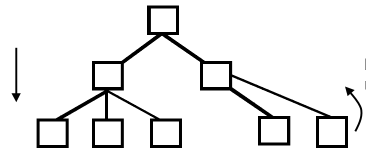
0 Admin
1 Recap
2 Synch design rules
3 Computer data path
4 Assembly language/big picture
5 Katz' 16-bit data path
6 Register Transfer Language

CS150 Fall 1999 Lecture 16 Copyright © 1999 The Regents of the University of California R. Fearing

1 RECAP

Top Down Design/Bottom Up Design



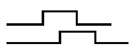
build test/debug in modules

Note discrete time, discrete amplitude

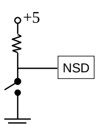
CS150 Fall 1999 Lecture 16 Copyright © 1999 The Regents of the University of California R. Fearing

2 SYNCHRONOUS DESIGN RULES

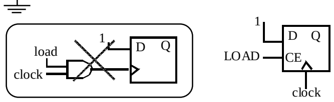
#1 One clock only, clock and $\overline{\text{clock}}$ ok.



#2 All inputs synched. No



#3 Don't gate the clock!



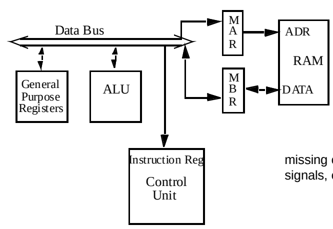
#4 Avoid clock skew – use BUFGS

#5 One asynchronous control okay: Global Set/Reset, Startup.GTS

CS150 Fall 1999 Lecture 16 Copyright © 1999 The Regents of the University of California R. Fearing

3 COMPUTER DATA PATH

Rough block design – single program/data memory



missing control signals, etc.

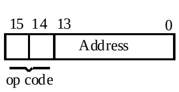
CS150 Fall 1999 Lecture 16 Copyright © 1999 The Regents of the University of California R. Fearing

4 ASSEMBLY LANGUAGE

SW: C source $\xrightarrow{\text{compiler}}$ assembly code $\xrightarrow{\text{assembler}}$ executable machine code

HW: machine code $\xrightarrow{\text{FSM/control unit}}$ control signals

4.1 Katz 16-bit computer figure 11.13
– one general purpose register "accumulator"
– 4 instructions: Load, Store, Add, Branch



How much memory?

$2^{14} \times 2 \text{ bytes} = 32\text{K RAM}$

CS150 Fall 1999 Lecture 16 Copyright © 1999 The Regents of the University of California R. Fearing

4 ASSEMBLY LANGUAGE (cont.)

Op Code		Operation			
0	0	Load	XXX	:	memory [XXX] $\rightarrow$ AC
0	1	Store	XXX	:	AC $\rightarrow$ memory [XXX]
1	0	Add	XXX	:	AC + mem [XXX] $\rightarrow$ AC
1	1	Brn	XXX	:	if A < 0, then go to XXX (accumulator bit 15 = 1 $\Rightarrow$ A < 0) two's complement

4.2 C Code

```
main ( )
{int i=N; int x=0;
while (i--) {x=x+i;}
```

$$x = \sum_{i=0}^{N-1} i$$

4 ASSEMBLY LANGUAGE (cont.)

4.3 Assembler Code

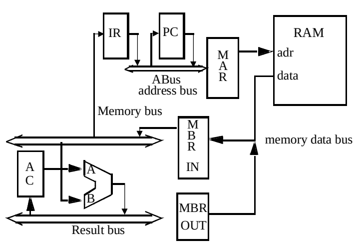
Address	Label		Operation	Comment	
10	_i	:	Data N	i=N;	data segment
11	_x	:	Data 0	x=0;	
12	_minus1	:	–1=0xffff	constant–1	
13	_main	:	load _minus1	$-1 \rightarrow$ AC	code or text segment
14	_main+1	:	add _i	$i-1 \rightarrow$ AC	
15			store _i		
16			BRN _done		
17			add _x	$x+i \rightarrow$ AC	x=x+i
18			store _x	AC $\rightarrow$ x	
19			load _minus1	branch always	
1A			BRN _main+1		
1B	_done				

4 ASSEMBLY LANGUAGE (cont.)

4.4 Machine Code/Memory Contents (16-bit-wide RAM)

10	N	i
11	0	N
12	$0 \times$ ffff	-1
13	012	load _minus1
14	0×8010	add i
15	0×4010	store i
16	$0 \times$ C01A	BRN _done
17	0×8011	add _x
18	0×4011	store _x
19	0×0012	load _minus1
1A	$0 \times$ C014	BRN 0×14

5 SIMPLE CPU DATA PATH (Katz 11.13)



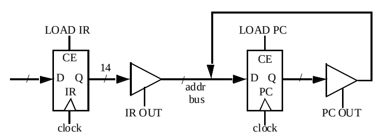
6 REGISTER TRANSFER LANGUAGE/MICRO-OPERATIONS

Register transfer notation

For example, BRN instruction (if branch taken)

IR $\rightarrow$ PC (usually one clock cycle, synchronous design)

Detail of Address Bus



1=enable, 0=tristate

5 REGISTER TRANSFER LANGUAGE (cont.)

RTL	Micro-operations		
IR→PC	IR OUT=1 LOAD PC=1	or	IR→ABus ABus→IR

Timing Diagram IR→PC