CS61C : Machine Structures

Lecture #20
Introduction to Synchronous Digital Systems

2008-3-12

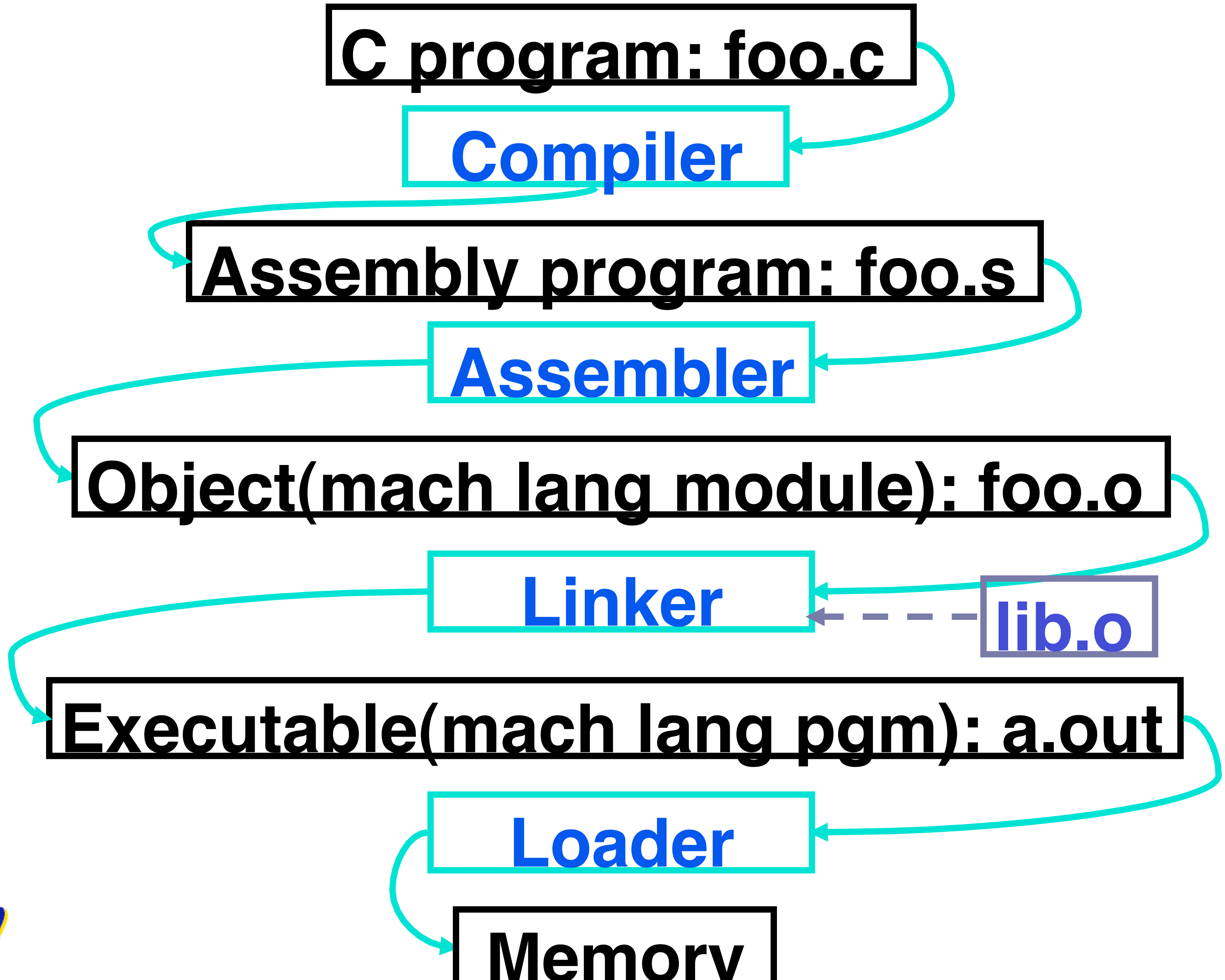
Scott Beamer, Guest Lecturer

Wifi in Air Coast to Coast

www.aircell.com

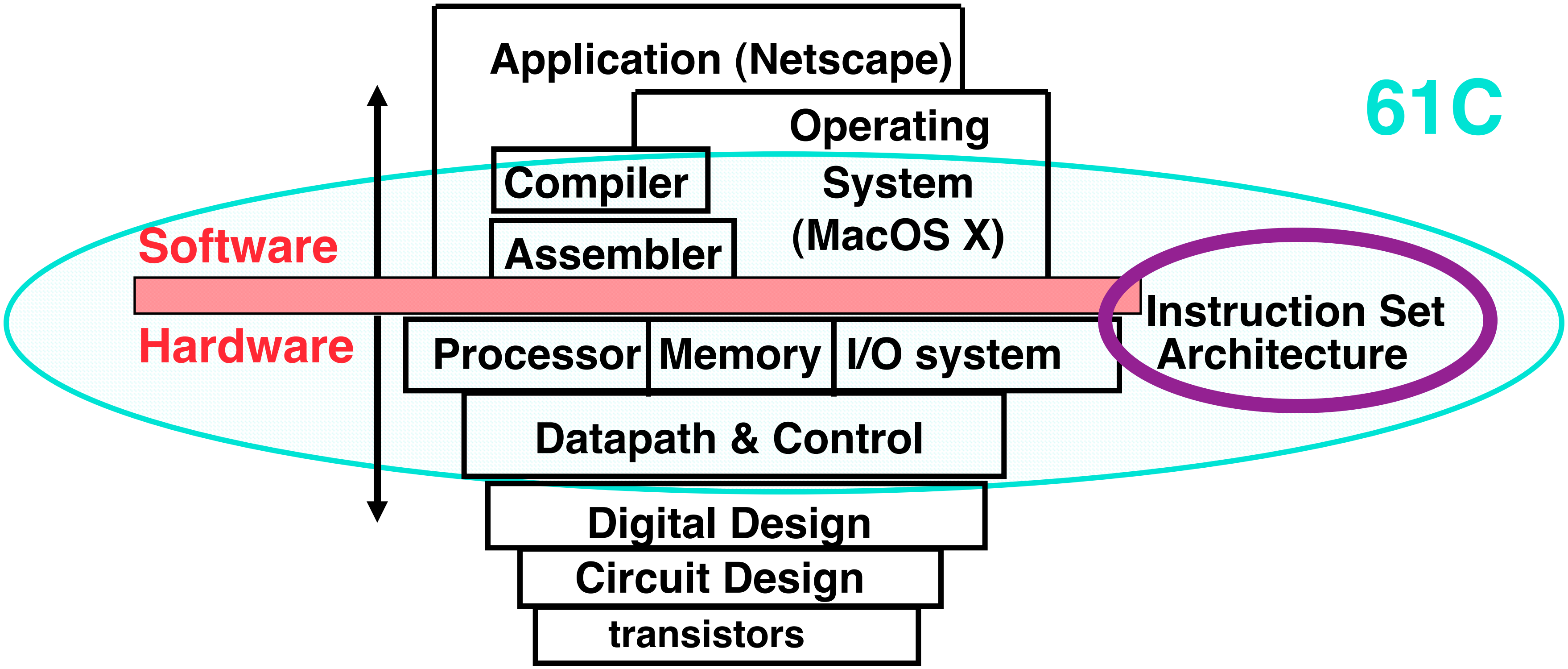
Beamer, Spring 2008 © UCB

Review



Beamer, Spring 2008 © UCB

What are "Machine Structures"?



Coordination of many *levels of abstraction*

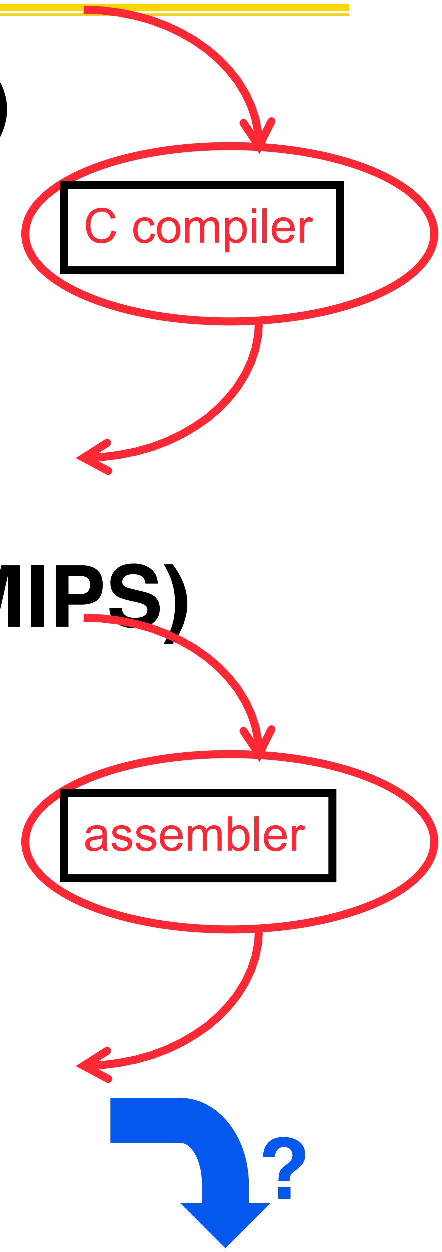
ISA is an important abstraction level: contract between HW & SW

Beamer, Spring 2008 © UCB

Below the Program

- **High-level language program (in C)**

```
swap  int v[], int k){
      int temp;
      temp = v[k];
      v[k] = v[k+1];
      v[k+1] = temp;
}
```



- **Assembly language program (for MIPS)**

```
swap: sll   $2, $5, 2
      add   $2, $4,$2
      lw    $15, 0($2)
      lw    $16, 4($2)
      sw    $16, 0($2)
      sw    $15, 4($2)
      jr    $31
```

- **Machine (object) code (for MIPS)**

```
000000 00000 00101 0001000010000000
000000 00100 00010 0001000000100000 . . .
```


Beamer, Spring 2008 © UCB

Synchronous Digital Systems

The hardware of a processor, such as the MIPS, is an example of a Synchronous Digital System

Synchronous:

- **Means all operations are coordinated by a central clock.**
  - **It keeps the "heartbeat" of the system!**

Digital:

- **Mean all values are represented by discrete values**
- **Electrical signals are treated as 1's and 0's and grouped together to form words.**

