System Designs

- Introduction
  - Methodology and Framework
- Components
- Specification
- Implementation
Introduction

- Methodology
  - Approach with success stories.
  - Hierarchical designs with interface between the modules (BSV).
- Data Subsystem and Control Subsystem
  - For n-bit data, each operation takes n times or more in hardware complexity.
  - Data subsystem carries out the data operations and transports.
  - Control system sequences the data subsystem and itself.

I. Introduction: Framework
4-Phase Handshaking

A Hello, Could I

Speak to B

This is B

B speaking

Thank you for your time.

You are welcome!
I. Introduction: Handshaking

Handshaking

---

Master that calls module S

Start/Request

Data Inputs

Data Outputs

Control Inputs

Control Outputs

Control Signals

Conditions

Done/Acknowledgment

module S

start

done

X

Y

Z

X

Y

Z

---

Z is ready

Z is not valid

X, Y ready

X, Y not ready

start
done

t
	t
4-Phase Handshaking

Host → Module S

Start → Ack (done)

Start → t

Ack → t

Host: Sender wants to start the operation.

Raises Start = 1

Mod S: Receiver detects Start = 1.

Raises Ack = 1, done = 0

Host: Sender detects Ack = 1.

Lowers Start = 0

Mod S: Receiver detects Start = 0.

Lowers Ack = 0
Handshaking

Handshaking: Example

The master module that calls module $S$ controls the following signals
A. Signal start
B. Signal done
C. Signals start and done
D. None of the above
## II. Components

<table>
<thead>
<tr>
<th>Components</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Subsystem</strong></td>
<td>Data storage</td>
</tr>
<tr>
<td>Storage Modules</td>
<td>Data operations</td>
</tr>
<tr>
<td>Operators ALU, INC</td>
<td>Data transport</td>
</tr>
<tr>
<td>Interconnections</td>
<td>Compare, True, False, Overflow</td>
</tr>
<tr>
<td>Wires, Decoders, Muxes</td>
<td>Control of data operations</td>
</tr>
<tr>
<td>Conditions</td>
<td>Control of data transports</td>
</tr>
<tr>
<td>Condi</td>
<td>Control of the sequential system</td>
</tr>
<tr>
<td>Control Signals</td>
<td></td>
</tr>
<tr>
<td><strong>Control Subsystem</strong></td>
<td></td>
</tr>
<tr>
<td>Sequential machines</td>
<td></td>
</tr>
</tbody>
</table>

## II. Data Subsystem Components

- **Storage**: Register, RAM, FIFO, LIFO, Counter, Shifter
- **Operator**: ALU, Floating Point Operators
- **Interconnect**: Wire, Buses, Crossbars
II. Components: Storage Modules, Register

LD: Load
CLR: Clear

\[
\begin{align*}
Q(t+1) &= (0, 0, \ldots, 0) & \text{if CLR} = 1 \\
&= D & \text{if LD} = 1 \text{ and CLR} = 0 \\
&= Q(t) & \text{if LD} = 0 \text{ and CLR} = 0
\end{align*}
\]

Modulo-\(n\) Counter

\[
\begin{align*}
Q(t+1) &= (0, 0, \ldots, 0) & \text{if CLR} = 1 \\
&= D & \text{if LD} = 1 \text{ and CLR} = 0 \\
&= (Q(t)+1) \mod n & \text{if LD} = 0, \text{ CNT} = 1 \text{ and CLR} = 0 \\
&= Q(t) & \text{if LD} = 0, \text{ CNT} = 0 \text{ and CLR} = 0
\end{align*}
\]

TC

\[
\begin{align*}
TC &= 1 & \text{if } Q(t) = n-1 \text{ and } CNT = 1 \\
&= 0 & \text{otherwise}
\end{align*}
\]
Storage Component: Registers, Array of Registers

Registers: If $C$ then $R \leftarrow D$

Register Array: If $C$ then $R_{address} \leftarrow D$
Sharing connections and controls

Storage Components: RAM, FIFO, LIFO

RAM

Address

Decoder

RAM

Size of RAM larger than registers
Performance is slower

FIFO (First in first out)

used as a buffer.

LIFO (Last in first out: Stack)

Stack.
Functional Modules

CASE Op-Sel Is
When F1, Z <= A op1 B
When F2, Z <= A op2 B
.
End CASE

Example:
CASE Op-Set Is
Z <= (A + B)mod 2^n if Op-Sel=addition,
Z <= (A - B)mod 2^n if Op-Sel=subtraction
End CASE

Interconnect Modules (Wires and Switches)

• Single Lines
• Band of Wires
• Shared Buses
• Crossbar

1. Single line (shifting, time sharing)
2. Band of Wires (BUS)

3. Shared Bus

Switches

4. Crossbar (Multiple buses running horizontally)
   m simultaneous transfers are possible, but more expensive.
Crossbar Switches: VLSI Design

Multistage Network: Clos Network

When $m = n$, we can connect left to right.
Ex. If C then R1 ← S1
   Else R2 ← S2
   Endif;

If C1 then X ← A
Else X ← B + C
Endif
If C2 then Rg ← X
Endif

System Designs

• Introduction
  – Methodology and Framework
• Components
• Specification
• Implementation
VI. Implementation

- Example
- Handshaking
  - Request and Acknowledgement
- Datapath Subsystem
  - Data Operators
  - Data Transporters
- Control Subsystem
  - One Hot Machine Design

VI. Implementation: Example

AddModule(X, Y, Z, start, done)
{ Input X[15:0], Y[15:0] type bit-vector, start type Boolean;
  Local-Object A[15:0], B[15:0] type bit-vector;
  Output Z[15:0] type bit-vector, done type Boolean;
  S0: If start’ goto S0 || done ⇐1;
  S1: A ⇐ X || B ⇐ Y || done ⇐0;
  S2: Z ⇐ Add(A, B) || goto S0;
}
Exercise: Go through the handshaking, data subsystem and control subsystem designs.
AddModule(X, Y, start, done)

Example

Suppose that each step (Si) takes one clock cycle. How many clock cycles does the AddModule take to complete one handshaking iteration?
A. One cycle
B. Two cycles
C. More than two cycles
III. Specification: Program

1. Objects (Registers, Outputs of combinational logic)
2. Operation (Logic, Add, Multiplication, DSP, and etc.)
3. Assignment
4. Sequencing

Example:

Signal S1, S2, R[15:0]: FFs, Registers, wires
Z \leftarrow A + B: Registers, Adder, Interconnect
R1 \leftarrow R2: Registers and Interconnect
Begin, End: Control
if ( ) then ( ), ENDIF: Control