ECE 274 Digital Logic – Fall 2008

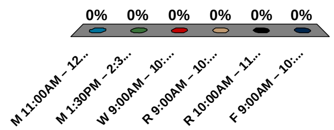
MWF 12-12:50PM, ILC 140

Roman Lysecky, rlysecky@ece.arizona.edu
http://www.ece.arizona.edu/~ece274

The University of Arizona
Tucson Arizona

Which office hours work best for you?

1. M 11:00AM – 12:00PM
2. M 1:30PM – 2:30PM
3. W 9:00AM – 10:00AM
4. R 9:00AM – 10:00AM
5. R 10:00AM – 11:00AM
6. F 9:00AM – 10:00AM



Digital Design

Chapter 1:
Introduction

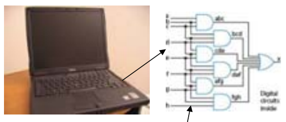
Slides to accompany the textbook *Digital Design*, First Edition,
by Frank Vahid, John Wiley and Sons Publishers, 2007.
http://www.ddvahid.com

Digital Logic – Introduction

1.1

Why Study Digital Design?

- Look "under the hood" of computers
  - Solid understanding --> confidence, insight, even better programmer when aware of hardware resource issues
- Electronic devices becoming digital
  - Enabled by shrinking and more capable chips
  - Enables:
    - Better devices: Better sound recorders, cameras, cars, cell phones, medical devices,...
    - New devices: Video games, PDAs, ...
  - Known as "embedded systems"
    - Thousands of new devices every year
    - Designers needed: Potential career direction



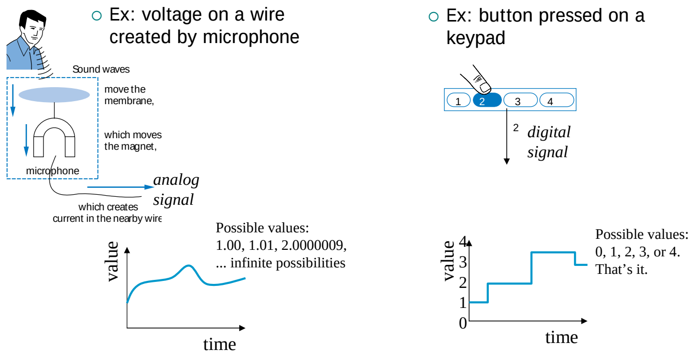
1995	1997	1999	2001	2003	2005	2007	
Portable music players	Satellites	Cell phones	DVD players	Cameras	Video recorders	TVs	Musical instruments ???

- Years shown above indicate when digital version began to *dominate*
  - (*Not* the first year that a digital version appeared)

Digital Logic – Introduction

What Does "Digital" Mean?

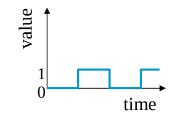
- Analog signal
  - Infinite possible values
    - Ex: voltage on a wire created by microphone
- Digital signal
  - Finite possible values
    - Ex: button pressed on a keypad



Digital Logic – Introduction

Binary - Digital Signals with Only Two Values

- ***Binary*** digital signal -- only *two* possible values
  - Typically represented as **0** and **1**
  - One *b*inary dig*it* is a ***bit***
  - We'll only consider *binary* digital signals
  - Binary is popular because
    - Transistors, the basic digital electric component, operate using *two* voltages
    - Storing/transmitting one of *two* values is easier than three or more (e.g., loud beep or quiet beep, reflection or no reflection)

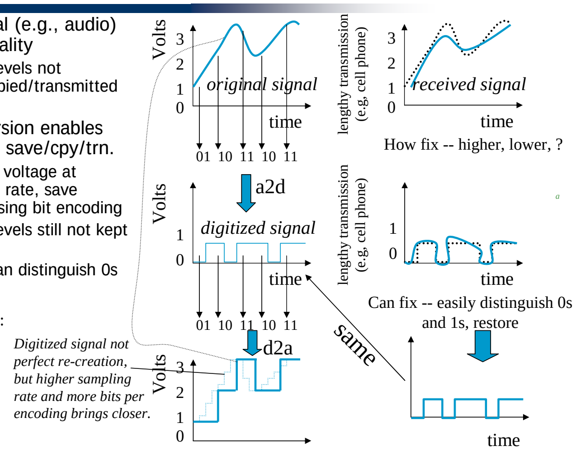


Digital Logic – Introduction

Example of Digitization

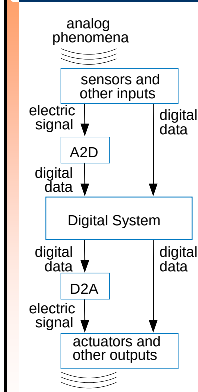
- Analog signal (e.g., audio) may lose quality
  - Voltage levels not saved/copied/transmitted perfectly
- Digitized version enables near-perfect save/cpy/trn.
  - "Sample" voltage at particular rate, save sample using bit encoding
  - Voltage levels still not kept perfectly
  - But we can distinguish 0s from 1s

Let bit encoding be:
1 V: "01"
2 V: "10"
3 V: "11"

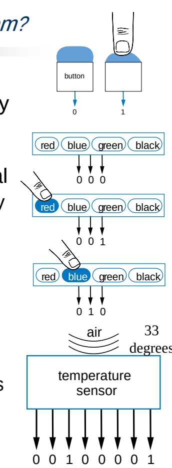


Digital Logic – Introduction

How Do We Encode Data as Binary for Our Digital System?