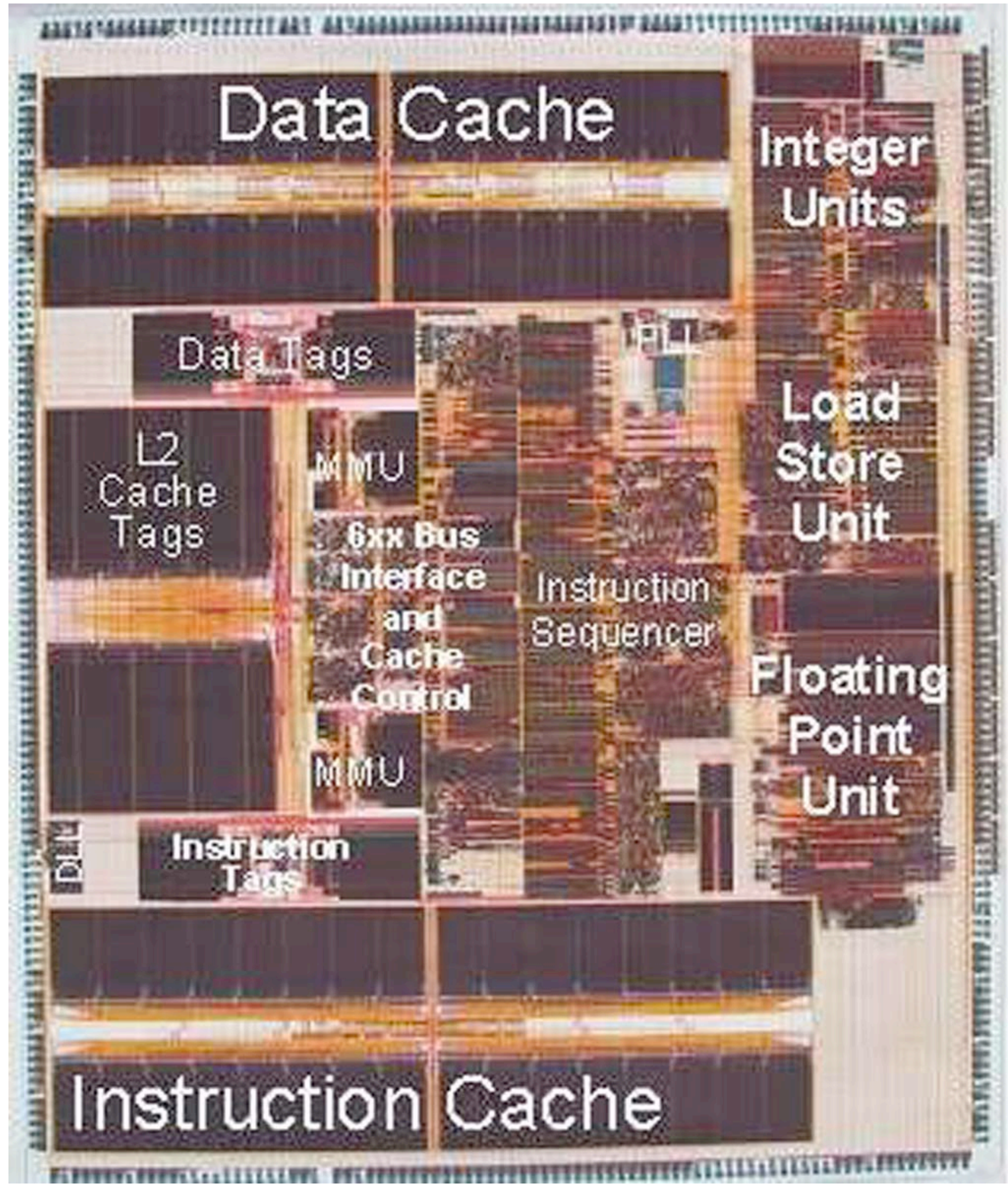
Beamer, Spring 2008 © UCB

Logic Design

- **Next 4 weeks: we'll study how a modern processor is built; starting with basic elements as building blocks.**
- **Why study hardware design?**
  - **Understand capabilities and limitations of hardware in general and processors in particular.**
  - **What processors can do fast and what they can't do fast (avoid slow things if you want your code to run fast!)**
  - **Background for more detailed hardware courses (CS 150, CS 152)**
  - **There is just so much you can do with processors. At some point you may need to design your own custom hardware.**

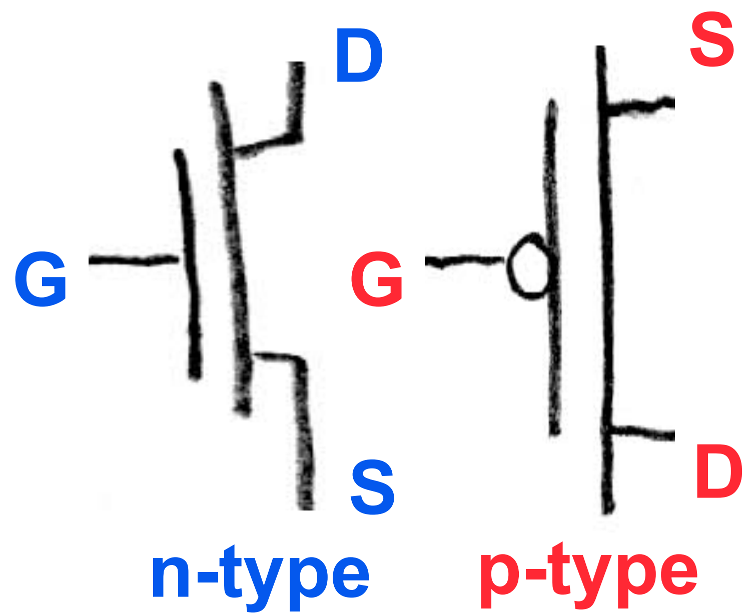
PowerPC Die Photograph

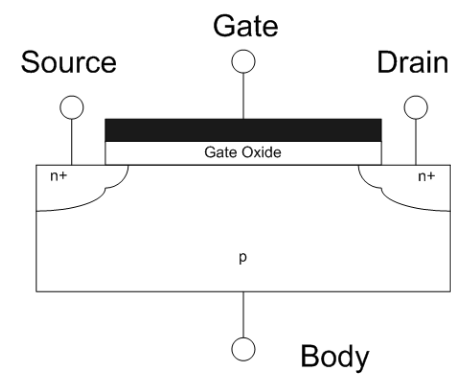
Let's look closer…



Transistors 101

- **MOSFET**
  - Metal-Oxide-Semiconductor Field-Effect Transistor
  - Come in two types:
    - n-type NMOSFET
    - p-type PMOSFET
- **For n-type (p-type opposite)**
  - If voltage not enough between G & S, transistor turns "off" (cut-off) and Drain-Source NOT connected
  - If the G & S voltage is high enough, transistor turns "on" (saturation) and Drain-Source ARE connected





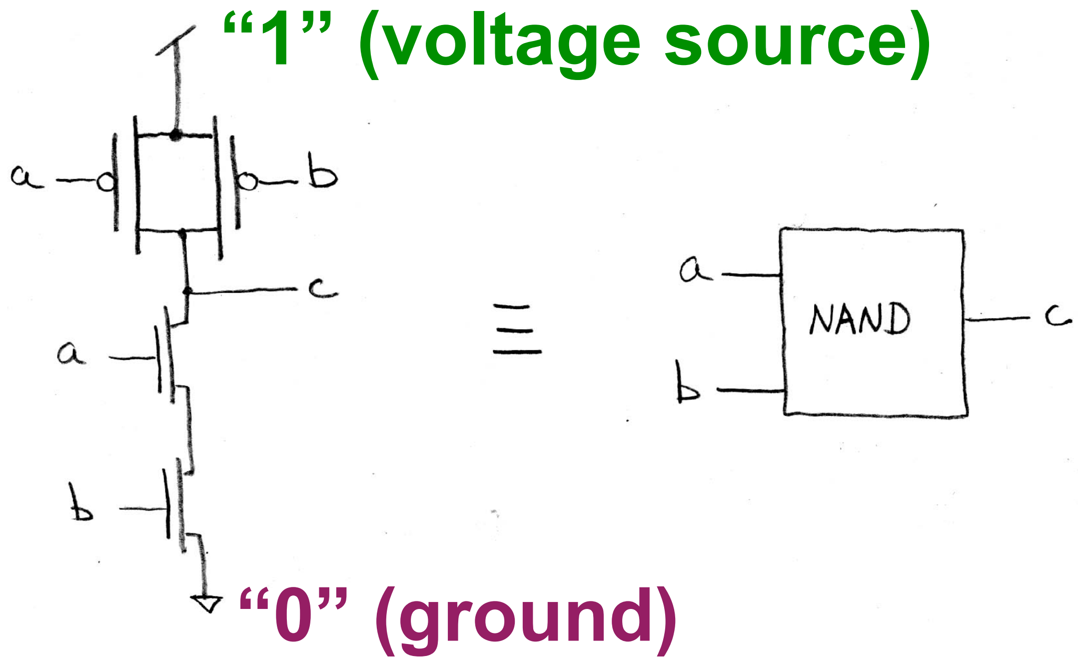
Side view

www.wikipedia.org/wiki/Mosfet

Beamer, Spring 2008 © UCB

Transistor Circuit Rep. vs. Block diagram

- Chips is composed of nothing but transistors and wires.
- Small groups of transistors form useful building blocks.

