~ 201nJ @ 1.0V – 1.4V, featured a delay of < 1.95ms @ $V_{DD} \geq$ 1.0V, and occupied an IC area of 1.6mm^2 @ 0.35μm dual-poly four-metal CMOS process. The proposed async FFT/IFFT processor has been benchmarked against its sync counterpart that embodied a similar architecture and fabricated using the same CMOS process technology. The proposed async FFT/IFFT processor has been shown to be ~ 37% and ~ 50% more energy efficient than its sync counterpart with and without the clock gating approach respectively. It has also been shown to feature an extremely low energy attribute (as benchmarked against reported FFT processor designs). The minor drawbacks of the async FFT/IFFT processor were the larger IC area required, an increase of 10%, and an inconsequential 1.4 times average worse delay; inconsequential due to the intended low speed (< 5MHz) hearing aid application.

A bottom-up approach has been adopted for the design of the async FFT/IFFT processor – by building the basic library microcells and macrocells, and finally

constructing the async FFT/IFFT processor. The low energy attribute of the proposed async FFT/IFFT processor was largely due to the adoption of established and proposed async microcells and proposed macrocells, and the adoption of the async approach (as opposed to the sync approach). These resulted in reduced redundant operations and reduced spurious switching.

Several proposed microcells, including the LA, Latch Accumulator, *Type-*γ adder (a 2-bit CCSA), and Broad B latch controller, have been presented and these microcells served as basic building blocks for general async designs, including the async FFT/IFFT processor. The proposed LA incorporated a design that integrated an adder and a latch, and the proposed Latch Accumulator incorporated a design that integrated the LA and a latch. The proposed 2-bit *Type-*γ CCSA was robust against delay variations by means of the insertion of four delay balancing transistors in its Carry block. The proposed Broad B latch controller featured a design that eliminated the possibility of malfunction of the interconnecting successive functional modules. These proposed microcells featured low voltage low energy attributes and in most cases have been shown to feature lower energy than reported designs. In summary, these async microcells were shown to be appropriate for low voltage energy-critical applications.

Several proposed macrocells, including the novel low voltage low energy Booth array-based multiplier core, the async Control-Multiplier and the async memory macrocell, have been presented. The proposed Booth array-based multiplier core employed the proposed LAs in the adder array of the multiplier to reduce spurious switching, hence reduced energy dissipation. The proposed multiplier core has

been shown to have the least transistor switching (spurious and that for computation) and the lowest energy dissipation compared to reported multiplier cores. The proposed async Control-Multiplier embodied the proposed novel 16×16-bit Booth array-based multiplier core and control circuits. The multiplier core was for non-trivial multiplications and the control circuits were for trivial multiplications and async handshake controls. The proposed Control-Multiplier has been shown to have a further 22% energy reduction compared to the proposed multiplier core based on the simulations with 40% (estimated % for the computation in the 128-point FFT algorithm) of the multiplications being trivial. The proposed async 128×16-bit memory macrocell embodied four smaller 32×16-bit sub-memory blocks and the proposed Word Line Controllers that activated the particular sub-memory block of interest and hence disabled other unselected sub-memory blocks. In summary, these macrocells were shown to be appropriate for low voltage energy-critical applications.

In summary, a low voltage low energy async FFT/IFFT processor that satisfied all specifications in Table 1.1 has been successfully designed. The objectives of this research program have been met.

6.2 Recommendations: Future Research

6.2.1 Development of Advanced Asynchronous-Logic Design Tools

Existing sophisticated commercial EDA tools are available primarily only for sync designs and not for async designs. This is, as described in Chapter 1, largely because of the prevalence, relative simplicity, and acceptance of sync designs.

In view of the complexity of current-art digital devices, it is obvious that if async designs were to be accepted as a mainstream design approach similar to sync designs, advanced async EDA tools would need to be developed.

It is recommended that advanced async EDA tools be developed. In addition to the usual technical challenges of EDA tools development, considerations for the adoption of async designs include: (i) verification, (ii) testability, including the fault coverage test, and (iii) circuit overhead for async signaling protocols, etc.

6.2.2 Design of Microcells and Macrocells

Following the first recommendation, it is recommended that a set of well-characterized microcells and macrocells be designed and established for async EDA tools. Much of the successes of EDA tools for digital circuit designs depend on the quality of the library cells.

In view of the diversity of design specifications, the library cells should include design variations for different speeds, operating voltages, etc – akin to what is available for sync designs. For example, for high speed microcells and macrocells, the dual-rail design techniques such as differential cascode voltage swing (DCVS) logic, and various transistor sizings (or logical effort [139]) should be considered. For low power low energy microcells and macrocells, considerations for low energy dissipation include the methodologies proposed and employed in this thesis. These include (i) low overhead circuit integration from different designs, (ii) reduced spurious switching technique, (iii) custom design approach, etc.

