Overview

• Wednesday, June 14th, 12–3 p.m.
• Same policy as before: no notes, books or calculators
• **Extra Credit**
  – If score is over 100% will still help final grade
  – One Problem **mid-term**
• **Show your work**
  • No credit if we can’t figure out how you did it
  • Partial credit
Partial List of Topics

• N-type, p-type transistors
• Realization of truth table from transistors and inverse
• Transistors to standard gates and inverse
• Truth table to gates and inverse

→ Sum-of-Products and Product-of-Sums
• Boolean Algebra
• Common logic elements
  – Mux etc including sequential

• Binary representations
  – Unsigned
    • Binary
    • Hex
    • Octal
  – Signed
    • Two’s complement
    • One’s complement
    • Sign magnitude
  – Bias

• Binary Math
  – Overflow indications
Partial List of Topics Continued

- Floating Point
  - Conversions
  - Addition, Subtraction, Multiplication

- LC-3 Architecture

- LC-3 Assembly
  - Op-code translation
  - LC-3 coding and running
  - Subroutine methods
  - Basic data structures

- MIPS
  - Role of registers
  - Theory of Function Calls

- Embedded Uno32
  - I/O
  - Special Purpose Registers (TRIS, etc.)

- No code
Transistor and Gates
Truth Table to Gates

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Transistor and Gates
Truth Table to Transistors

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Transistors and Gates

Transistors to truth Table

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
Logic Elements to Gates

• Draw the gate level diagram of a 2-4 decoder
## Base Conversion Table 8 bits

<table>
<thead>
<tr>
<th>Decimal</th>
<th>1’s Complement</th>
<th>2’s Complement</th>
<th>Signed Magnitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>-35</td>
<td>X</td>
<td>0110 0001</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>1001 1101</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>X</td>
<td>10011101</td>
<td></td>
</tr>
</tbody>
</table>
Binary Arithmetic

- Unsigned
  - 1001 1111
  - 0011 0101

- 2’s Complement
  - 0111 0101
  - 1111 0001

- Signed Magnitude
  - 0110 0101
  + 0001 1101
Fractional Representation

• 4.6 in Base 2?
Binary Division

• 101 | 101110111
Floating Point Format

Representation:

31 30  23  22  0
  S   E   F

- S is one bit representing the sign of the number
- E is an 8 bit biased integer representing the exponent
- F is an 23-bit unsigned integer

The true value represented is: \((-1)^S \times f \times 2^e\)

- S = sign bit
- e = E – bias
- f = F/2^n + 1
- for single precision numbers n=23, bias=127
\[-476\]

\[
\begin{align*}
47 - 32 &= 15 \\
47 - 8 &= 39 \\
-47 - 8 &= 101111
\end{align*}
\]

\[
\begin{align*}
.6 \times 2 &= 1.2 \\
.2 \times 2 &= 0.4 \\
.4 \times 2 &= 0.8 \\
.8 \times 2 &= 1.6 \\
.6 \times 2 &= 1.2
\end{align*}
\]

\[
\begin{align*}
1.0111111001 & \times 2^{20} \\
5 + 127 &= 132 \\
10000100 & \rightarrow 01111100110011001100110011001100
\end{align*}
\]
### 2 to 1 Mux

<table>
<thead>
<tr>
<th>S</th>
<th>A</th>
<th>B</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
\[
\begin{array}{c}
\text{128 + 16 + 2} \\
\text{146 - 127} \\
\hline
\text{19}
\end{array}
\]

\[
\times 49742400
\]

0 100 100 1 0 111 0 100 0 0 10 0 100 → 0

\[
1.1110100001001 \times 2^{19}
\]

11.101 1 \times 2^{5}
1.101101

\[ 1 + \frac{1}{2} + \frac{1}{8} + \frac{1}{16} + \frac{1}{64} \]

