

# CSE372

## Digital Systems Organization and Design Lab

Prof. Milo Martin

Unit 6: P37X Pipelined

CSE 372 (Martin): P37X Pipelined

1

## Non-Pipelined Processor Lab

---

- Any comments or problems?
- Any problems getting it to work on the board?
  - No? Restricted Verilog has been successful!
- Was it harder or easier than you expected?
- Reminder: final demo by Friday
- Next lab:
  - Use what you have learned...

CSE 372 (Martin): P37X Pipelined

3

## Agenda for Today

---

- Discuss Lab 3 and Lab 4
  - Pipelined design
- Discuss video device and standards
  - Slides from UNC
- The Evils of Clock Gating
  - Slides from UC-Berkeley
- Discuss rest of semester

CSE 372 (Martin): P37X Pipelined

2

## Final Lab: Pipelined Processor

---

- Familiar 5-stage pipeline
  - Fetch, Decode, Execute, Memory, Writeback
- Fully bypassed
- One-cycle "load use" penalty
  - A dependent instruction right after the load
- Branch handling
  - Resolved in "execute" stage, two-cycle penalty
  - Initially: just stall after branches
  - Final: use simple branch predictor to efficiently execute branches
- Performance counters
  - Cycle counter
  - Instruction counter, branch stall counter, load stall counters

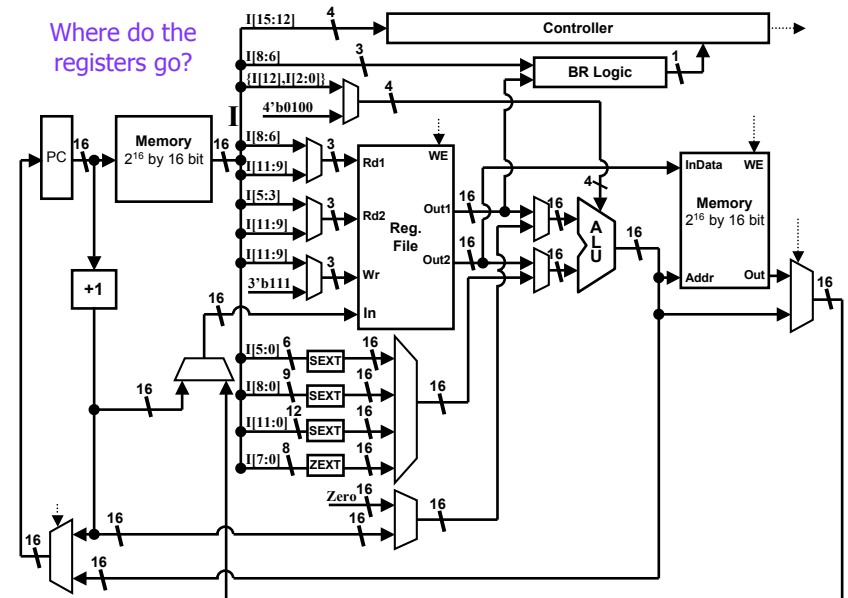
CSE 372 (Martin): P37X Pipelined

4

## First Step: Design Document

- The pipelined datapath design is up to you
  - (as is how to test it)
- Design document
  - Describe the datapath
    - Where are the pipeline registers
    - List or table of which signals are latched in which stages
  - Include a diagram (or diagrams) of the datapath
  - Schematics for any new components (e.g., branch predictor)
  - Include testing strategy
- Note:
  - Memories already have implicit registering of output values
  - No need for "global write enable" pseudo multi-cycle tricks

Where do the registers go?



## Bypassing, Stalling, and Flushing

- Bypassing
  - Which value to use in a given stage?
  - Control logic looks at "recent past"
  - Look at instruction in later stage
- Stalling
  - Dependent instruction after load
  - How?
- Flushing
  - Speculatively execute instructions after branch
  - Using the prediction
  - If wrong, need to cancel two instructions
    - Fetch and Decode
  - Again, how?

## Branch Predictor

- Predict the "next PC" for an instruction
  - Predict during Fetch: PC in, next-PC prediction out
  - Train at Execute: PC in, actual next-PC in, write enable in
- Two memory arrays
  - "Tag" array
  - "Next PC" array
  - If the tag matches, return "Next PC"
    - Else, return PC+1
- Detect mispredict via comparing PC
  - If wrong, train predictor
  - Write PC into tag array, write "actual next PC" into "next PC" array