In view of the current-art advanced process technology (e.g. 90nm CMOS), the library cells should also consider design issues (e.g. delay variations and leakage power dissipation) related to advanced fabrication processes – applicable to both sync and async library cells. In all cases, the power/energy dissipation due to switching is expected to be significant. In view of this, the methodologies proposed and employed in this thesis for the library cells are also applicable to advanced fabrication processes. However, the added design considerations include accounting for more delay variations (e.g. by means of more delay lines or using dual-rail circuits) and for leakage current reduction [67], [68].

6.2.3 Leakage Current Reduction by Asynchronous-Logic Techniques in Advanced CMOS Processes

In line with the CMOS technology roadmap, advanced CMOS processes are very short-channel or nano-scaled (i.e. < 0.18μm feature-size) and are ever shrinking (although 'limits' have been asserted by some). The leakage current, as described in Chapter 2, in these processes has become significant and resultant leakage power dissipation may even exceed the switching power dissipation.

The usual methods to reduce leakage current include stacked CMOS, dual-threshold CMOS, and dynamic-threshold CMOS, etc [67], [68]. Most of these methods, to some extent, require extra control circuits (such as stacked transistors, sleep transistors, and monitoring circuits) to reduce the leakage current according to the input vector applied and the operating conditions therein. It appears that the async handshake circuits and its associated overhead may also serve as control circuits for leakage current reduction. This implies, although somewhat contentious, that async designs are more apt to accommodate the leakage current issues and that the design effort to reduce leakage current can be mitigated.

It is recommended that there are some research effort to discover async design techniques for circuits realized in advanced CMOS processes to reduce the leakage current in advanced CMOS processes. The challenges include: (i) similar to the recommendation in Section 6.2.2, the design and realization of async microcells and macrocells with reduced leakage current, and (ii) a design style(s) for monitoring the input vectors and operating conditions within the async circuits.

6.2.4 Adoption of Asynchronous-Logic Circuits in the Sub-Threshold Region

In line with the CMOS technology roadmap, the operating voltage will also likely be scaled down and the operating voltage may possibly fall well below the threshold voltage of transistors, e.g. in the sub-threshold regions of 0.2V – 0.3V, for ultra low energy-critical applications [9], [34]. The reduced operating voltage is likely to pose some very challenging problems, including larger delay variations, clock slews and clock jitters in prevalent sync designs. It is likely that async designs could more easily accommodate these challenging issues because of their inherent skew-less and handshake operations.

It is recommended that the async approach be adopted to circuits operated in the sub-threshold region for ultra low energy applications. In line with Section 6.2.3, circuits operated in the sub-threshold region also experience reduced leakage current. The issue of small noise margin, for both sync and async sub-threshold operations, should also be addressed.

Author's Publications

Journals

[1] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "A micropower low-voltage multiplier with reduced spurious switching," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, Regular Paper, vol. 13, no. 2, pp. 255–265, Feb. 2005.

[2] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "Energy-efficient synchronous-logic and asynchronous-logic FFT/IFFT processors," accepted for publication in *IEEE J. Solid-State Circuits*, Regular Paper, 2007.

[3] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "Design of several asynchronous-logic macrocells for a low-voltage micropower cell library," *IET Proc. Circuits Devices Syst.*, vol. 1, no. 2, pp. 161–169, 2007.

[4] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "A 16-channel low power non-uniform spaced filter bank core for digital hearing aids," *IEEE Trans. Circuits Syst. II*, vol. 53. no. 9, pp. 853-857, Sep. 2006.

[5] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "Low energy 16-bit Booth leapfrog array multiplier using dynamic adders," *IET Proc. Circuits Devices Syst.*, vol. 1, no. 2, pp. 170–174, 2007.

Patents

[1] J. S. Chang, B.-H. Gwee, and K.-S. Chong, "A digital multiplier with reduced switching by means of latch adders", Singapore Patent, 2003, Granted.

[2] J. S. Chang, B.-H. Gwee, and K.-S. Chong, "A digital multiplier with reduced switching by means of latch adders" U.S. Patent, 2007, Granted.

Conference Papers

[1] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "A low energy asynchronous FFT/IFFT processor for hearing aid applications," in *Proc. IEEE Int. Conf. Electron Devices Solid-State Circuits*, 2005, pp. 751–754.

[2] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "Low-voltage micropower multipliers with reduced spurious switching," in *Proc. IEEE Int. Sym. Circuits Syst.*, vol. 4, 2005, pp. 4078–4081.

[3] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "A robust low voltage low energy asynchronous carry-completion sensing adder for biomedical applications," in *Proc. IEEE Int. Workshop Biomedical Circuits Syst.,* 2004, pp. S1/2-18–21.

[4] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "A low-voltage low power accumulator," in *Proc. IEEE Int. Sym. Integrated Circuits Devices Syst.*, 2004, pp. F1057–GTJ9.

