CMPT 295
Unit – Processor Design & Instruction Execution
Lecture 27 – Combinational and Sequential Logic Circuits
Last Lecture

- Now that we have had a look at components of an instruction set architecture (ISA), we are now to explore aspects of an instruction set architecture (ISA) from a processor’s perspective.

- Our goal: understand how the processor executes machine instructions and understand issues related to the speed of this execution.

  - We started having a closer look at the processor itself:
    - Made of resistors, capacitors, diodes, and transistors.
    - Billions of them, so understanding their functioning is too onerous.
    - So we resort to abstraction (black box) in order to understand their functioning.
    - Logic gates: perform a Boolean function.
    - Propagation delay.
Today’s Menu

- Instruction Set Architecture (ISA)
  - Definition of ISA
  - ISA design
  - ISA evaluation
    - Improving our ISA -> Decreasing effect of von Neumann bottleneck
    - 3 Strategies

- Execution of machine instructions
  - Intro to logic design, combinational and sequential logic circuits
  - Sequential execution of machine instructions
  - Pipelined execution of machine instructions
From our last lecture - Digital circuits

- To satisfy our goal, we need to have a look at the components of a processor and how they function
  - Combinational logic -> compute functions on bits
  - Memory elements -> store bits
  - Clock signals -> regulate update of memory elements

and what affects their execution speed such as propagation delay

- So, we need to understand a few things about digital circuits
Combinational logic circuits - 1

- Made by connecting several logic gates together
- Compute more complex functions than just AND or XOR
- Example: Combinational logic circuit that adds 2 bits together

Propagation delay: “1 gate” delay
Combinational logic circuits - 2

- Can connect several combinational logic circuits together to perform more complex functions and/or to perform function on a wider input
- What does this circuit do?
- Propagation delay: “3 gate” delay
Can connect several combinational logic circuits together as black boxes (black box design can be modular).

The resulting circuit can operate on word size data such as integers, memory addresses, instruction codes.

Example: **Full Adder** operating on date of word size \( (w) = 4 \).
where ...
Why is the propagation delay of this Full Adder 4 (also known as a 4-bit (w-bit) ripple carry adder) 9 gates (2w + 1 gates) where w is the number of bits?
ALU - Example of combinational logic circuit

- Arithmetic/logic unit found in processor
  - Black box diagram
- ALU involved in executing *data manipulation instructions* (from ISA)

<table>
<thead>
<tr>
<th>S</th>
<th>operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0001</td>
<td>A + B</td>
</tr>
<tr>
<td>0010</td>
<td>A - B</td>
</tr>
<tr>
<td>1000</td>
<td>A * B</td>
</tr>
<tr>
<td>0100</td>
<td>A ^ B</td>
</tr>
<tr>
<td>0101</td>
<td>A &amp; B</td>
</tr>
</tbody>
</table>
Summary: Combinational logic circuits

- **Definition**: A combinational logic circuit computes a function, where its outputs based only on its inputs
- Contains logic gates
  - Input values propagate through logic gates of combinational logic circuit whenever they change
  - Input to a logic gate must come from either input of combinational logic circuit itself (input to black box) or output of other logic gate
  - Outputs of two or more logic gates cannot be connected together unless it is via a logic gate
- Acyclic: there are no feedback loops
- Does not require a clock signal, does not contain memory elements
From our last lecture - Digital circuits

To satisfy our goal, we need to have a look at the components of a processor and how they function:

- Combinational logic -> compute functions on bits
- Memory elements -> store bits
- Clock signals -> regulate update of memory elements

and what affects their execution speed such as propagation delay.

So, we need to understand a few things about digital circuits.
Sequential logic circuits

- Has a state -> memory
- Made by connecting several logic gates together and memory elements
- “Clocked”
- Perform computations on its state
Memory elements and Clock signals

- Memory element #1 -> registers (clocked registers – hardware registers)
  - On the CPU
  - Store n bits (state)
  - Synchronized by system-wide clock
- How clocked registers work:
  - Output current state
  - Input next state
  - Next state remembered only on rising edge of clock
- System-wide clock
  - A system-wide clock sends 0 1 0 1 0 1 0 1...
  - Clock period: 1 full cycle duration:
  - Clock frequency: 1/period
Timing

If clock period, then clock frequency is

<table>
<thead>
<tr>
<th>Clock</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$t_{pd}$ for register

Function Table

<table>
<thead>
<tr>
<th>Load</th>
<th>D</th>
<th>Q(t + 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>X</td>
<td>Q(t)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Memory elements #2 -> Random access memories

- Or memories
- Use an address to select which memory word should be read or written
- Examples:
  1. Virtual memory (with various segments) -> M
  2. Register file -> register id used as address
     - program registers
     - For example: x86-64 register file holds 16 reg., our x295+ and x295++ have 8 registers
Put together:

- Combinational logic circuits “deal” with an instruction
- Memory elements hold state info, i.e., “remember”
Our goal: understand how the processor executes machine instructions and issues related to the speed of this execution

So, we started having a look at the components of a processor

✓ Combinational logic circuits
- Made of many logic gates
- Multi-functional combinational logic circuits: cannot adjust wires while executing so select a function using control signals
  - Consider the ALU, for example, if we wanted to have the ALU compute an addition (ADD instruction), we could adjust its wires (at the hardware level) while the CPU is executing, but that would not be feasible (actually, that would be quite impossible to do). So instead, the ALU is designed such that one of its input is a control signal (S). The value of S (e.g., 0001) is what dictates/selects the function the ALU is to perform.

✓ Combinational logic circuits do not store values
- Cost: propagation delay

✓ Sequential logic circuits
- Made of combinational logic circuits, memory elements (clocked registers) and clock
- Circuit that “remembers” values (state) and perform computations on these values
Next Lecture

- Instruction Set Architecture (ISA)
  - Definition of ISA
  - ISA design
  - ISA evaluation
    - Improving our ISA - Decreasing effect of von Neumann bottleneck
    - 3 Strategies

- Execution of machine instructions
  - Intro to logic design, combinational and sequential logic circuits
  - Sequential execution of machine instructions
  - Pipelined execution of machine instructions