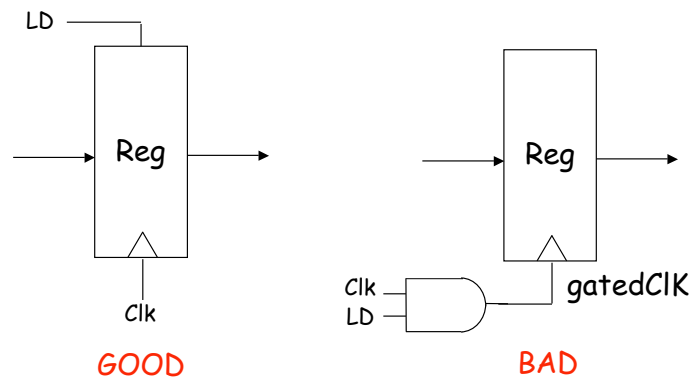
## Honor Points

- Better branch predictor
  - Two-bit saturating counters, lots of others
- High-frequency design
  - Tune design to get fastest clock possible
  - Deeper pipeline - two-cycle "Fetch" and "Memory" stages
  - Add data and/or instruction cache
- Add some instructions
  - Conditional move, hardware divide, 16-bit floating point
- Add new devices (audio in/out, flash memory)
- Write some software
  - Text on video support, compiler for P37X, etc.
- Superscalar (two instructions per cycle)
- Any of your ideas (talk to me first)

## Video Device

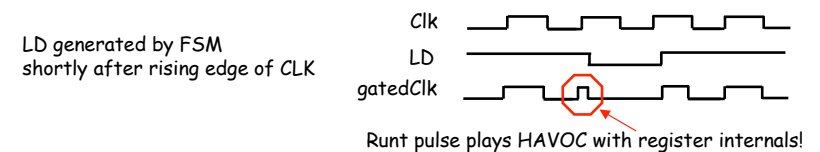
- How does our video device actually work?
- Frame buffer
  - RAM that holds the current image on the screen
  - Hardware walks over frame buffer to generate analog signal
  - In LC-3 and P37X, we put this frame buffer right in memory
    - Most systems today have dedicated frame buffers
- In some classes, they make you build the video circuit, too
  - Switch to slides from UNC

## Why Gating of Clocks is Bad!



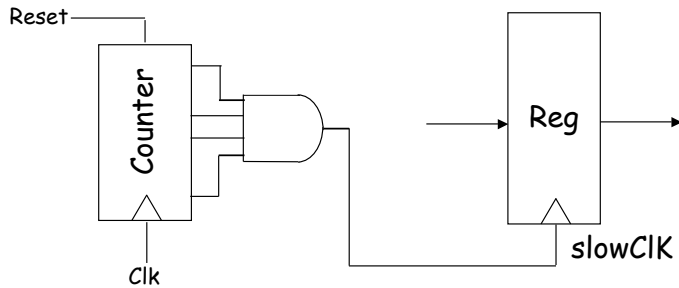
**Do NOT Mess With Clock Signals!**

## Why Gating of Clocks is Bad!



**Do NOT Mess With Clock Signals!**

## Gating of Clocks: Bad



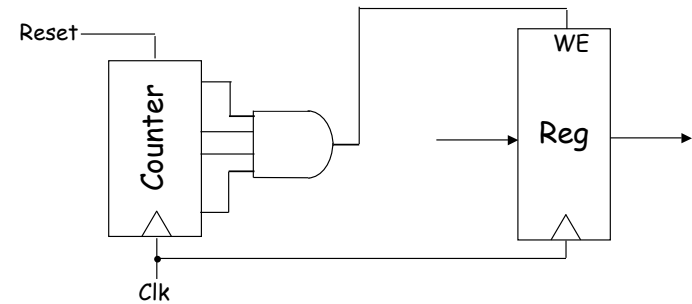
BAD

Do NOT Mess With Clock Signals!

CSE 372 (Martin): P37X Pipelined

From UC Berkeley CS 150 - Fall 2005 13

## Non-Gating of Clocks: Good



Good!

Do NOT Mess With Clock Signals!

CSE 372 (Martin): P37X Pipelined

From UC Berkeley CS 150 - Fall 2005 14

## Remainder of Course

- Only three class periods after today
- Next week: exam
  - Swap CSE371 lecture into Monday slot
  - CSE372 exam on Tuesday or Thursday
- Following week: Saul Gorn Memorial Lecture
- Last week: wrapup, course evaluations, etc.
  - Any other topics you want to hear about

CSE 372 (Martin): P37X Pipelined

15

## CSE372 Exam

- Written exam - 20% of course grade
  - Nothing actually on the computer
- One page (two sides) page of notes
  - Anything you want
- Topics
  - Basic Verilog
    - Schematic to Verilog
    - Verilog to Schematic
  - FPGAs and design flows
  - The "design process"
  - Synthesis
  - Questions on labs

CSE 372 (Martin): P37X Pipelined

16

## Saul Gorn Memorial Lecture

---

- Prof. Yale Patt
  - Author of CSE240 textbook
  - Prolific computer architecture researcher
  - "Interesting" guy
- Monday April 10th, 3pm
- Possible CSE371 exam question on the lecture

