CSE 140 Lecture 14
System Designs
CK Cheng
CSE Dept.
UC San Diego
System Designs

• Introduction
• Components
• Spec
• Implementation
Introduction

• Methodology
  • Approach with success stories.
  • Hierarchical designs with interface between the levels.

• 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

Data Subsystem

Control Subsystem

Conditions

Control Signals

Data Inputs

Control Inputs

n=64

Start/Request

Done/Acknowledgement

Data Outputs

Control Outputs

n=64

Inputs

Outputs
# Introduction

<table>
<thead>
<tr>
<th>Data Subsystem</th>
<th>Components</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Storage Modules</td>
<td>Data storage</td>
</tr>
<tr>
<td></td>
<td>Operators</td>
<td>Data operations</td>
</tr>
<tr>
<td></td>
<td>Interconnections</td>
<td>Data transport</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Control Subsystem</th>
<th>Sequential machines</th>
<th>Control of data operations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Control of data transports</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Control of the sequential system</td>
</tr>
</tbody>
</table>
Data Subsystem Components

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

LD: Load
CLR: Clear

Q(t+1) = (0, 0, .. , 0) if CLR = 1
= D if LD = 1 and CLR = 0
= Q(t) if LD = 0 and CLR = 0
Storage Component: Registers, Array of Registers

Registers: If C then R $\leftrightarrow$ D

Register Array: If C then $R_{\text{address}}$ $\leftrightarrow$ D

Sharing connections and controls
Storage Components: RAM, FIFO, LIFO

RAM

Decoder
Address

RAM

Size of RAM larger than registers
Performance is slower

FIFO (First in first out)

LIFO (Last in first out: Stack)
Modulo-n Counter

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

\[ TC = \begin{cases} 
1 & \text{if } Q(t) = n-1 \text{ and } CNT = 1 \\
0 & \text{otherwise}
\end{cases} \]
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)

![Diagram showing shift registers connected]
2. Band of Wires (BUS)

3. Shared Bus

Switches

MUX DEMUX

Switches

MUX DEMUX
4. Crossbar (Multiple buses running horizontally) in simultaneous transfers are possible, but more expensive.
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 ( ), ENDF: Control
Ex. If C then R1 $\leftarrow$ S1
Else R2 $\leftarrow$ S2
Endif;

If C1 then X $\leftarrow$ A
Else X $\leftarrow$ B + C
Endif
If C2 then G $\leftarrow$ X
Endif
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)
Hand Shaking