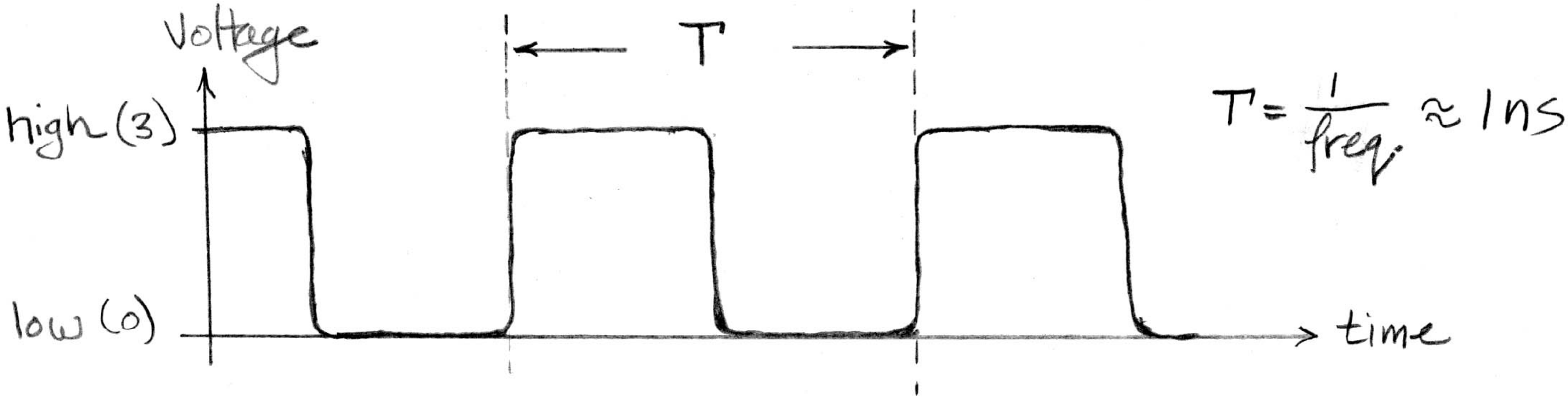


a	b	c
0	0	1
0	1	1
1	0	1
1	1	0

- Block are organized in a hierarchy to build higher-level blocks: ex: adders.

Beamer, Spring 2008 © UCB

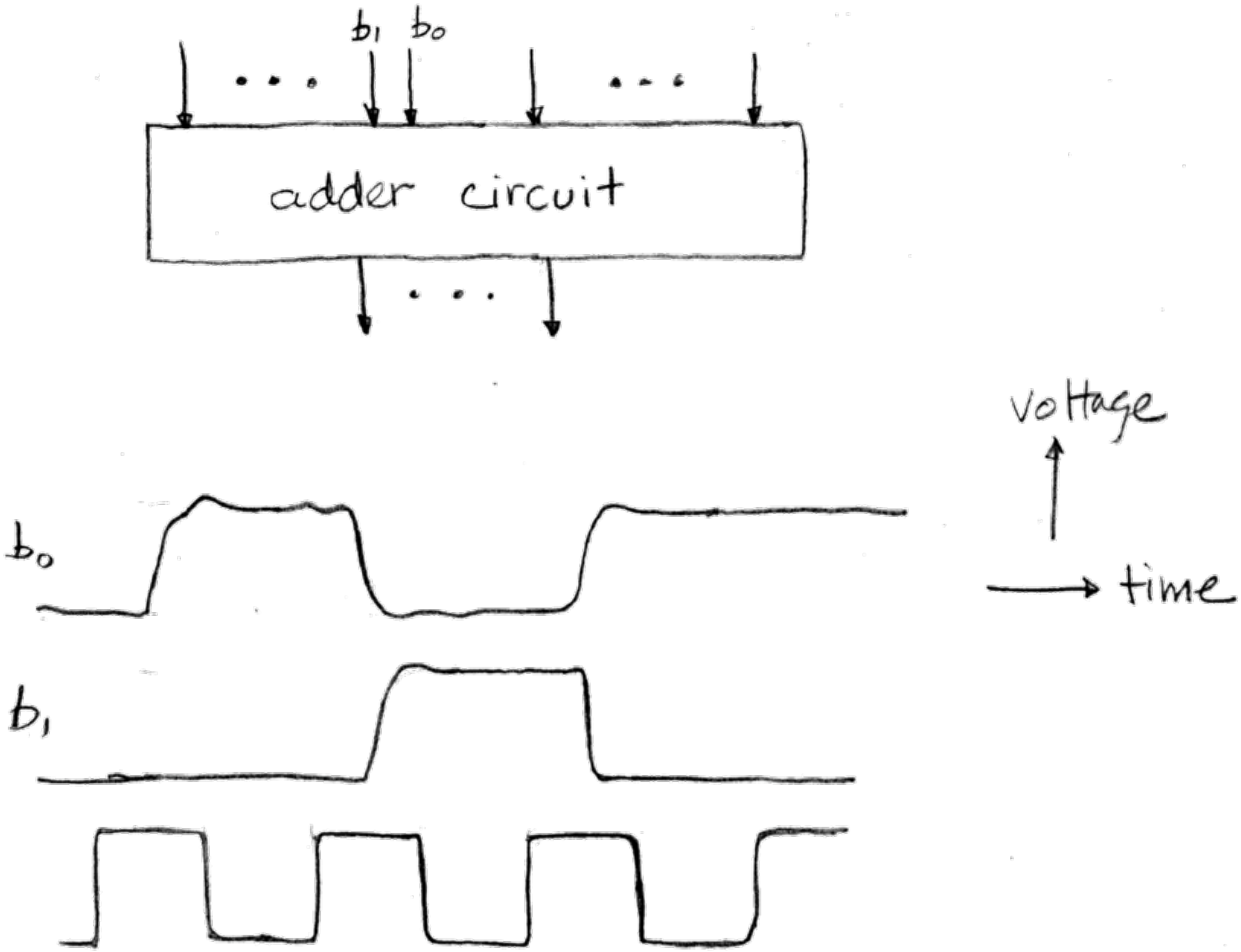
Signals and Waveforms: Clocks



- **Signals**
  - When digital is only treated as 1 or 0
  - Is transmitted over wires continuously
  - Transmission is effectively instant
    - Implies that any wire only contains 1 value at a time

