

CSE 141L

Computer Architecture Lab

Summer Session I, 2005

Lecture 3
Pramod V. Argade
July 12th, 2005

CSE141: Computer Architecture Lab

Instructor: Pramod V. Argade (p2argade@cs.ucsd.edu)
Office Hour:
Tue. 7:30 - 8:50 PM (Center 105)
Wed. 5:00 - 6:00 PM (HSS 1330)
And, by appointment.

TAs:
Anjum Gupta: a3gupta@cs.ucsd.edu
Jianhua Liu: jhliu@cs.ucsd.edu

Readers:
Anthony Choi: buchoi@ucsd.edu
Than-Khar Chin: thchin@ucsd.edu

Textbook: LogicWorks 5, Interactive Circuit Design Software
Capilano Computing Systems, 2004
(Available in UCSD Bookstore.)

Web-page: <http://www.cse.ucsd.edu/classes/su05/cse141L>

CSE141L Course Schedule



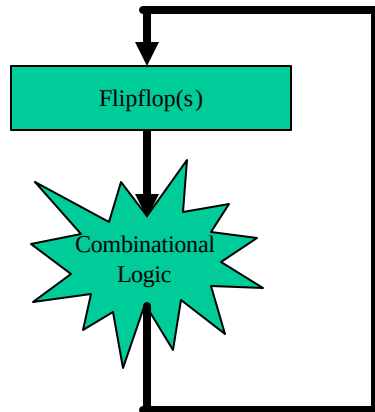
Lecture #	Date	Time	Room	Topic	Assignment Due
1	Tue. 6/28	6 - 7:15 PM	Center 105	Assignment 1: ISA	-
2	Tue. 7/5	6 - 7:15 PM	Center 105	Assignment 2: Assembler & ISS	#1
3	Tue. 7/12	6 - 7:15 PM	Center 105	Assignment 3: Datapath	#2
4	Tue. 7/19	6 - 7:15 PM	Center 105	Assignment 4: CPU	#3
-	Tue. 7/26	No Lecture			
July 26 - 28: Students demonstrate their working CPU Design to TAs by appointment. Assignment #4 reports are due July 26th at 6 PM in the Lab (AP&M 2444?).					

Microprocessor Design Steps



- ✓ Design Instruction Set Architecture (ISA)
- ✓ Code applications
- ✓ Develop software generation tools
- ✓ Develop instruction set simulator (ISS)
 - Design datapath, verify it
 - Design the Processor, simulate logic
 - Verify the processor
 - Fabricate the chip

Processor: A State Machine!



What is a data path?



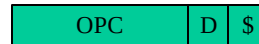
- Path along which data in a processor passes
- Data path consists of
 - Internal storage
 - General purpose register file
 - Special registers
 - ALU
 - Some control logic
 - e.g. signals to control ALU operation

Example Instruction Formats



Example 1: ADD Instruction

ADD $R_D, \$Constant$



$$R_D = R_D + \$Constant$$

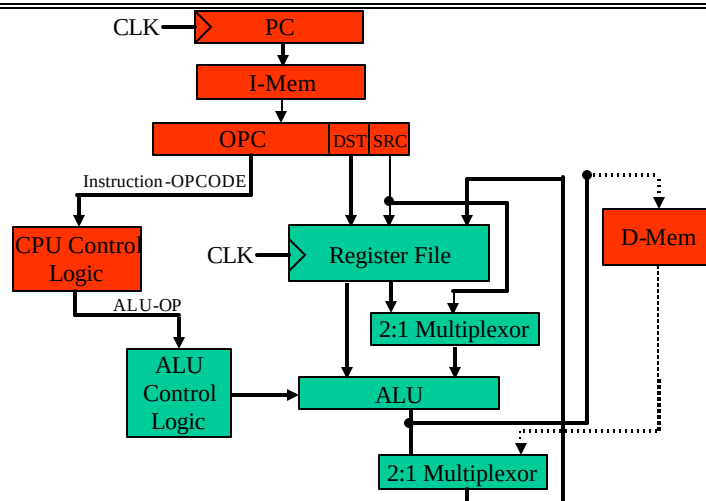
Example 2: MOV Instruction

MOV R_D, R_S



$$R_D = R_S$$

Processor Organization



Note: Blocks in your architecture equivalent to those in green are to be implemented in Lab3

Lab 3 Assignment



- Design data path for your 8-bit processor architecture
- Use LogicWorks 5 to design the data path
- Simulate the data path

What you must include



- All internal storage in your architecture.
 - General purpose register file
 - Stack
 - Accumulator
 - Special registers, status flag(s), etc.
- Arithmetic Logical Unit
 - Implement data path for all instructions that manipulate data in some way
 - MOV, both immediate-to-register and register-to-register
 - ADD, SUB, XOR, SHIFT, ...
- Interconnections between modules
- Logic to control components of ALU