[5] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "A low power 16-bit Booth leapfrog array multiplier using dynamic adders," in *Proc. IEEE Int. Sym. Circuits Syst.*, vol. 2. 1, 2004, pp. 437–440.

[6] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "A critical-band filterbank based on IFIR and frequency-response masking techniques for digital hearing instruments," in *Proc. Int. Sym. Consumer Electronics*, 2003.

[7] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "Low-voltage asynchronous adders for low power and high-speed applications," in *Proc. IEEE Int. Sym. Circuits Syst.*, vol. 1, 2002, pp. 873–876.

[8] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "Low-voltage micropower asynchronous multiplier for hearing instruments," in *Proc. IEEE Int. Sym. Circuits Syst.*, vol. 1, 2002, pp. 865–868.

Bibliography

[1] H. Neutboom, B. M. J. Kup, and M. Jassens, "A digital-based hearing instrument IC," *IEEE J. Solid-State Circuits*, vol. 32, no. 11, pp. 1790–1806, Nov. 1997.

[2] P. Mosch *et al.*, "A 660-μW 50-Mops 1-V DSP for a hearing aid chip set," *IEEE J. Solid-State Circuits*, vol. 35, no. 11, pp. 1705–1712, Nov. 2000.

[3] D. G. Gata *et al.*, "A 1.1-V 270-μA mixed-signal hearing chip," *IEEE J. Solid-State Circuits*, vol. 37, no. 12, pp. 1670–1678, Dec. 2002.

[4] L. S. Nielsen, and J. Sparsø, "Designing asynchronous circuits for low power: an IFIR filter bank for a digital hearing aid," *Proc. IEEE*, vol. 87, no. 2, pp. 268–281, Feb. 1999.

[5] A. Deiss, and Q. Huang, "A low-power 200-MHz receiver for wireless hearing aid devices," *IEEE J. Solid-State Circuits*, vol. 38, no. 5, pp. 793–804, May 2003.

[6] A. Deiss, D. Pfaff, and Q. Huang, "A 200-MHz sub-mA RF front-end for wireless hearing aid applications," *IEEE J. Solid-State Circuits*, vol. 35, no. 7, pp. 977–986, Jul. 2000.

[7] S. Kim, J.-Y. Lee, S.-J. Song, N. Cho, and H.-J. Yoo, "An energy-efficient analog front-end circuit for sub-1-V digital hearing aid chip," *IEEE J. Solid-State Circuits*, vol. 41, no. 4, pp. 876–882, Apr. 2006.

[8] F. Serra-Graells, L. Gómez, and J. L. Heurtas, "A true-1-V 300-μW CMOS-subthreshold log-domain hearing-aid-on-chip," *IEEE J. Solid-State Circuits*, vol. 39, no. 8, pp. 1271–1281, Aug. 2004.

[9] C. H.-I. Kim, H. Soeleman, and K. Roy, "Ultra-low-power DLMS adaptive filter for hearing aid applications," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 11, no. 6, pp. 1058–1067, Dec. 2003.

[10] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "A 16-channel low power non-uniform spaced filter bank core for digital hearing aids," *IEEE Trans. Circuits Syst. II*, vol. 53, no. 9, pp. 853–857, Sep. 2006.

[11] J. S. Chang, and Y. C. Tong, "A micropower-compatible time-multiplexed SC speech spectrum analyzer design," *IEEE J. Solid-State Circuits*, vol. 28, no. 1, pp. 40–48, Jan. 1993.

[12] B. L. Sim, Y. C. Tong, J. S. Chang, and C. T. Tan, "A parametric formulation of the generalized spectral subtraction method," *IEEE Trans. Speech Audio Process.*, vol. 6, no. 4, pp. 328–337, Jul. 1998.

[13] J. M. Kates, "Feedback cancellation in hearing aids: results from a computer simulation," *IEEE Trans. Signal Process.*, vol. 39, no. 3, pp. 553–562, Mar. 1991.

[14] J. Yang, M.-T. Tan, and J. S. Chang, "Modeling external feedback path of an ITE digital hearing instrument for acoustic feedback cancellation," in *Proc. IEEE Int. Sym. Circuits Syst.*, 2005, pp. 1326–1329.

[15] B.-H. Gwee, J. S. Chang, and H. Li, "A micropower low-distortion digital pulsewidth modulator for a digital class D amplifier," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 49, no. 4, pp. 245–256, Apr. 2002.

[16] B.-H. Gwee, J. S. Chang, and A. Victor, "A micropower low distortion class D amplifier based on an algorithmic pulse width modulator," *IEEE Trans. Circuits Syst. I*, vol. 52, no. 10, pp. 2007–2022, Oct. 2005.

[17] E. C. Dijkmans, "Hearing instruments go digital," in *Proc. IEEE European Solid-State Circuits Conf.*, 1997, pp. 16–28.

[18] R. W. Brodersen, M. A. Horowitz, D. Markovic, B. Nikolić, and V. Stojanoiv, "Methods for true power minimization," in *Proc. IEEE Int. Conf. Comput. Aided Design*, 2002, pp. 35–42.

[19] A. P. Chandrakasan, S. Sheng, and R. W. Brodersen, "Low-power CMOS digital design," *IEEE J. Solid-States Circuits*, vol. 27, no. 4, pp. 473–484, Apr. 1992.

[20] J. M. Rabaey, A. Chandrakasan, and B. Nikolić, *Digital Integrated Circuits: A Design Perspective*, 2nd ed. Reading, Upper Saddle River, NJ: Prentice Hall, 2002.

[21] K. Roy, and S. C. Prasad, *Low-Power CMOS VLSI Circuit Design*. Reading, New York: Wiley, 2000.

[22] L. Benini, G. De Micheli, and E. Macci, "Designing low-power circuits: practical recipes," *IEEE Magazine*, vol. 1, no. 1, pp. 6–25, 2001.

[23] T. Gemmeke, M. Gansen, H. J. Stockmanns, and T. G. Noll, "Design optimization of low-power high-performance DSP building blocks," *IEEE J. Solid-State Circuits*, vol. 39, no. 7, pp. 1131–1139, Jul. 2004.

[24] S. L. Mitra, *Digital Signal Processing: A Computer-Based Approach,* 2nd ed. Reading, McGraw-Hill, 2002.

[25] __, *Digital Signal Processing Applications: using the ADSP*-2100 *Family*. Analog Devices, Prentice Hall, Englewood Cliffs, NJ, 1990.

[26] W.-C. Yeh, and C.-W. Jen, "High-speed and low-power split-radix FFT," *IEEE Trans. Signal Process.*, vol. 51, no. 3, pp. 864–874, Mar. 2003.

[27] Y.-W. Lin, H.-Y. Liu, and C.-Y. Lee, "A dynamic scaling FFT processor for DVB-T applications," *IEEE J. Solid-State Circuits*, vol. 39, no. 11, pp. 2005–2013, Nov. 2004.

[28] Y.-W. Lin, H.-Y. Liu, and C.-Y. Lee, "A 1-GS/s FFT/IFFT processor for UWB applications," *IEEE J. Solid-State Circuits*, vol. 40, no. 8, pp. 1726–1735, Aug. 2005.

[29] M. A. Richards, "On hardware implementation of the split-radix FFT," *IEEE Trans. Acoustic Speech Signal Process.*, vol. 36, no. 10, pp. 1575–1581, Oct. 1988.

[30] T. P. Loy, "Low power FFT processor IC," MSc dissertation, Sch. Electrical Electronic Eng., Nanyang Tech. Univ., 2004.

[31] T. J. Ding, J. V. McCanny, and Y. Hu, "Rapid design of application specific FFT cores," *IEEE Trans. Signal Process*., vol. 47, no. 5, pp. 1371–1381, May 1999.

[32] L. Jia, B. Li, Y. Gao, and H. Tenhunen, "Implementation of a low power 128-point FFT," in *Proc. IEEE Int. Conf. Solid-State & Integrated Circuit Tech.*, 1998, pp. 369–372.

[33] B. M. Bass, "A low-power, high-performance, 1024-point FFT processor," *IEEE J. Solid-State Circuits*, vol. 34, no. 3, pp. 380–387, Mar. 1999.

[34] A. Wang, and A. Chandrakasan, "A 180-mV subthreshold FFT processor using a minimum energy design methodology," *IEEE J. Solid-State Circuits*, vol. 40, no. 1, pp. 310–319, Jan. 2005.

[35] Cadence, Inc. Http://www.cadence.com

[36] Synopsys, Inc. Http://www.synopsys.com

[37] Mentor Graphics, Inc. Http://www.mentor.com

[38] J. Sparsø, and S. Furber, *Principles of Asynchronous Circuit Design: A Systems Perspective*. Reading, Kluwer Academic Publishers, 2001.

[39] S. Hauck, "Asynchronous design methodologies: an overview," *Proc. IEEE*, vol. 83, no. 1, pp. 69–93, Jan. 1995.

[40] A. L. Davis, and S. M. Nowick, "An introduction to asynchronous circuit design," UUCS-97-013, 1997.

[41] Y. W. Li, G. Patounakis, K. L Shepard, and S. M. Nowick, "High-throughput asynchronous datapath with software-controlled voltage scaling," *IEEE J. Solid-State Circuits*, vol. 39, no. 4, pp. 704–708, Apr. 2004.

[42] K. S. Stevens *et al*., "An asynchronous instruction length decoder," *IEEE J. Solid-State Circuits*, vol. 36, no. 2, pp. 217–227, Feb. 2001.

[43] A. J. Martin *et al*., "The Lutonium: A sub-nanojoule asynchronous 8051 microprocessor," in *Proc. IEEE Int. Sym. Async. Circuits Syst.*, 2003, pp. 14–23.

[44] J. Kessels, and P. Marston, "Designing asynchronous standby circuits for a low power pager," *Proc. IEEE*, vol. 87, no. 2, pp. 257–267, Feb. 1999.

[45] L. S. Nielsen, C. Niessen, J. Sparsø, and C. H. van Berel, "Low-power operation using self-timed circuits and adaptive scaling of the supply voltage," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 2, no. 4, pp. 391–397, Dec. 1994.

[46] J. V. Woods, P. Day, S. B. Furber, J. D. Garside, N. C. Paver, and S. Temple, "AMULET 1: Asynchronous ARM processor," *IEEE Trans. Comput.*, vol. 46, no. 4, pp. 385–398, Apr. 1997.

[47] G. K. Konstadinidis *et al*., "Implementation of a third-generation 1.1-GHz 64-bit microprocessor," *IEEE J. Solid-State Circuits*, vol. 37, no. 11, pp. 1461–1469, Nov. 2002.

[48] T. Werner, and V. Akella, "Asynchronous processor survey," *IEEE Comput.*, vol. 30, no. 11, pp. 67–76, Nov. 1997.

[49] J. Butas, C.-S. Choy, J. Povazanec, and C-F. Chan, "Asynchronous cross-pipelined multiplier," *IEEE J. Solid-State Circuits*, vol. 36, no. 8, pp. 1272–1275, Aug. 2001.

[50] S. Kim, and R. Sridhar, "Comparison of power consumption among asynchronous design styles with their synchronous counterparts," in *Proc. IEEE Mid West Symp. Circuits Syst.*, vol. 1, 1994, pp.7 –10.

[51] S. B. Furber *et al.*, "AMULET2e: An asynchronous embedded controller," *Proc. IEEE*, vol. 87, no. 2, pp. 243–256, Feb. 1999.

[52] N. R. Poole, "Experiences with asynchronous design methodologies," in *Proc. IEE Coll. Design Test Async Syst.*, pp. 3/1-3/6, 1996.

[53] C. F. Law, "Design methodologies for low power asynchronous logic digital systems," Sch. Electrical Electronic Eng., Nanyang Tech. Univ., 2004.

[54] I. Obridko and R. Ginosar, "Low energy asynchronous adders," in *Proc. IEEE Int. Conf. Elec. Circuits Syst.,* 2004, pp. 164–167.

[55] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "Low-voltage asynchronous adders for low power and high-speed applications," in *Proc. IEEE Int. Sym. Circuits Syst.*, vol. 1, 2002, pp. 873–876.

[56] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "A micropower low-voltage multiplier with reduced spurious switching," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 13, no. 2, pp. 255–265, Feb. 2005.

[57] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "Energy-efficient synchronous-logic and asynchronous-logic FFT/IFFT processors," accepted for publication in *IEEE J. Solid-State Circuits*, Regular Paper, 2007.

[58] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "Design of several asynchronous-logic macrocells for a low-voltage micropower cell library," accepted for publication in *IEE Proc. Circuits Devices Syst.*, 2007.

[59] J. S. Chang, B.-H. Gwee, and K.-S. Chong, "A digital multiplier with reduced switching by means of latch adders," Singapore Patent, 2003, Granted.

[60] J. S. Chang, B.-H. Gwee, and K.-S. Chong, "A digital multiplier with reduced switching by means of latch adders," U.S. Patent, 2007, Granted.

[61] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "Low-voltage micropower multipliers with reduced spurious switching," in *Proc. IEEE Int. Sym. Circuits Syst.*, vol. 4, 2005, pp. 4078–4081.

[62] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "A low energy asynchronous FFT/IFFT processor for hearing aid applications," in *Proc. IEEE Int. Conf. Electron Devices Solid-State Circuits*, 2005, pp. 751–754.

[63] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "A robust low voltage low energy asynchronous carry-completion sensing adder for biomedical applications," in *Proc. IEEE Int. Workshop Biomedical Circuits Syst.,* 2004, pp. S1/2-18–21.

[64] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "A low-voltage low power accumulator," in *Proc. IEEE Int. Sym. Integrated Circuits Devices Syst.*, 2004, pp. F1057–GTJ9.

[65] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "Low-voltage micropower asynchronous multiplier for hearing instruments," in *Proc. IEEE Int. Sym. Circuits Syst.*, vol. 1, 2002, pp. 865–868.

[66] S. M. Kang, and Y. Leblebigi, *CMOS Digital Integrated Circuits: Analysis and Design,* 2nd ed. Reading, McGraw-Hill, 1999.

[67] K. Roy, "Design and test of scaled CMOS circuits," *Tutorial Notes on Int. Sym. Integr. Circuits Devices Syst*., 2004.

[68] K. Roy, S. Mukhopadyay, and H. Mahmoodi-Meimand, "Leakage current mechanisms and leakage reduction techniques in deep-submicrometer CMOS circuits," *Proc. IEEE*, vol. 91, no. 2, pp. 305–327, Feb. 2003.

[69] R. Gonzalez, and M. Horowitz, "Energy dissipation in general purpose microprocessors," *IEEE J. Solid-State Circuits*, vol. 31, no. 9, pp, 1277–1284, Sep. 1996.

[70] A. P. Chandrakasan, "Low power digital CMOS design," Ph.D. thesis, Electrical Eng. & Comput. Science, Univ. California, Berkeley, 1994.

[71] R. Zimmermann, and W. Fichtner, "Low power logic styles: CMOS versus pass-transistor logic," *IEEE J. Solid-State Circuits*, vol. 32, no. 7, pp. 1079–1090, Jul. 1997.

[72] A. Correale, "Overview of the power minimization techniques employed in the IBM PowerPC 4xx embedded controller," in *Proc. ACM/IEEE Int. Sym. Low Power Design*, 1995, pp. 75–80.

[73] A. J. Martin, "The limitations to delay-insensitivity in asynchronous circuits," in *Proc. IEEE/MIT Conf. Adv. Res. VLSI*, 1990, pp. 263–278.

[74] L. Lavagno, and A. Sangiovanni-Vinventelli, *Algorithms for Synthesis and Timing of Asynchronous Circuits*. Reading, Kluwer Academic Publishers, 1993.

[75] A. Peeters, and K. van Berkel, "Single-rail handshake circuits," in *Proc. IEEE Conf. Async. Design Methodologies*, 1995, pp.53–63.

[76] K. Y. Yun, P. A. Beerel, and J. Arceo, "High-performance asynchronous pipeline circuits," in *Proc. IEEE Int. Sym. Adv. Research Async. Circuits Syst.*, 1996, pp. 17–28.

[77] A. Peeters, "Single-rail handshake circuits," PhD Thesis, Eindhoven Univ. Technology, 1995.

[78] I. E. Sutherland, "Micropipelines," *Communications of ACM*, vol. 32, no. 6, pp. 720–738, Jun. 1989.

[79] K. T. Christensen, P. Jensen, P. Korger, and J. Sparsø, "The design of an asynchronous TinyRISC™ TR4101 microprocessor core," in *Proc. IEEE Int. Sym. Adv. Reseach Asycn. Circuits Syst.*, 1998, pp.108–119.

[80] M. Lewis, J. Garside, and L. Brackenbury, "Reconfigurable latch controllers for low power asynchronous circuits," in *Proc. IEEE Int. Sym. Adv. Research Async. Circuits Syst.*, 1999, pp. 27–35.

[81] A. Bardsley, and D. A. Edwards, "Synthesising an asynchronous DMA controller with Balsa," *J. Syst. Architecture*, vol. 46, pp. 1309–1319, 2000.

[82] C. H. van Berkel, J. Kessels, M. Roncken, R. Saeijs, and F. Schalij, "The VLSI-programming language Tangram and its translation into handshake circuits," in *Proc. European Conf. Design Auto*., 1991, pp. 384–289.

[83] A. J. Martin, "Compiling communicating processes into delay-insensitive VLSI circuits," *Distributed Computing*, vol. 1, no. 4, pp. 226–234, 1986.

[84] Handshake Solutions, Inc. Http://www.handshakesolutions.com

[85] The Asynchronous Logic Home Page. Http://www.cs.man.ac.uk/async/

[86] P. Day, and J. V. Woods, "Investigation into micropipeline latch design styles," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 3, no. 2, pp. 264–272, Jun. 1995.

[87] S. B. Furber, and P. Day, "Four-phase micropipeline latch control circuits," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 4, no. 2, pp. 247–253, Jun. 1996.

[88] J. Yuan, and C. Svensson, "High-speed CMOS circuit technique," *IEEE J. Solid-State Circuits*, vol. 24, no. 1, pp. 62–70, Feb. 1989.

[89] N. C. Paver, "Design and implementation of an asynchronous microprocessor," Ph.D. thesis, Depart. Comput. Science, Univ. Manchester, UK, 1994.

[90] A. Efthymiou, and J. D. Garside, "Adaptive pipeline structure for speculation control," in *Proc. IEEE Int. Sym. Async. Circuits Syst.*, 2003, pp. 46–55.

[91] K. S. Stevens *et al*., "CAD directions for high-performance asynchronous circuits," in *Proc. IEEE Digit. Automation Conf*., 1999, pp. 116–121.

[92] K. S. Stevens, R. Ginosar, and S. Rotem, "Relative timing," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 11, no. 1, pp. 129–140, Feb. 2003.

[93] __, "Asynchronous circuit technology: an alternative for high speed semiconductors," Falcrum Microsystems, Inc. Http://www.fulcrummicro.com

[94] S. E. Schuster, and P. W. Cook, "Low-power synchronous-to-asynchronous-to-synchronous interlocked pipelined CMOS circuits operating at 3.3–4.5GHz," *IEEE J. Solid-State Circuits*, vol. 38, no. 4, pp. 622–630, Apr. 2003.

[95] Theseus Logic, Inc. Http://www.theseus.com

[96] Q. K. Zhu, *Power Distribution Network Design for VLSI*. Reading, Wiley, 2004.

[97] D. Johnson, V. Akella, and B. Stott, "Micropipelined asynchronous discrete cosine transform (DCT/IDCT) processor," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 6, no. 4, pp. 129–140, Dec. 1998.

[98] A. Bink, M. de Clercq, and R. York, "White paper: ARM996HS™ processor," Handshake Solution, Inc., 2006.

[99] R. J. Baker, H. W. Li, and D. E. Boyce, *CMOS: Circuit Design, Layout, and Simulation*. Reading, New York: IEEE Press, 1997.

[100] __, *Abstract Generator User Guide*. Cadence, Jun. 2004.

[101] A. M. Shams, T. K. Darwish, and M. A. Bayoumi, "Performance analysis of low power 1-bit CMOS full adder cells," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 10, no. 1, pp. 20–29, Feb. 2002.

[102] N. H. E. Weste, and K. Eshraghian, *Principles of CMOS VLSI Design: A Systems Perspective*. Reading, Addison-Wesly Publising Company, 1993.

[103] M. Sayed, and W. Badawy, "Performance analysis of single-bit full adder cells using 0.18, 0.25, and 0.35μm CMOS technologies," in *Proc. IEEE Int. Sym. Circuits Syst.*, vol. 3, 2002, pp. 559–562.

[104] H. T. Bui, Y. Wang, and Y. Jiang, "Design and analysis of low-power 10-transistor full adders using novel XOR-XNOR gates," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 49, no. 1, pp. 25–30, Jan. 2002.

[105] N. Zhuang, and H. Hu, "A new design of CMOS full adder," *IEEE J. Solid-State Circuits*, vol. 27, no. 5, pp. 840–844, May 1992.

[106] N. F. Goncalves, and H. J. De Man, "NORA: A racefree dynamic CMOS technique for pipelined logic structure," *IEEE J. Solid-State Circuits*, vol. SC-18, no. 3, pp. 261–266, Jun. 1983.

[107] C. H. Chang, J. Gu, and M. Zhang, "A review of 0.18μm full adder performances for tree structured arithmetic circuits," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 13, no. 6, pp. 686–695, Jun. 2005.

[108] C. Lemonds, and S. S. Mehant-Shetti, "A low power 16 by 16 multiplier using transition reduction circuitry," in *Proc. Int. Workshop Low Power Design*, 1994, pp. 139–142.

[109] D. Johnson, and V. Akella, "Design and analysis of asynchronous adders," *IEE Proc. Comp. Dig. Tech.*, vol. 145, no. 1, pp. 1–8, Jan. 1998.

[110] S. Perri, P. Corsonello, and G. Cocorullo, "VLSI circuits for low-power high-speed asynchronous addition," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 10, no. 5, pp. 608–613, Oct. 2002.

[111] F.-C. Cheng, S. H. Unger, and M. Theobald, "Self-timed carry-lookahead adders," *IEEE Trans. Comput*., vol. 49, no. 7, pp. 659–672, Jul. 2000.

[112] G. A. Ruiz, "Evaluation of three 32-bit CMOS adders in DCVS logic for self-timed circuits," *IEEE J. Solid-State Circuits*, vol. 33, no. 4, pp. 604–613, Apr. 1998.

[113] J. D. Garside, "A CMOS VLSI implementation of an asynchronous ALU," *IFIP*, 1993, pp. 181–192.

[114] D. Soudris, C. Piguet, and C. Goutis, *Designing CMOS Circuits for Low Power*. Reading, Kluwer Academic Publishers, 2002.

[115] D. J. Kinniment, "An evaluation of asynchronous addition," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 4, no. 1, pp. 137–140, 1996.

[116] K.-S. Chong, "Analysis and design of micropower asynchronous adders," M.Phil Thesis, Sch. Electrical and Electronic Eng., Nanyang Tech. Univ., 2001.

[117] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "A low power 16-bit Booth leapfrog array multiplier using dynamic adders," in *Proc. IEEE Int. Sym. Circuits Syst.*, vol. 2. 1, 2004, pp. 437–440.

[118] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "Low energy 16-bit Booth leapfrog array multiplier using dynamic adders," accepted for publication in *IEE Proc. Circuits Devices Syst*, 2007.

[119] I. Koren, *Computer Arithmetic Algorithms*. Prentice Hall, 1993.

[120] B. Parhami, *Computer Arithmetic: Algorithms and Hardware Designs*. Reading, New York: Oxford Univ. Press, 2000.

[121] C. S. Wallace, "A suggestion for fast multipliers," *IEEE Trans. Comput.*, vol. 13, pp. 14–17, Feb. 1964.

[122] V. G. Oklobdzijia, D. Villeger, and S. S. Liu, "A method for speed optimized partial product reduction and generation of fast parallel multipliers using an algorithmic approach," *IEEE Trans. Comput.*, vol. 45, no. 3, pp. 294–306, Mar. 1996.

[123] G. Goto, T. Sato, M. Nakajima, and T. Sukemura, "A 54×54-b regularly structured tree multiplier," *IEEE J. Solid-State Circuits*, vol. 27, no. 9, pp. 1229–1235, Sep. 1992.

[124] K. Yano *et al.*, "A 3.8-ns CMOS 16×16-b multiplier using complementary pass-transistor logic," *IEEE J. Solid-State Circuits*, vol. 25, no. 2, pp. 388–394, Apr. 1990.

[125] N. Ohkubo *et al.*, "A 4.4ns CMOS 54×54-bit multiplier using pass-transistor multiplexer," *IEEE J. Solid-State Circuits*, vol. 30, no. 3, pp. 251–257, Mar. 1995.

[126] P. C. H. Meier, R. A. Rutenbar, and L. R. Carley, "Exploring multiplier architecture and layout for low power," in *Proc. IEEE Custom Integrated Circuits Conf.*, 1996, pp. 513–516.

[127] S. S. Mahant-Shetti, P. T. Balsara, and C. Lemonds, "High performance low power array multiplier using temporal tiling," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 7, no. 1, pp. 121–124, Mar. 1999.

[128] E. de Angel, "Ultra low power multiplier," U.S. Patent 5,787,029, Jul. 28, 1998.

[129] F. Lu, and H. Samueli, "A 200-MHz CMOS pipelined multiplier-accumulator using a quasi-domino dynamic full adder cell designs," *IEEE J. Solid-State Circuits*, vol. 28, no. 2, pp. 123–132, Feb. 1993.

[130] G. E. Sobelman, and D. L. Raatz, "Low-power multiplier design using delay evaluation," in *Proc. IEEE Int. Sym. Circuits Syst.*, vol. 3, 1995, pp. 1564–1567.

[131] T. Sakuta, W. Lee, and P. T. Balsara, "Delay balanced multipliers for low power/ low voltage DSP core," in *Proc. IEEE Sym. Low Power Electronics*, 1995, pp. 36–37.

[132] W.-C. Yeh, and C.-W. Jen, "High-speed Booth encoded parallel multiplier design," *IEEE Trans. Comput.*, vol. 49, no. 7, pp. 692–701, Jul. 2000.

[133] O. T. C. Chen, S. Wang, and Y. W. Wu, "Minimization of switching activities of partial products for designing low-power multipliers," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 11, no. 3, pp. 418–433, Jun. 2003.

[134] Y. C. Lim, "Single precision multiplier with reduced circuit complexity for signal processing applications," *IEEE Trans. Comput*., vol. 41, no. 10, pp. 1333–1336, Oct. 1992.

[135] S. S. Kidambi, F. El-Guibaly, and A. Antoniou, "Area-efficient multipliers for digital signal processing applications," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 43, no. 2, pp. 90–95, Feb. 1996.

[136] A. D. Booth, "A signed binary multiplication technique," *Quarterly J. Mechanics Applied Math*., vol. 4, no. 2, pp. 236–240, Jun. 1951.

[137] S. K. Rao, "Multiplier circuit," U.S. Patent 5,038,315, Aug. 6, 1991.

[138] K. S. Stevens, and B. W. Suter, "A mathematical approach to a low power FFT architecture," in *Proc. IEEE Int. Sym. Circuits Syst*., vol. 2, 1998, pp. 21–24.

[139] I. Sutherland, B. Sproull, and D. Harris, *Logical Effort: Designing Fast CMOS Circuits*. Reading, San Francisco: Morgan Kaufmann Publishers, Inc., 1999.

[140] D. A. Hodges, H. G. Jackson, and R. A. Saleh, *Analysis and Design of Digital Integrated Circuits in Deep Submicron Technology*. Mc-Graw Hill, 2004.

[141] W.-H. Chang, and T. Nguyen, "An OFDM-specified lossless FFT architecture," *IEEE Trans. Circuits Syst. I*, vol. 53, no. 6, pp. 1235–1243, Jun. 2006.