CMPE12 Final Review
Overview

• Final is March 15th 12PM-3PM
• Same policy as before: no notes, books or calculators
• Extra Credit
  – If score is over 100% will still help final grade
  – Two Problems
    • One repeat from the midterm exactly
    • One random one
  – 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

- PIC32 Architecture
  - General information about the system
  - Not required to write MIPS assembly
  - Understanding how the ports work
  - Conceptual understanding of function calls
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** = \(E - \text{bias}\)
- **f** = \(F/2^n + 1\)

- For single precision numbers \(n=23\), bias=127

\[
\frac{1}{\sqrt{2^{23}}} \times (1.10101010101010101010101)_2
\]
Floating Point Conversion

- What is the decimal value for this SP FP number 0x4228 0000?

\[
\begin{align*}
100 & 00100 \\
128 + & 4 - 127 \\
5 & \\
1.0101000 & \times 2^{5} \\
101010 & \\
42 & 
\end{align*}
\]
Floating Point Math

\[ 0x45\text{FFC000} + 0x45\text{6B0000} \]

\[
\begin{array}{cccccccc}
0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 1 & 0 & 1 & 0 & 1 & 1 & 1
\end{array}
\]

\[
\begin{array}{cccccccc}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 0
\end{array}
\]

\[
\begin{array}{cccccccc}
0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 1 & 0 & 1 & 1 & 0 & 1 & 0
\end{array}
\]

\[
\begin{array}{cccccccc}
0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}
\]

\[
\begin{array}{cccccccc}
0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 0
\end{array}
\]

\[
\begin{array}{cccccccc}
0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}\]

\[
0x463\text{AA0000}
\]
LC-3 Assembly Coding

- Write LC-3 assembly code that will OR the values in R1 with R3 and store the result in R0.

\[
\begin{align*}
A + B & \quad \text{NOT} \quad R1, R1 \\
\overline{A} \overline{B} & \quad \text{NOT} \quad R3, R3 \\
\text{AND} & \quad R0, R3, R1 \\
\text{NOT} & \quad R0, R0
\end{align*}
\]
LC-3 Sub-Routine Coding

- Write the Load Function from Lab 6
  - 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

\[ R0 = \text{Mem}[R1 \cdot \text{Size} + \text{Col} + \text{Base}] \]
; load Routine
; save off registers used
LEA R0, Base
ADD R0, R0, R2
AND R1, R1, R1
BRz Row0
LD R1, Size
Row0: ADD R0, R0, R1
LDR R0, R0, 0
RET
LC-3 Data Structures
Array, Stacks and Queues

• Basic theory of each
  – Understanding of how to write basic routines

Push
ADD R6, R6, -1
STR R0, R6, 0

Push
ADD R6, R6, #-1 ; decrement stack ptr
STR R0, R6, #0 ; store data (R0)

Pop
LDR R6, #0 ; load data from TOS
ADD R6, R6, #1 ; increment stack ptr
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>x1234</td>
</tr>
<tr>
<td>R1</td>
<td>x3200</td>
</tr>
<tr>
<td>R2</td>
<td>xDEAD</td>
</tr>
<tr>
<td>R3</td>
<td>x3D5B</td>
</tr>
<tr>
<td>R4</td>
<td></td>
</tr>
<tr>
<td>R5</td>
<td>x4434</td>
</tr>
<tr>
<td>R6</td>
<td>x3207</td>
</tr>
<tr>
<td>R7</td>
<td>x4424</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Label</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Label0</td>
<td>xDEAD xDEAD</td>
</tr>
<tr>
<td>Label1</td>
<td>x1234 x4424</td>
</tr>
<tr>
<td>Label2</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

\[ 32 \quad 32 \]
\[ t0 \]
\[ 50 \]
\[ \text{Zero} \]
PIC32 Ports

- What do TRIS LAT and PORT do?
- Pseudocode on how to use them

Turn E4 into output and set High without disturbing

t0 = 0b10000 // mask

TRISCLR = t0
LATESET = t0
PIC32 Function Calls

L<3 subroutine foo:

PIC32 function foo:

Stack foo ↓

foo memory
$16 - 10$

$\times 4180 \to 0 \times 4112$

$\overrightarrow{01000001} \overrightarrow{100000} \overrightarrow{0}$

$\overrightarrow{0100000010} \overrightarrow{0}$

$\overrightarrow{0100000010} \overrightarrow{0}$

$1.0000 \rightarrow 0$

$1.0100 \rightarrow 0$

$0.000111$

$1.0100 \rightarrow 0$

$0.0111 \times 2^4$

$0.1011 \times 2^4$

$1.1 \cdot 2^2$

$-0.1001 \times 2^4$

$0.0111 \times 2^4$

$0.10000001 \left(22 \text{ zeros}\right)$

$\times 40 < 000$
\[
\begin{align*}
16.10 \times 4180 & \rightarrow 0 \\
& \begin{array}{cccc}
4 & 0 & 0 & 0 \\
3 & 0 & 1 & 0 \\
\hline
7134 & 0000 & 0110 & 0010 \\
\end{array} \\
& \begin{array}{cccc}
1.00 \\
1.01 \\
\hline
1.00 \\
10000 \\
\hline
1.0100 \\
\end{array} \\
& 0.01000 \rightarrow 0 \times 1.0100 (a 10 \text{ } + \text{ more})
\end{align*}
\]
1) E. (Commons is wrong for grade calculations
   
2) there is a shift
   96%  100%

3) grade boundaries not even
   10% A < B
   15% B < C