What you don't have to implement

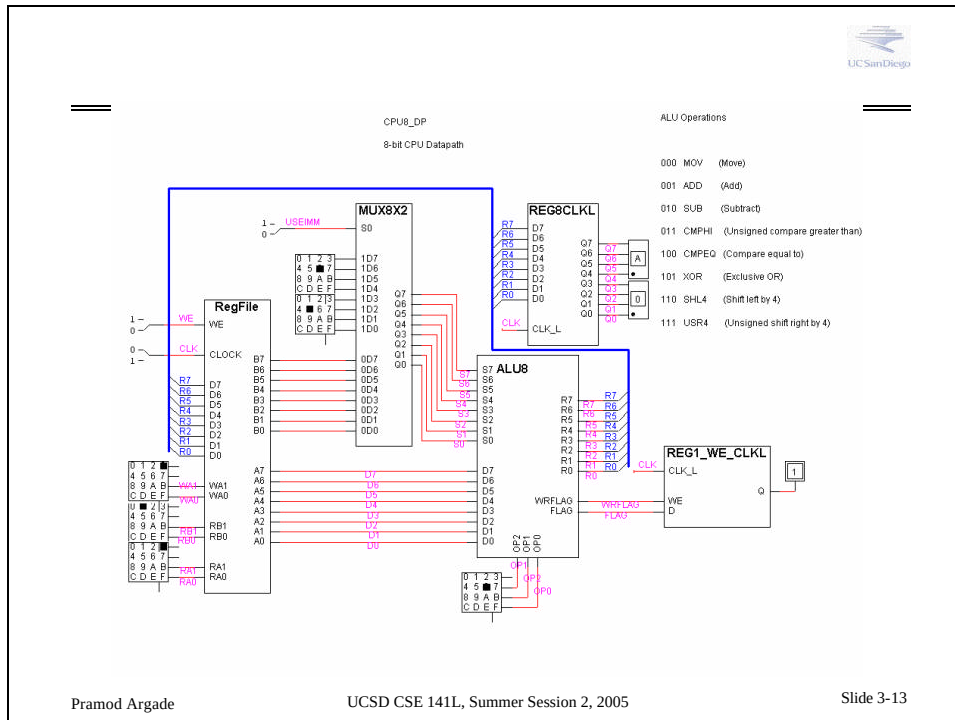


- Following instructions
 - Control transfer instructions (branches, jumps, etc.)
 - Memory load/store instructions
- Program Counter (PC)
- I-Mem
- D-Mem

How you should test data path



- Stimulus generation:
 - Binary switches
 - Hex keypads
- Mechanisms to observe results:
 - Binary displays
 - Hex displays
- Internal variables on the timing diagrams



Pramod Argade

UCSD CSE 141L, Summer Session 2, 2005

Slide 3-13

What you will turn in for this Lab

- Summary of your ISA from Lab 1.
- Printed schematics for the top level data path you designed in LogicWorks 5. No need to submit all the lower level schematics
- Printout of waveforms that show example input data, ALU opcode and the result from the ALU for each instruction.
- All the LogicWorks 5 files you created (to be submitted using turnin script).
- Description of the procedure to follow in order to test the operation of the data path for each instruction for your processor. This is so your TA can test your design.
- Answers to following questions.

Pramod Argade

UCSD CSE 141L, Summer Session 2, 2005

Slide 3-14

Questions



- Which instruction is the most expensive in terms of number of gates it requires (no need to give the exact gate count, just give the reasoning).
- What tricks did you use to decrease the logic in your data path by sharing the logic among more than one instruction?
- How does your "move constant to register" and "move register to register" instruction work? Is it a special case of another ALU instruction, or does it use special data path elements?
- Using your data path, explain how you will load a register with a value from a memory location and store contents of a register to a memory location.

You should show on waveforms and Demo to TA



- Ability to load a constant with the required number of bits for your ISA into any appropriate register (e.g. any general purpose register).
- Correct operation of the ALU for all the ALU opcodes supported.
- Ability to have the same register to be the source and destination of an instruction.
- Show the ALU operation for interesting input data. For example, if the carry from the adder is saved in your architecture, use data that both does and does not set the carry bit.

Working knowledge of LogicWorks 5 Assumed!



- Various components available in Standard Libraries, such as, D Flip-flop, logic gates, multiplexors, registers, adders, clock, binary switch, binary display, hex keypad, hex display, etc.
- How to connect a bus to various components.
- How to define a sub-circuit bottom up, i.e. create a circuit and then use it to define the pins on the parent symbol.
- How to simulate a circuit and generate waveforms.
- How to print a circuit and waveforms.

Useful Hints



- Build hierarchical design
- Test thoroughly at every level of hierarchy
 - Connect binary switches and hex keypads to provide inputs
 - Connect binary and hex displays to observe behavior
 - Test combinations of control signals
 - Test all corner cases for data
- Bugs buried deep inside hierarchy are hard to find!

Lab Due Dates



- Lab 3 Due:
 - ⇒ Before 6:00 PM Tuesday, July 19th
- No late submissions!
- Make an appointment with TA before July 19th to demo your Lab 3 design
- You are not allowed to make changes to your design from the time you “turnin” to the time you demo your design to the TA