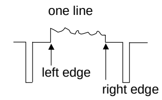
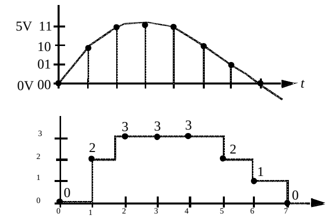
CS150 Fall 1999 Lecture 15 Copyright © 1999 The Regents of the University of California R. Fearing

1 RECAP

- Video signal



- A/D converter



Note discrete time, discrete amplitude

CS150 Fall 1999 Lecture 15 Copyright © 1999 The Regents of the University of California R. Fearing

2 TOP DOWN DESIGN/BOTTOM UP IMPLEMENTATION

Steps 2.1

1. Understand problem specifications (goal) for (sub)system
2. Identify inputs/outputs
3. Determine/decide on algorithm (e.g., parallel vs. sequential, combinatorial vs. sequential)
4. Draw rough block diagram of data path, showing storage elements, interconnect, processing elements
   → If too hairy, choose different algorithm, go to #3
5. Detailed block diagram all I/Os
   Identify required control signals for data path elements
   → If subsystem is more complicated than library block, go to #1
6. Specify order and duration of control signals: i.e., timing diagram for each module/subsystem
   Specify timing interface between modules
7. Design data path and simulate modules
   → Go to #6
8. Design control FSM and simulate each module
   → Go to #6
9. Put whole thing together and go to #3 or #6 – may need to iterate to get all pieces working together!

CS150 Fall 1999 Lecture 15 Copyright © 1999 The Regents of the University of California R. Fearing

2 TOP DOWN DESIGN/BOTTOM UP IMPLEMENTATION(cont.)

2.2 Constraints in Design

1. Space/time tradeoff: parallel → fast and large
   serial → slow and small
   (e.g.) 32-bit adder
2. Design/develop time: inefficient design, but easy to understand and debug
   (e.g., 80/20 rule: 80% done in 20% of time. Multiply time to completion by π).
3. Robustness: customer will find kludges if you don't!
   [e.g., (i) Sun keyboards typing limitation and (ii) computer board bus controller not self resetting]
4. Testable

CS150 Fall 1999 Lecture 15 Copyright © 1999 The Regents of the University of California R. Fearing

3 TOP DOWN DESIGN EXAMPLE

Design a system that updates 16 object positions. Each quantity 16 bits.

$$X_i^{(n+1)} = X_i^{(n)} + \delta X_i^{(n)} \qquad i = 0 \ldots 15$$

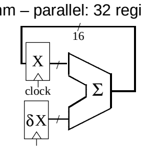
32 storage locations

$X_i, \delta X_i$

#1 ✓

#2 informal, no I/O

#3 algorithm – parallel: 32 registers, 16 adders



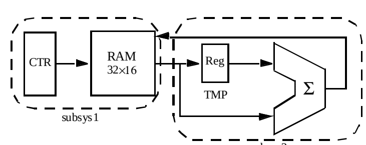
time: 1 clock cycle
too much space

serial: time share adder, use RAM instead of registers

CS150 Fall 1999 Lecture 15 Copyright © 1999 The Regents of the University of California R. Fearing

3 TOP DOWN DESIGN EXAMPLE (cont.)

#4 Rough block design

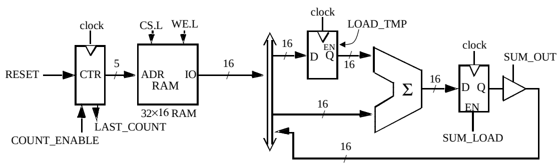


3 TOP DOWN DESIGN EXAMPLE (cont.)

#5 Control signals and details (e.g., RAM only 1 I/O bus, need tri-state on adder output, need register on SUM OUT

$$\left.\begin{array}{l}\xrightarrow{\text{in}} \text{COUNT\_ENABLE} \\ \xleftarrow{\text{out}} \text{LAST\_COUNT}\end{array}\right\}\text{counter} \qquad \left.\begin{array}{l}\xrightarrow{\text{in}} \text{CS} \\ \xrightarrow{\text{in}} \text{WE}\end{array}\right\}\text{RAM}$$

$$\left.\begin{array}{l}\text{LOAD\_TMP} \\ \text{SUM\_LOAD} \\ \text{SUM\_OUT}\end{array}\right\}\text{Adder}$$



check all library components? ✓

3 TOP DOWN DESIGN EXAMPLE (cont.)

#6 Timing signals, (RTL)

$$\left.\begin{array}{l} 0 \rightarrow i \\ \text{While}(\, i \neq 32) \\ \{ \quad \delta_i \rightarrow \text{TMP} \\ \quad i+1 \rightarrow i \\ \quad X_i + \text{TMP} \rightarrow \text{sum} \\ \quad \text{sum} \rightarrow X_i \\ \quad i+1 \rightarrow i \quad \} \end{array}\right\} \text{algorithm}$$

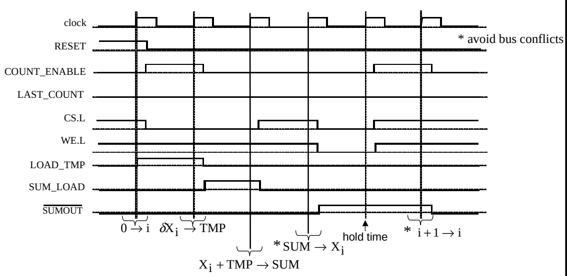
0	δX_0
1	X_0
2	δX_1
3	X_1
	⋮
31	X_{15}

. . . .

3 TOP DOWN DESIGN EXAMPLE (cont.)

6 Timing signals (cont.)

assume clock slow ⇒ functional timing diagram



4 SYNCHRONOUS DESIGN RULES

1 One clock only
2 All inputs synchronized
3 One global asynch reset
4 Don't gate the clock; use clock enables instead
5 Avoid clock skew; use BUFGS