Lecture 15

RTL Design Methodology

SORTING example
Structure of a Typical Digital System

Data Inputs

Datapath (Execution Unit)

Control Inputs

Controller (Control Unit)

Data Outputs

Control Outputs

Data Inputs

Control Signals

Status Signals

Control Inputs
Hardware Design with RTL VHDL

- Pseudocode
- Interface
- Datapath
  - Block diagram
  - VHDL code
- Controller
  - ASM chart
  - VHDL code
Steps of the Design Process

1. Text description
2. Interface
3. Pseudocode
4. Block diagram of the Datapath
5. Interface divided into Datapath and Controller
6. ASM chart of the Controller
7. RTL VHDL code of the Datapath, Controller, and Top-Level Unit
8. Testbench for the Datapath, Controller, and Top-Level Unit
9. Functional simulation and debugging
10. Synthesis and post-synthesis simulation
11. Implementation and timing simulation
12. Experimental testing using FPGA board
Steps of the Design Process
Introduced in Class Today

1. Text description
2. Interface
3. Pseudocode
4. Block diagram of the Datapath
5. Interface divided into Datapath and Controller
6. ASM chart of the Controller
7. RTL VHDL code of the Datapath, Controller, and Top-level Unit
8. Testbench for the Datapath, Controller, and Top-Level Unit
9. Functional simulation and debugging
10. Synthesis and post-synthesis simulation
11. Implementation and timing simulation
12. Experimental testing using FPGA board
SORTING example
Sorting - Required Interface

Sort

Clock
Resetn
DataIn
RAdd
WrInit
S
(0=initialization 1=computations)
Rd

N
L

DataOut
Done
# Sorting - Required Interface

<table>
<thead>
<tr>
<th>Port</th>
<th>Width</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clock</td>
<td>1</td>
<td>System clock</td>
</tr>
<tr>
<td>Resetn</td>
<td>1</td>
<td>System reset – clears internal registers. Active low.</td>
</tr>
<tr>
<td>DataIn</td>
<td>N</td>
<td>Input data bus</td>
</tr>
<tr>
<td>RAdd</td>
<td>L</td>
<td>Address of the internal memory where input data is stored</td>
</tr>
<tr>
<td>WrInit</td>
<td>1</td>
<td>Synchronous write control signal</td>
</tr>
<tr>
<td>s</td>
<td>1</td>
<td>Operating mode: 0 = initialization, 1 = computations.</td>
</tr>
<tr>
<td>Rd</td>
<td>1</td>
<td>Read enable. 0 = high impedance on the output bus, 1 = valid output on the output data bus.</td>
</tr>
<tr>
<td>DataOut</td>
<td>N</td>
<td>Output data bus used to read results</td>
</tr>
<tr>
<td>Done</td>
<td>1</td>
<td>Asserted when all results are ready</td>
</tr>
</tbody>
</table>
Simulation results for the sort operation (1)
Loading memory and starting sorting
Simulation results for the sort operation (2)
Completing sorting and reading out memory
## Sorting - Example

### Before sorting

<table>
<thead>
<tr>
<th>Address</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### During Sorting

<table>
<thead>
<tr>
<th></th>
<th>i=0</th>
<th>i=0</th>
<th>i=0</th>
<th>i=1</th>
<th>i=1</th>
<th>i=2</th>
</tr>
</thead>
<tbody>
<tr>
<td>j=1</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>j=2</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>j=3</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
</tr>
</tbody>
</table>

### After sorting

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

**Legend:**
- Position of memory indexed by **i**:
  - \( M_i \)
- Position of memory indexed by **j**:
  - \( M_j \)
Pseudocode

FOR k = 4
[load input data]
wait for s=1
for i = 0 to 2 do
  A = M_i;
  for j = i + 1 to 3 do
    B = M_j;
    if B < A then
      M_i = B;
      M_j = A;
      A = M_i;
    endif;
  endfor;
endfor;
Done
wait for s=0
[read output data]
go to the beginning

FOR any k ≥ 2
[load input data]
wait for s=1
for i = 0 to k-2 do
  A = M_i;
  for j = i + 1 to k - 1 do
    B = M_j;
    if B < A then
      M_i = B;
      M_j = A;
      A = M_i;
    endif;
  endfor;
endfor;
Done
wait for s=0
[read output data]
go to the beginning
Pseudocode

wait for s=1
for i=0 to k-2 do
  A = M_i
  for j=i+1 to k-1 do
    B = M_j
    if A > B then
      M_i = B
      M_j = A
      A = M_i
    end if
  end for
end for
end for
Done
wait for s=0
go to the beginning
Block diagram of the Datapath