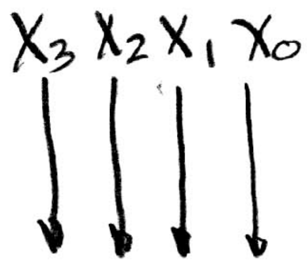
Beamer, Spring 2008 © UCB

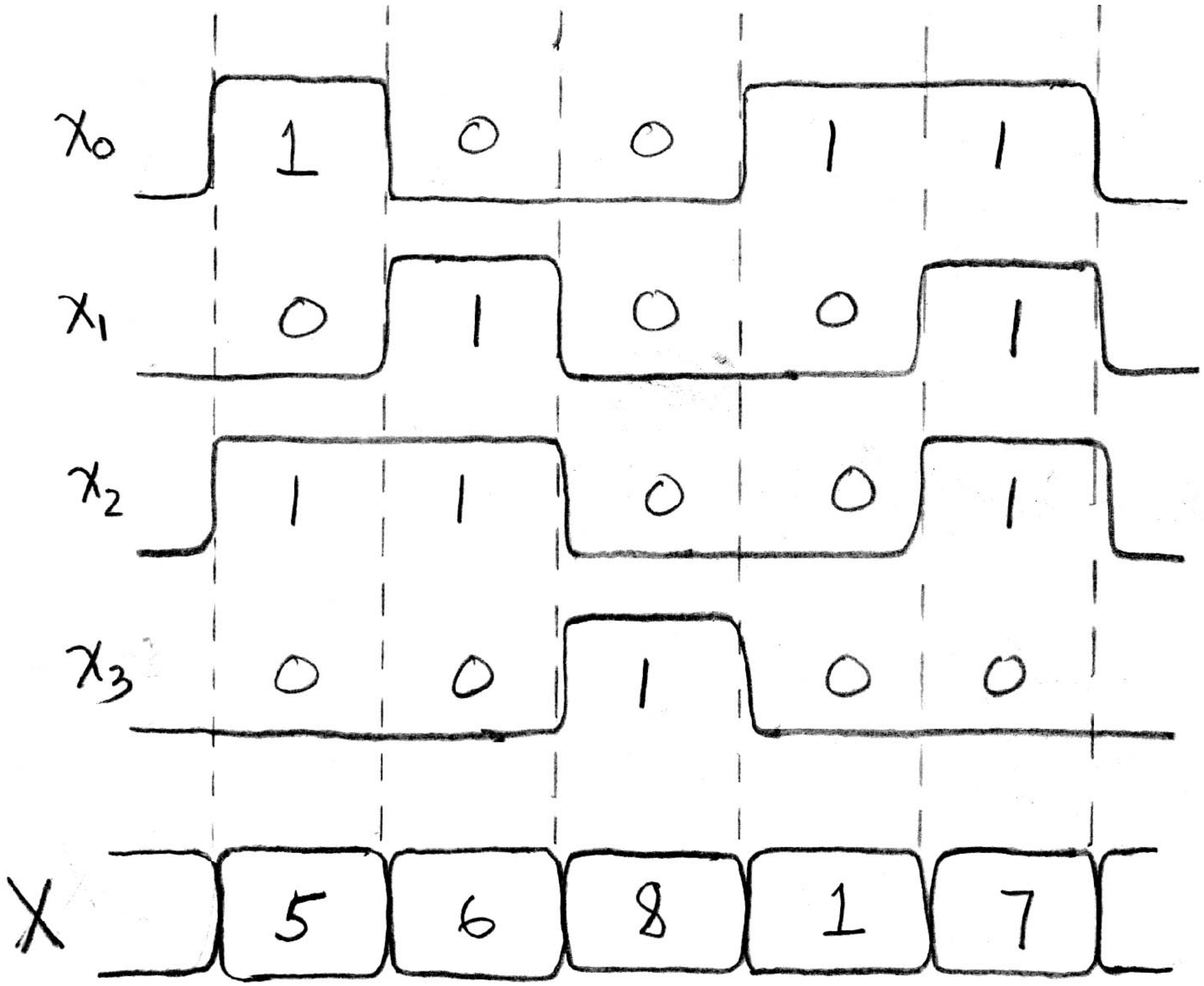
Signals and Waveforms



Beamer, Spring 2008 © UCB

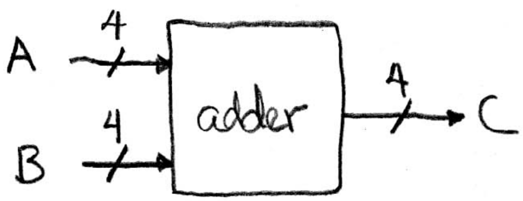
Signals and Waveforms: Grouping



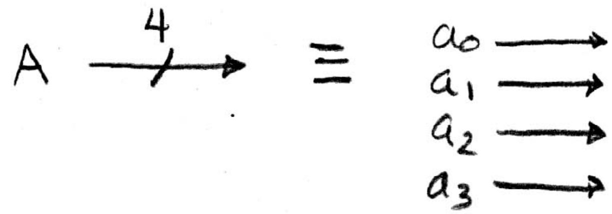


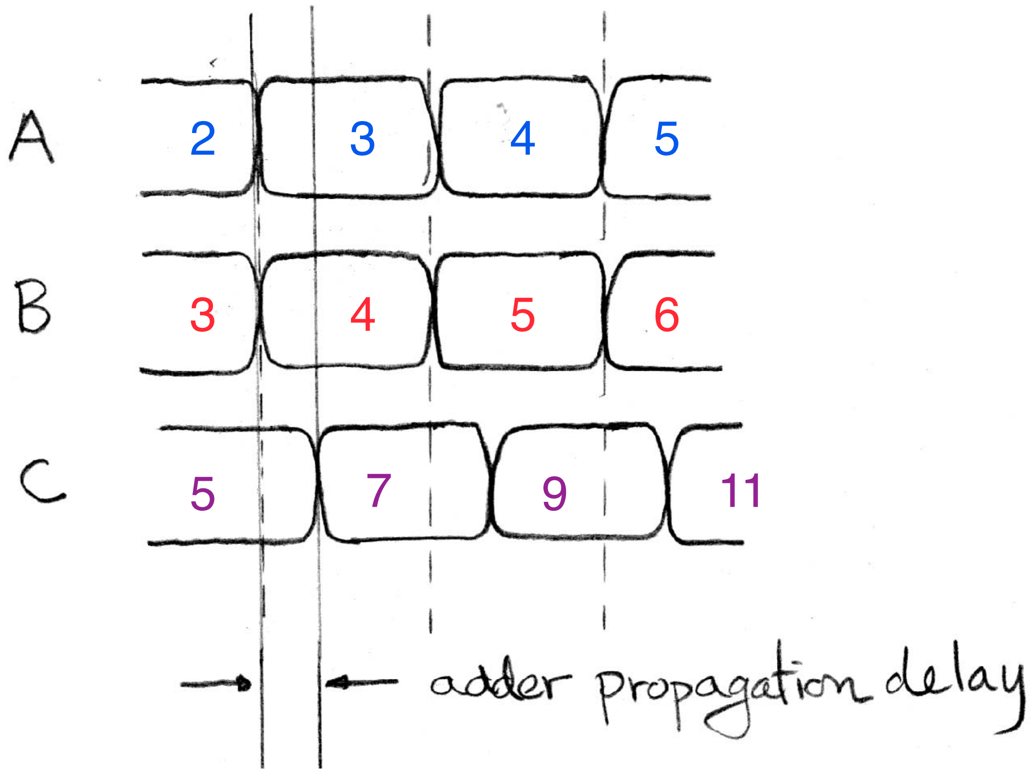
Beamer, Spring 2008 © UCB

Signals and Waveforms: Circuit Delay



$$A = [a_3, a_2, a_1, a_0]$$
$$B = [b_3, b_2, b_1, b_0]$$





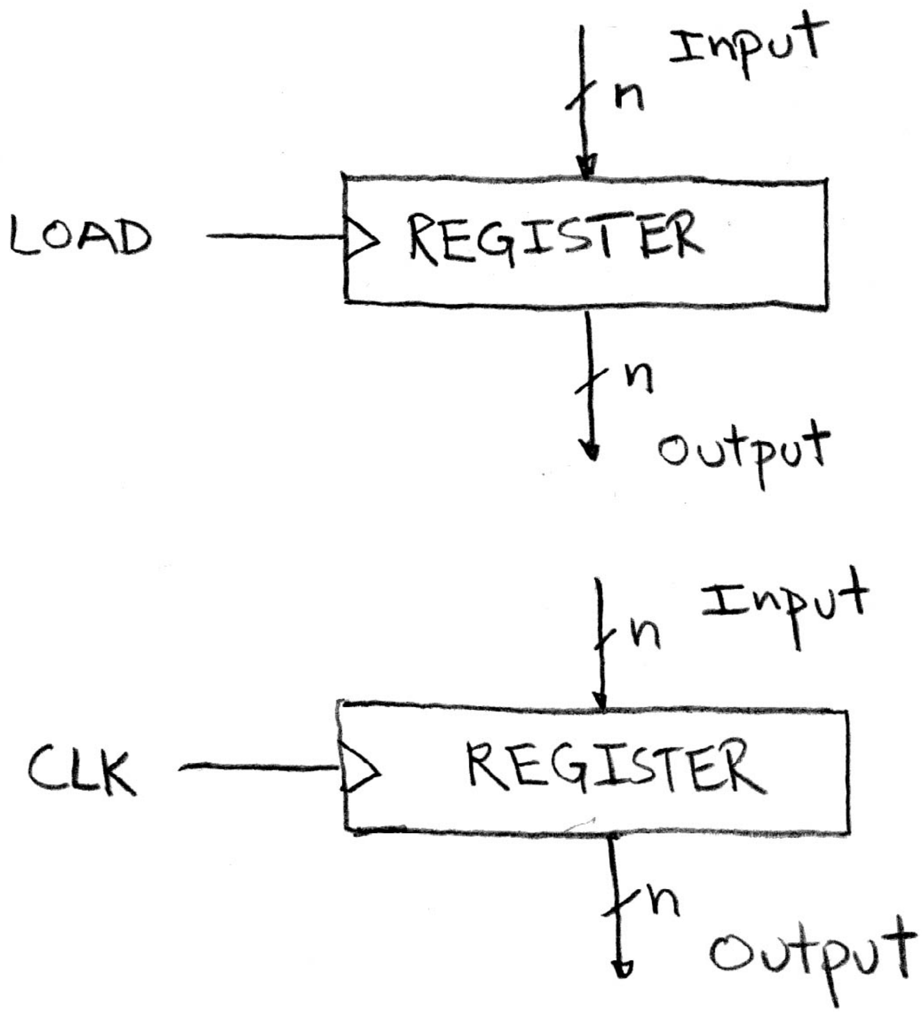
Beamer, Spring 2008 © UCB

Type of Circuits

- Synchronous Digital Systems are made up of two basic types of circuits:
- Combinational Logic (CL) circuits
  - Our previous adder circuit is an example.
  - Output is a function of the inputs only.
  - Similar to a pure function in mathematics, y = f(x). (No way to store information from one invocation to the next. No side effects)
- State Elements: circuits that store information.