.101

.5 + .125
Floating Point Conversion

- What is the decimal value for this SP FP number 0x421A 0000?
\[
3180000 \times \text{C3B50000}
\]

\[
\begin{array}{c}
0011111100011000 \\
1.0011
\end{array}
\]

\[
\begin{array}{c}
1100001110110101 \\
1.0110101
\end{array}
\]

\[
64 + 32 + 16 + 8 + 4 + 2
\]

\[
126 = -1
\]

\[
128 + 4 + 2 + 1 = 135 - 127 = 8
\]

\[-1 + 8 = 7\]
Binary Arithmetic

- Unsigned
  \[
  \begin{array}{c}
  1001 1111 \\
  0011 0101 \\
  \hline
  1110 0101 \\
  \end{array}
  \]

- 2’s Complement
  \[
  \begin{array}{c}
  0111 0101 \\
  1111 0001 \\
  \hline
  1001 1110 \\
  \end{array}
  \]

- Signed Magnitude
  \[
  \begin{array}{c}
  0110 0101 \\
  + 0001 1101 \\
  \hline
  1011 1001 \\
  \end{array}
  \]
Arbitrary Base Conversion

• $1210_3$ in base 10
LC-3 Sub-Routine Coding

• Write the Load Function from Lab 4
  – Assume R0,R1,R2 are used as arguments
  – Be sure to save off registers used
  – Label Base has first address of array and Label Size holds the column length
1. (10pts) LC-3 ISA

After the following LC-3 code executes what are the ending contents of the registers and memory? Assume some registers/memories have starting values as indicated. If blank, the content is unknown. Remember that both registers and memory locations are 16-bits wide. The memory portion starts at address 0x3200.

```
LEA    R1, label0
LDR    R2, R1, #0
STR    R0, R1, #4
LEA    R6, label12
ADD    R5, R0, R1
LEA    R0, label1
AND    R7, R2, R5
NOT    R3, R0
STR    R7, R6, #-2
STR    R2, R1, #1
```

<table>
<thead>
<tr>
<th>Register</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>R0</td>
<td>0x1234</td>
</tr>
<tr>
<td>R1</td>
<td></td>
</tr>
<tr>
<td>R2</td>
<td></td>
</tr>
<tr>
<td>R3</td>
<td></td>
</tr>
<tr>
<td>R4</td>
<td></td>
</tr>
<tr>
<td>R5</td>
<td></td>
</tr>
<tr>
<td>R6</td>
<td></td>
</tr>
<tr>
<td>R7</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Memory</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Label0</td>
<td>0xDEAD</td>
</tr>
<tr>
<td>Label1</td>
<td></td>
</tr>
<tr>
<td>label12</td>
<td></td>
</tr>
</tbody>
</table>
PIC32 Architecture

– Word size, register count, similar to questions asked about the lc-3
– Standard registers available and their uses
PIC32 Ports

- What do TRIS LAT and PORT do?
- Pseudocode on how to use them
PIC32 Function Calls
**PoS and SoP**

\[(A + B + C)(A + \overline{B} + C)(\overline{A} + \overline{B} + C)\]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
**PoS and SoP**

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[
\begin{align*}
\hat{A} \bar{B} \bar{C} &+ \hat{A} \bar{B} C + \hat{A} B \bar{C} + \hat{A} B \bar{C} + A \bar{B} C + \bar{A} B C \\
= &\quad \frac{1}{2} \quad \frac{1}{2} \\
\bar{A} c (\bar{B} + \bar{B}) + A \bar{B} (\bar{C} + C) + A B C \\
\hat{A} C + A \bar{B} + A B C \\
\hat{A} C + A (\bar{B} + B C) \\
\hat{A} C + A (\bar{B} + C) \\
\hat{A} C + A \bar{B} + A C \\
(\hat{A} + A) C + A \bar{B} C + A \bar{B}
\end{align*}
\]
Bias of 7

$-4 \rightarrow B_2$

$-4 + 7 = 3$

$0011$

$1010$

$8 + 2 = 10$

$10 - 7 = 3$
MIPS Function Call

Save:
push $0-7
    ra
    t
    jal printE

Pop $0-7
Pop ry

Restore: