Low Power Asynchronous-Logic Circuit Design

by

Bah-Hwee Gwee
Associate Professor, Nanyang Technological University
Senior Consultant, Digital Hearing Aid Centre
Founder & Director, Advanced Electroacoustics Pte Ltd
ebhgwee@ntu.edu.sg
Outline

- Motivations
- Background
- Asynchronous-logic
- Our Group and Projects
- Conclusions
- Future Projects
Motivations

1. Huge Demand for Biomedical Electronics
   - Human needs due to health, aging, better quality of life (QoL) considerations
   - Wide range of applications, e.g. hearing aids (instruments), EEG, health-monitoring bio-sensors, wearable biomedical systems, implantable devices including pacemakers, vision processors (cornea), etc.
   - Sales – estimated US $66b in 2010 (*INEMI 2007)

*INEMI – International Electronic Manufacturing Initiatives
2. Power-/Energy-Critical Requirements for Biomedical Devices
   - Long battery life-span (e.g. pacemaker 10-12 years)
   - Low energy capacities (e.g. < 1mA for hearing aids)
   - Associated small size requirement (e.g. miniature battery is used)

3. Reliability Issues
   - Operation robustness
   - Insensitive to variations due to process, voltage, and temperature (PVT)
   - Reduced timing assumptions (or clock issues)

4. Need of Ultra Low Power Digital Circuits
   - ASIC designs
   - Microprocessors and DSPs
Background

Technology Roadmap

- Red: Power density
- Green: Clock frequency
- Blue: Number of cores
- Yellow: Process variations

Dotted lines: ‘Power Aware’ Scaling
Solid lines: Classical Scaling

Year:
- 2004

Process:
- 180nm
- 130nm
- 90nm
- 65nm
- 45nm

Time/Process
Background

Computing Trend and Challenges

High-performance computing

Ultra High-performance:
- PCs, Servers, Network Routers, etc.
- High-performance:
  - PCs, notebooks, DVDs, Imaging Processors, Cellular Phones, etc.

Low Power Dissipation:
- Audio Devices, Portable Consumer Electronics, Controllers, etc.

Ultra Low Power Dissipation:
- Bio-medical Devices, Hearing Aids, Sensors, etc.

Two common challenges for ultra low power devices:
1. Limited energy capacities (small power density)
2. Process, temperature, voltage (PVT) variabilities
### Background

Some energy source examples

<table>
<thead>
<tr>
<th>Battery</th>
<th>Capacity</th>
<th>Application</th>
<th>Duration</th>
<th>Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>A13</td>
<td>100mA.h</td>
<td>Hearing aids</td>
<td>100 hours</td>
<td>&lt;1mA</td>
</tr>
<tr>
<td>UB6120</td>
<td>12A.h</td>
<td>Pacemaker</td>
<td>10 years</td>
<td>&lt;0.15mA</td>
</tr>
</tbody>
</table>

Other Energy Source | Power Density  
---                | ---            
Solar cell (indoor) | 3.2μW/cm²      
Solar cell (direct sunshine) | 3700μW/cm² 
Vibrations           | 5 – 500μW/cm²  

*Energy is limited!!*
45nm CMOS transistors can switch faster… But higher margins are required!!
More and more timing margins required due to complexity of the design!!
Asynchronous-Logic (Async-Logic) : Review
Async-logic : Review

Brief Time Line

1950s – Digital circuits were implemented (either async or sync)

1960s – Sync digital circuits became dominant

1989 – • Sutherland (Sun Microsystems) reignited the interests in async designs
  • First async microprocessor was designed (Martin, Caltech)

1990s – • Many university research groups focused on async designs
  • Big firms (Intel, Philips, IBM, etc.) joined in
  • Philips introduced first async commercial chip in pagers
  • Start-up company: Theseus Logic

2000s – Start up companies: Fulcrum Microsystems, Handshake Solutions and Self-Timed Solutions

Future – An alternative solution in many ICs, as predicted by International Technology Roadmap for Semiconductors
Async-logic: Review

Fundamental difference between sync and async approaches

Sync (clocked-based) approach

VS

Async (handshake) approach
Async-logic : Review

Sync operation

Diagram showing a synchronous operation with data inputs, logic gates, and registers.
Async operation

*HCC: Handshake control circuits
Async-logic: Review

Good and Bad for async-logic

Good (Potential advantages)

1. No (or little) clock-related issues (e.g. clock skews, etc.)
2. Low power dissipation
3. Faster speed performance
4. Robust towards PVT variations
5. High modularity
6. Many more ...

Bad (so far)

1. Lack of EDA tools
2. Lack of methodologies (educational training is insufficient)
3. Some bad design experiences (in the past)
4. Difficult in testing
Our Group

NTU Team:

1. Dr Joseph S. Chang (PI)
2. Dr Gwee Bah Hwee (PI)
3. Dr Chong Kwen Siong (Research Scientist)
4. Law Chong Fatt (Research Associate, PhD thesis under examination)
5. Chang Kok Leong (PhD, A*Star Scholarship)
6. Shi Yiqiong (PhD, NTU Scholarship)
7. Lin Tong (PhD, President Scholarship)

Collaborators:

1. Prof. Alain Martin (Caltech, USA)
2. Prof. Lars Wanhammar (Likoping Uni, Sweden)
3. DSO (Temasek Laboratories @ NTU)
4. Prof Ser Wei (CSP @ NTU)
Our Projects

Primary Application: Digital Hearing Aid (Instrument)

General block diagram in a hearing aid
Project 1: Async FFT Processor

Objective: To develop a low power async Fast Fourier Transform (FFT) processor (as a filter bank) for digital hearing aid applications.

FFT Algorithm

\[
x(k) = \sum_{n=0}^{N-1} x(n) \cdot W_{nk}^{n}\n\]

\[
W_{nk}^{n} = e^{-j2nk\pi / N} = \cos(2nk\pi / N) - j\sin(2nk\pi / N)
\]

Radix-2 butterfly (core computation for FFT)

[Diagram of Radix-2 butterfly]
Basic Features:

- 16-bit data, 128-point radix 2 format
- Semi-custom/full-custom approaches
- Commercial EDA tools used
- Async state-machine concepts (for controllers) with async handshake
- Ultra low power library cells developed (for datapaths)
- Fine-grain gating (innately controlled by async handshake)
- 4-phase, hybrid single-rail & dual-rail implementation
- Simple data flows with 2 multipliers, 3 adders (see data flows below)

(a) computing the real signals
(b) computing the imaginary signals
Project 1: Async FFT Processor

Block diagram of the async FFT processor
Project 1: Async FFT Processor

IC realization for the proposed async FFT processor

IC realization for the benchmarked sync FFT processor
Result: Energy dissipation per FFT operation from 1.1V to 1.4V

![Graph showing energy dissipation]

- Sync FFT Processor
- Async FFT Processor

~ 40% lower energy!
Result: Characteristics of the FFT Processors

<table>
<thead>
<tr>
<th></th>
<th>Benchmarked sync FFT processor</th>
<th>Proposed async FFT processor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delay*</td>
<td>~ 1.215 ms @ 1MHz</td>
<td>~ 1.2 ms</td>
</tr>
<tr>
<td>Energy*</td>
<td>~ 188 nJ</td>
<td>~ 120 nJ</td>
</tr>
<tr>
<td>IC Area</td>
<td>~ 1.44mm$^2$ @ 0.35um CMOS</td>
<td>~ 1.6mm$^2$ @ 0.35um CMOS</td>
</tr>
</tbody>
</table>

* Based on one complete FFT computation @ 1.1V
Our Project 2: Async 8051 Microcontroller

Objective: To develop a low power async 8051 microcontroller for digital hearing aid applications

Basic features:

- Harvard Architecture
- 1-bit operations
- Abundant development tools and benchmarks
- Non-orthogonal instruction set (different instruction lengths and cycle times)
- Control-dominated applications
- 8051 (a CISC) contains numerous exceptions, which must be traded-off by margining, if designing using synchronous logic, it is not the case for asynchronous logic
- 8051 core is still a popular processor for ubiquitous computing
Methodology adopted, in part, by async Balsa tool
Block Diagram of the async 8051 microcontroller

- 4kByte ROM
- 128×8 Random access Memory (RAM)
- 128Byte RAM
- Main pipeline
- Instruction Pointer (IP)
- Instruction Fetch (IF)
- Decode and Execute (D&X)
- Arithmetic and Logic Unit (ALU)
- Asynchronous data channels
- Asynchronous handshake channels

1st stage pipeline

2nd stage pipeline
Read-Only-Memory IP Interface

The channels addr and Q are asynchronous.
- Interface circuits are added to ‘synchronize’ the asynchronous channels.
- Timing assumption for addr_a+ → Q_a+ (α)

α: Worst case latency
β: Actual latency
Random-Access-Memory IP Interface

- The channels RAMctrl and Q are asynchronous
- A, D and WEN are bundled in RAMctrl
- Timing assumption for RAMctrl_a+ → Q_a+ (α) during the read cycle
Project 2: Async 8051 Microcontroller

Results: Energy x Delay\(^2\) (Et\(^2\)) on different designs

Matched-Delay

QDI

IC Layout for the proposed QDI Async 8051 microcontroller and benchmarked sync 8051 microcontroller (under fabrication)

Design Name: IBM_8051_0.13um

Technology: 0.13um IBM CMOS (DM) Process

Area: 2.027mm x 2.027mm = 4.108729mm^2

Package: LCC84M

Number of I/Os: 84

Synopsys DW8051 IP core

QDI Asynchronous 8051 core
Our Project 3: Async 56002 DSP

Objective: To develop a low power async Motorola 56002 DSP for digital hearing aid applications

Block Diagram of the async 56002 DSP
Preliminary result: Synchronous and asynchronous Data Arithmetic and Logic Units (DALUs) – fabricated in May 08

Block diagram

Microphotograph of the DALUs and its package
Our Project 4: Verilog HDL Asynchronous Compiler

Objective: To develop a low power async EDA tool

Basic features:

- Low control overheads considerations
- Low power dissipation
- Argumentation with existing sync and async EDA tools
- Standard HDL languages
- Additional semantics for async components
- Intelligent translation for async components
- Network optimization
- Easy of design (virtually transparent to designers for async handshake)
Proposed design methodology

- Verilog HDL model
  - Channel list
  - Compilation
  - Semi-structural RTL model
    - Logic synthesis
      - Gate-level netlist
        - Timing analysis
          - Channel delays
            - Delay matching
              - Delay-matched gate-level netlist
    - Handshake circuit library
  - Place & Route

Our contribution to design flow
Compilation example

• **Step 1**
  – Extract all causal relations (i.e., control and data paths).
  – Determine sources and destinations of extracted paths.
  – Establish corresponding channels.

  ![Diagram showing causal relation](image)

• **Step 2**
  – Determine the nature of each channel (such as **push** or **pull**, **data** or **control**, **guarded**).

  ![Diagram showing channel nature](image)
Compilation example

• **Step 3**
  – Elaborate established channels into explicit handshake signals (i.e. request and acknowledge lines).
  – Infer handshake components and instantiate them.

• **Step 4**
  – Rewrite part of specification to establish the handshake components’ control over their respective datapath components.
Design example: Reed-Solomon error decoder

- In general, provide correction if $2E + R \leq P$, where $E$, $R$, and $P$ denote number of errors, erasures, and parity symbols, respectively.
- Implemented version is based on DCC Error Corrector by Philips Research Laboratories and correct only 1 symbol.
- Accepts code words of 28 or 32 8-bit symbols, including 4 or 6 parity symbols, respectively.
- Suitable for async implementation: error detection block is automatically powered down when code word is correct.
Design example: Reed-Solomon error decoder

- Async control network

![Diagram of Reed-Solomon error decoder](image.png)
Design example: Reed-Solomon error decoder

- **Comparisons**
  - Process: AMS 0.35 μm CMOS
  - Supply Voltage: 3.3 V
  - Simulator: Synopsys Nanosim
  - Design Style: standard cell, 4-phase handshaking, delay-matching
  - Test Scenario: 95% correct code words, 5% incorrect code words (error at first symbol)

<table>
<thead>
<tr>
<th></th>
<th>Balsa Implementation</th>
<th>Our Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td># Trans. (k)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Control</td>
<td>12.3</td>
<td>0.72</td>
</tr>
<tr>
<td>Datapath</td>
<td>19.6</td>
<td>5.27</td>
</tr>
<tr>
<td>Total</td>
<td>31.9</td>
<td>5.99</td>
</tr>
<tr>
<td>Throughput (mega codeword per sec)</td>
<td>0.68</td>
<td>5.59</td>
</tr>
<tr>
<td>Energy per codeword (nJ)</td>
<td>Control</td>
<td>19.2</td>
</tr>
<tr>
<td></td>
<td>Datapath</td>
<td>9.00</td>
</tr>
<tr>
<td></td>
<td>Total</td>
<td>28.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.35</td>
</tr>
</tbody>
</table>
Conclusions

- We have conducted 3 async IC design projects and 1 async EDA tool project
- We have demonstrated that async-logic has lower power attribute (over the standard sync-logic)
- We have shown that async-logic is highly an alternative solution for low power bio-medical applications