Beamer, Spring 2008 © UCB

Circuits with STATE (e.g., register)



Beamer, Spring 2008 © UCB

Peer Instruction

A. SW can peek at HW (past ISA abstraction boundary) for optimizations

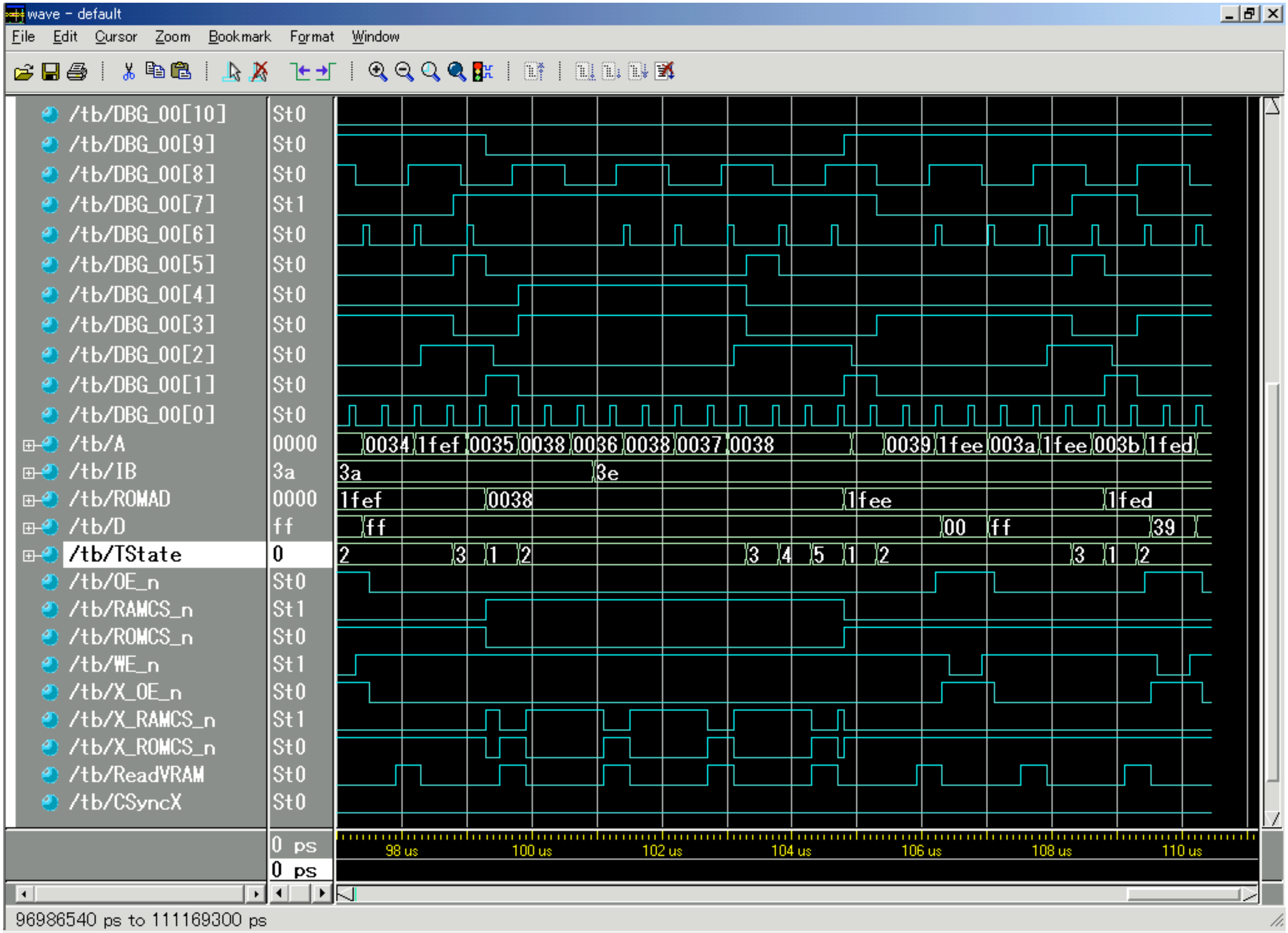
B. SW can depend on particular HW implementation of ISA

C. Timing diagrams serve as a critical debugging tool in the EE toolkit

	ABC
0:	FFF
1:	FFT
2:	FTF
3:	FTT
4:	TFF
5:	TFT
6:	TTF
7:	TTT

Beamer, Spring 2008 © UCB

Sample Debugging Waveform



And in conclusion…

- ISA is very important abstraction layer
  - Contract between HW and SW
- Clocks control pulse of our circuits
- Voltages are analog, quantized to 0/1
- Circuit delays are fact of life
- Two types of circuits:
  - Stateless Combinational Logic (&,|,~)
  - State circuits (e.g., registers)