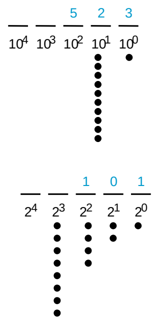


- Some inputs inherently binary
  - Button: not pressed (0), pressed (1)
- Some inputs inherently digital
  - Just need encoding in binary
  - e.g., multi-button input: encode red=001, blue=010, …
- Some inputs analog
  - Need analog-to-digital conversion
  - As done in earlier slide -- sample and encode with bits



Digital Logic – Introduction
How to Encode Numbers: Binary Numbers

- Each position represents a quantity; symbol in position means how many of that quantity
  - Base ten (***decimal***)
    - Ten symbols: 0, 1, 2, …, 8, and 9
    - More than 9 -- next position
      - So each position power of 10
    - Nothing special about base 10 -- used because we have 10 fingers
  - Base two (***binary***)
    - Two symbols: 0 and 1
    - More than 1 -- next position
      - So each position power of 2



Digital Logic – Introduction
How to Encode Numbers: Binary Numbers

- Working with binary numbers
  - In base ten, helps to know powers of 10
    - one, ten, hundred, thousand, ten thousand, …
  - In base two, helps to know powers of 2
    - one, two, four, eight, sixteen, thirty two, sixty four, one hundred twenty eight
      - (Note: unlike base ten, we don't have common names, like "thousand," for each position in base ten -- so we use the base ten name)

10^3 10^2 10^1 10^0

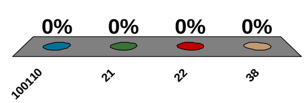
1000 100 10 1

2^9 2^8 2^7 2^6 2^5 2^4 2^3 2^2 2^1 2^0

512 256 128 64 32 16 8 4 2 1

Digital Logic – Introduction
Converting from Decimal to Binary

- What is the value of the binary number 100110 in decimal?
  1. 100,110
  2. 21
  3. 22
  4. 38



Digital Logic – Introduction
Converting from Decimal to Binary Numbers

- Subtraction Method (Easy for Humans)
  - *Goal:* Get the binary weights to add up to the decimal quantity
    - Work from left to right
      - (Right to left – may fill in 1s that shouldn't have been there – try it). Subtraction method
    - Subtract a selected binary weight from the (remaining) quantity
      - Then, we have a new remaining quantity, and we start again (from the present binary position)
    - Stop when remaining quantity is 0

Remaining quantity: 12

__ __ __ __ __ __
32 16 8 4 2 1

1 __ __ __ __ __ — 32 is too much
32 16 8 4 2 1

0 1 __ __ __ __ — 16 is too much
32 16 8 4 2 1

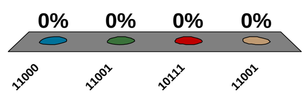
0 0 1 __ __ __ — 12 – 8 = 4
32 16 8 4 2 1

0 0 1 1 __ __ — 4-4=0 DONE
32 16 8 4 2 1

0 0 1 1 0 0 — answer
32 16 8 4 2 1

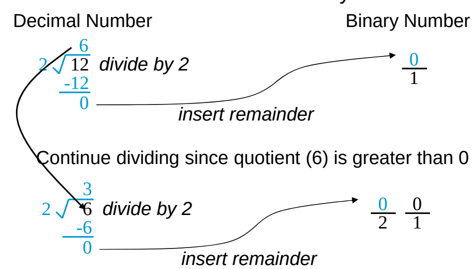
Digital Logic – Introduction
Converting from Decimal to Binary

- What is the value of the decimal number 25 in binary?
  1. 11000
  2. 11001
  3. 10111
  4. 011001



Digital Logic – Introduction
Converting from Decimal to Binary

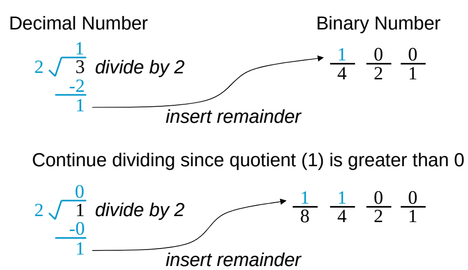
- Division Method (Good for Computers)
  - Divide decimal number by 2 and insert remainder into new binary number.
  - Continue dividing quotient by 2 until the quotient is 0.
- Example: Convert decimal number 12 to binary



Continue dividing since quotient (3) is greater than 0

Digital Logic – Introduction
Converting from Decimal to Binary

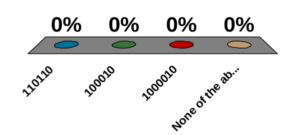
- Example: Convert decimal number 12 to binary (continued)



Since quotient is 0, we can conclude that 12 is 1100 in binary

Digital Logic – Introduction
Converting from Decimal to Binary

- What is the value of the decimal number 54 in binary?
  1. 110110
  2. 100010
  3. 1000010
  4. *None of the above*



Digital Logic – Introduction
Hexadecimal Numbers

8 A F
16^4 16^3 16^2 16^1 16^0

8 A F
1000 1010 1111

hex	binary	hex	binary
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111

- Nice because each position represents four base two positions
  - Used as compact means to write binary numbers
- Known as ***hexadecimal***, or just ***hex***

Convert 11110000 to hex:
F 0

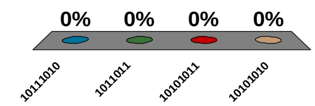
17

Digital Logic – Introduction
Converting from Hexadecimal to Binary

- What is the value of the hexadecimal number AB in binary?

1. 10111010
2. 01011011
3. 10101011
4. 10101010

hex	binary	hex	binary
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111



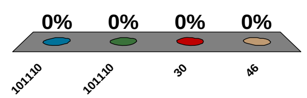
18

Digital Logic – Introduction
Converting from Hexadecimal to Decimal

- What is the value of the hexadecimal number 2E in decimal?

1. 101110
2. 00101110
3. 30
4. 46

hex	binary	hex	binary
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111



19